Front. Electr. Electron. Eng. China 2011, 6(1): 86–119
DOI 10.1007/s11460-011-0135-1

RESEARCH ARTICLE

Lei XU

Codimensional matrix pairing perspective of BYY harmony learning: hierarchy of bilinear systems, joint decomposition of data-covariance, and applications of network biology

Abstract One paper in a preceding issue of this journal has introduced the Bayesian Ying-Yang (BYY) harmony learning from a perspective of problem solving, parameter learning, and model selection. In a complementary role, the paper provides further insights from another perspective that a co-dimensional matrix pair (shortly co-dim matrix pair) forms a building unit and a hierarchy of such building units sets up the BYY system. The BYY harmony learning is re-examined via exploring the nature of a co-dim matrix pair, which leads to improved learning performance with refined model selection criteria and a modified mechanism that coordinates automatic model selection and sparse learning. Besides updating typical algorithms of factor analysis (FA), binary FA (BFA), binary matrix factorization (BMF), and nonnegative matrix factorization (NMF) to share such a mechanism, we are also led to (a) a new parametrization that embeds a de-noise nature to Gaussian mixture and local FA (LFA); (b) an alternative formulation of graph Laplacian based linear manifold learning; (c) a co-decomposition of data and covariance for learning regularization and data integration; and (d) a co-dim matrix pair based generalization of temporal FA and state space model. Moreover, with help of a co-dim matrix pair in Hadamard product, we are led to a semi-supervised formation for regression analysis and a semi-blind learning formation for temporal FA and state space model. Furthermore, we address that these advances provide with new tools for network biology studies, including learning transcriptional regulatory, Protein-Protein Interaction network alignment, and network integration.

Received December 1, 2010; accepted January 7, 2011

Lei XU (✉)
Department of Computer Science and Engineering, The Chinese University of Hong Kong, Hong Kong, China
E-mail: lxu@cse.cuhk.edu.hk

Keywords Bayesian Ying-Yang (BYY) harmony learning, automatic model selection, bi-linear stochastic system, co-dimensional matrix pair, sparse learning, de-noise embedded Gaussian mixture, de-noise embedded local factor analysis (LFA), bi-clustering, manifold learning, temporal factor analysis (TFA), semi-blind learning, attributed graph matching, generalized linear model (GLM), gene transcriptional regulatory, network alignment, network integration

1 Introduction

Firstly proposed in 1995 and systematically developed over a decade, the Bayesian Ying Yang (BYY) harmony learning provides not only a general framework that accommodates typical learning approaches from a unified perspective but also a new road that leads to improved model selection criteria, Ying-Yang alternative learning with automatic model selection, as well as coordinated implementation of Ying based model selection and Yang based learning regularization. In one preceding issue of this journal [1], one paper provided the fundamentals of the BYY harmony learning, the basic implementing techniques, and a tutorial on algorithms for typical learning tasks.

As shown by Fig. 2(d) in Ref. [1], an intelligent system is believed to jointly perform a mapping $X \to R$ that projects a set X of observations into its corresponding inner representation R and also a mapping $R \to X$ that reconstructs or interprets X from the inner representation R, which are described via the joint distribution of X, R in two types of Bayesian decompositions:

$$\begin{aligned} \text{Ying}: q(X,R) &= q(X|R)q(R), \\ \text{Yang}: p(X,R) &= p(R|X)p(X). \end{aligned} \tag{1}$$

In a complement of the Ying-Yang philosophy, we call

this pair BYY system. The Ying is primary, and the probabilistic structures of $q(X|R)$ and $q(R)$ come from the natures of learning tasks. The Yang is secondary, and $p(X)$ comes from a set of observation samples, while the probabilistic structure of $p(R|X)$ is a functional with $q(X|R), q(R)$ as its arguments, designed from Ying according to a Ying-Yang variety preservation principle. A Ying- Yang best harmony principle is proposed for learning all the unknowns in the system, mathematically implemented by maximizing the harmony functional:

$$H(p\|\, q) = \int p(R|X)p(X)\ln[q(X|R)q(R)]\mathrm{d}R\mathrm{d}X. \quad (2)$$

From the standard perspective of Ref. [1], the inner representation R of a set of observation samples $X = \{x_t\}$ consists of three types that correspond to three inverse problems, as shown by Fig. 2 in Ref. [1]. One is $Y = \{y_t\}$ of the inner coding vector y_t per sample x_t. Usually, the mapping $X \to Y$ performs problem solving, called the first level inverse problem. The second type of inner representation consists of a parameter set Θ, and the mapping $X \to \Theta$ is the second level inverse problem called parameter learning. The third type is $\boldsymbol{k}$ that consists of one or several integers that closely relate to the complexities of Y and Θ, and $X \to \boldsymbol{k}$ is the third level inverse problem called model selection. Further details are referred to Sect. 1.1 in Ref. [1]. Instead of this standard perspective, this paper provides further insights on the BYY harmony learning from a new perspective that a co-dimensional matrix pair (shortly co-dim matrix pair) forms a building unit and a hierarchy of such building units sets up the BYY system.

Two matrices A and Y are regarded as a co-dim matrix pair if they share a same rank m. Such a matrix pair forms a building unit $\eta^x(Y, A)$ via a simple combination, e.g., a matrix product or a Hadamard product. One typical building unit is a product $\eta^x(Y, A) = AY$ with its rank m being a common dimension or shortly co-dimension shared by all the row vectors of A and also all the column vectors of Y. This building unit is a stochastic system, featured by A that is also a stochastic matrix. With a stochastic matrix Y as its input, it outputs $\boldsymbol{\eta}^x = \eta^x(Y, A)$ that is observed as a data matrix X subject to residuals $E = X - \boldsymbol{\eta}^x$ with

$$\begin{aligned} q(E|\Psi^x) &= \prod_t q(e_t|\Psi^x),\ e_t = x_t - \eta_t^x, \quad (3)\\ X &= [x_1, \cdots, x_N],\ E = [e_1, \cdots, e_N],\\ \boldsymbol{\eta}^x &= [\eta_1^x, \cdots, \eta_N^x],\\ \mathcal{E}[e_t] &= 0,\ \mathcal{E}[\boldsymbol{\eta}^x E^{\mathrm{T}}] = \mathcal{E}[\boldsymbol{\eta}^x]\mathcal{E}^{\mathrm{T}}[E] = 0,\\ \mathcal{E}[e_t e_t^{\mathrm{T}}] &= \Psi^x = \mathrm{diag}[\psi_1^x, \psi_2^x, \cdots, \psi_d^x], \end{aligned}$$

where $\mathcal{E}[u]$ denotes the expectation of u. That is, elements of E are mutually uncorrelated not only among all its elements but also with $\boldsymbol{\eta}^x$.

For $\boldsymbol{\eta}(Y, A) = AY$ in particular, as illustrated at the center of Fig. 1, the columns of A forms a coordinate system in the observation space, and we are lead to the following bi-linear stochastic system :

$$X = AY + E. \quad (4)$$

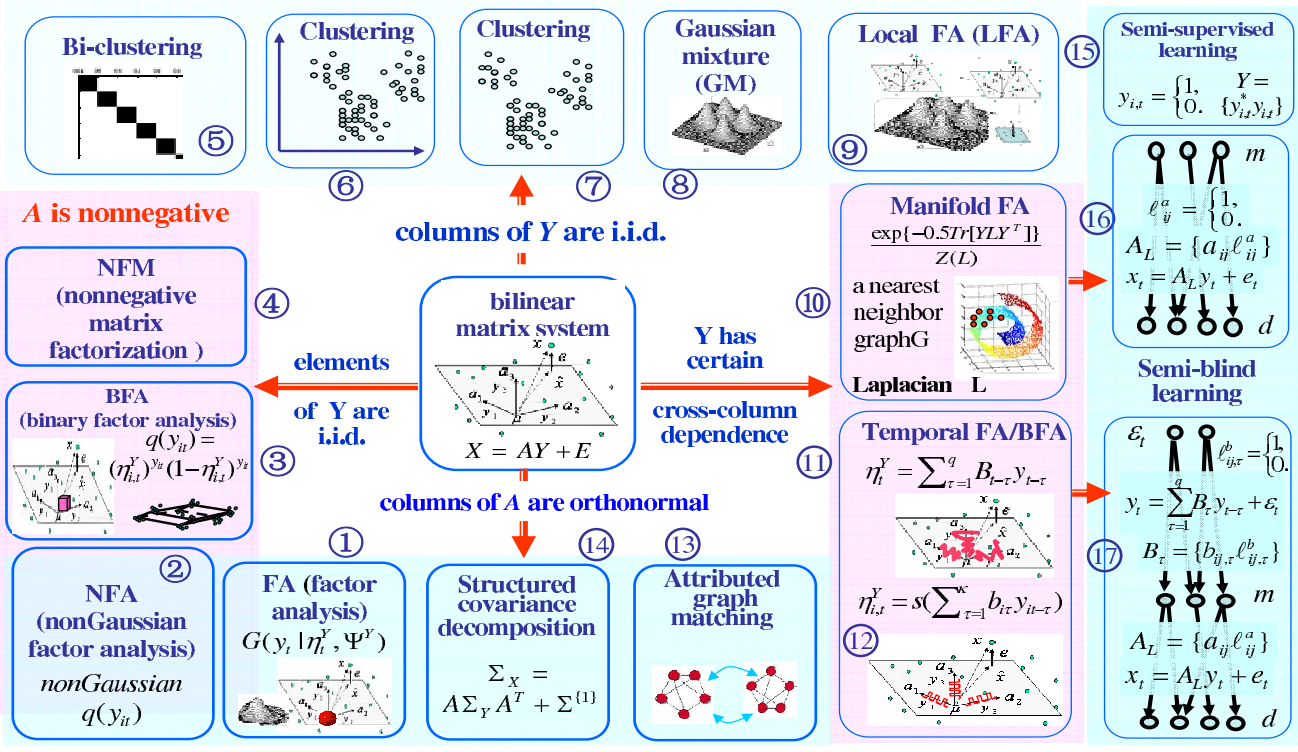

Fig. 1 Family of typical learning tasks from a perspective of the stochastic bilinear matrix system

To be further addressed in Sect. 2.1, typical learning tasks are revisited when different constraints are imposed on Y, A, and X. It follows from the Boxes ①–③ in Fig. 1 that we are led to a family of FA [2–4] and independent FA extensions [5–18], and from the Boxes ④–⑥ that we are led to a family of nonnegative matrix factorization (NMF) [19–28]. Also, we are led to not only a new parametrization that embeds a de-noise nature to Gaussian mixture [29–36] as shown in the Boxes ⑦–⑧, but also an alternative formulation of graph Laplacian based manifold learning [37,38] as shown in the Box ⑩.

Extensive efforts have been made on learning $X = AY + E$ under the principle of the least square errors (i.e., minimizing $\mathrm{Tr}[EE^{\mathrm{T}}]$), or generally the principle of maximizing the likelihood $\ln q(X|\Theta)$ to estimate Θ of unknown parameters with a probabilistic structure $q(X|\Theta)$. One major limitation is that the rank of Y needs to be known in advance. The problem is tackled with the help of the Bayesian approach in two typical ways. One is directly maximizing

$$\ln[q(X|\Theta)q(\Theta|\Xi)], \tag{5}$$

with help of a priori $q(\Theta|\Xi)$, e.g., as encountered in learning Gaussian mixture by minimum message length (MML) [39,40], and in sparse learning that prunes away extra weights by a Laplace prior $q(\Theta|\Xi)$ for a regression or interpolation task [41–43]. However, a choice of $q(\Theta|\Xi)$ directly affects the estimation of Θ, and thus the performance is sensitive to whether an appropriate priori $q(\Theta|\Xi)$ is available. The other way is maximizing an approximation of the marginal likelihood

$$q(X) = \int q(X|\Theta)q(\Theta|\Xi)\mathrm{d}\Theta, \tag{6}$$

e.g., Bayesian inference criterion (BIC) [44], minimum description length (MDL) [45,46], and variational Bayes [47–49]. Details are referred to Sect. 2.1 of Ref. [1].

As shown in Fig. 5 of Ref. [1], the BYY harmony learning on Eq. (4) leads to improved model selection via either or both of improved selection criteria and Ying-Yang alternative learning with automatic model selection, with help of not only the role of $q(\Theta|\Xi)$ as above but also the role of $q(Y|\Theta^y)$. In this paper, as to be stated in Sect. 2.2, the BYY harmony learning is made on a BYY system by Eq. (1) with

$$q(R) = q(Y - \boldsymbol{\eta}^y|\Psi^y)q(A - \boldsymbol{\eta}^a|\Psi^a)q(\Upsilon), \tag{7}$$

which differs from the following one in Ref. [1]:

$$q(R) = q(Y|\Theta^y)q(\Theta). \tag{8}$$

That is, A is taken out of $\Theta = A \cup \Upsilon$ and is considered in a paring with $q(Y - \boldsymbol{\eta}^y|\Psi^y)$ in order to explore the nature of co-dimension matrix pair A, Y. We are further led to improved learning performances with refined model selection criteria and an interesting mechanism that coordinates automatic model selection and sparse learning.

Complementarily, the data decomposition by Eq. (4) is associated with a decomposition of the data covariance S_X into the covariance S_Y of Y and the covariance Σ of E in a quadratic matrix equation

$$S_X = AS_YA^{\mathrm{T}} + \Sigma. \tag{9}$$

There are also typical tasks that aim at this decomposition with X unavailable but S_X and S_Y available. Illustrated in the Boxes ⑬–⑭ at the bottom of Fig. 1, one type of such tasks is encountered by graph isomorphism and attributed graph matching [50–54], where S_X and S_Y describe two unidirectional attributed graphs, while A is a permutation matrix and Σ stands for matching errors. The other type of tasks comes from the signal processing literature [55,56], where S_X is a positive semi-definite Toeplitz matrix, S_Y is diagonal, and A is particularly structured with every element in a form of $\exp[j(k-1)\omega_l]$.

Given E that satisfies Eq. (3), making data decomposition by Eq. (4) implies the decomposition by Eq. (9), while making Eq. (9) also leads to Eq. (4) if we also have one additional condition that A is orthogonal and $\Sigma = \sigma^2 I$, as encountered in principal component analysis (PCA) [2,13]. In practice, the additional condition may not hold, which is alternatively enhanced via making both the decompositions by Eq. (4) and Eq. (9). Moreover, this co-decomposition provides a formulation that integrates different data types, namely X and S_X. Also, making the decomposition by Eq. (9) can be regarded as imposing a structural regularization on learning the model by Eq. (4).

According to the natures of learning tasks, the building unit by Eq. (4) may further get supported from an upper layer. In addition to the standard way of using a prior $q(\Theta|\Xi)$ in Eq. (8), either or both of $\boldsymbol{\eta}^y, \boldsymbol{\eta}^a$ may also itself be the output of another co-dim matrix pair e.g., in a format of Eq. (4), which may be regarded as structural priors. Moreover, either or both of Ψ^y and Ψ^a may itself be the output of another co-dim matrix pair in a format of Eq. (9). So on and so forth, one new layer may have its next new layer. Whether we add a new upper layer depends on if there is some priors available or we want to simplify computation. As a whole, a BYY system is featured by a hierarchy of co-dim matrix pairs.

To be specific, we consider two typical examples featured with a two layer hierarchy. With $\boldsymbol{\eta}^a = \eta^a(\boldsymbol{\zeta}, \Phi)$, the de-noise nature of the above new parametrization is also embedded to a Gaussian mixture within a dimension reduced subspace and further to local FA [57–67] as illustrated in the Box ⑨. With $\boldsymbol{\eta}^y = \eta^y(\boldsymbol{\varepsilon}, B)$, we are further led to a co-dim matrix pair based generalization

of temporal FA and structural state space model [68–75], as illustrated in the Boxes ⑪ and ⑫.

Featured with merely data X available, all the above discussed tasks belong to what called unsupervised learning on Eq. (4). With both X and its corresponding Y available, the problem becomes linear regression analysis or a special example of supervised learning on Eq. (4). There are also many practical cases that are somewhere a middle of the two ends. One example is that the corresponding columns of both X and Y are partially known in addition to merely having X available. Another example is encountered on studying what called networks component analysis (NCA) for transcriptional regulation networks (TRN) in molecular biology, where A is known to be sparse and the locations of zero elements are known [76–79]. In fact, the two examples are instances of a general scenario that we know not only the output observations X of a system but also partially either or both of the input Y and the system (i.e., A and the property of E).

Instances of this scenario were also encountered in the signal processing studies on the linear convolution system, a special type of $X = AY + E$. The term blind deconvolution [80,81] refers to the tasks of estimating unknowns only from its output observations X, while semi-blind deconvolution [82] refers to the cases that we know partially either or both the system and its input. Moreover, instances of this scenario are also found in those efforts made under the term semi-supervised learning [83] for pattern classification. The columns of X are observed patterns from the outputs of a system that generates samples of selected classes, based on the input Y with its columns indicating which classes to select. We observe that semi-blind deconvolution and semi-supervised learning share a similar concept but differ in a specific system and specific types of input and output. Probably, semi-blind learning is a better name for efforts that put attention on the general scenario of knowing partially either or both of system and input.

In Ref. [1, Sect. 4.4], the BYY system is shown to provide a unified framework to accommodate various cases of semi-supervised learning. To be stated in Sects. 3.2 and 4.3, we are further led to a general formulation for semi-blind learning. As illustrated in Fig. 1 from the Box ⑮ to the Box ⑰, letting Y in Eq. (4) to be supported from its upper layer by a Hadamard product of co-dim matrix pair, we are lead to a formation of semi-supervised learning for regression analysis with a nature of automatic selection on variables; while letting A to be supported from its upper layer by another Hadamard product of co-dim matrix pair, we are lead to a formation of semi-blind learning for Eq. (4) that covers the above mentioned NCA [76–79] as a special case. This formation is further generalized for temporal modeling, with Y supported by $\eta^y(\varepsilon, B)$ and then B further supported by a Hadamard product of another co-dim matrix pair.

Last but not least, this paper also explores molecular biology applications of the advances achieved from the new perspective of the BYY harmony learning.

The existing studies on molecular networks rely on technologies available for data gathering, featured by two waves. The first is driven by a large number of "genome" projects on transcriptome mechanisms and particularly TRN in the past two decades. In Sect. 5.2, the past TRN studies will be summarized in three streams of advances, and further progresses are suggested in help with the co-dim matrix pair perspective of the BYY harmony learning on $X = AY + E$, especially the general formulation for semi-blind learning and its extension for temporal modeling.

The second wave is featured by the term interactome, due to recent large-scale technologies for measuring protein-to-protein interactions (PPIs) [84]. PPI data are represented by undirectional networks or graphs, and two major tasks on PPI data are graph partitioning for module detection and graph matching for network alignment. Recently, a BYY harmony learning based biclustering algorithm has also been developed for graph partitioning and shown favorable performances in comparison with several well known clustering algorithms for the same purpose [28]. Further improvements are suggested from the co-dim matrix pair perspective in this paper. Moreover, the problem of network alignment is also taken in consideration with graph matching algorithms from the perspective of Eq. (9) with help of the BYY harmony learning.

Additionally, there are several data sources available for the studies of transcriptome mechanisms, which lead to different networks and thus arise the needs of network integration. A similar scenario is also encountered for the studies of interactome mechanisms. Actually, two domains of mechanisms are related too. Therefore, network integration becomes increasingly important in the current network biology studies [85]. The problem of network integration is closely coupled with network alignment, and the co-decomposition by Eq. (4) and Eq. (9) provides a potential formulation for integrating data types across the domains.

The rest of this paper is arranged as follows. Section 2 starts from a bi-linear stochastic system and its post-linear extensions, together with a brief outline of typical learning tasks it covers. Then, a joint consideration of a co-dim matrix pair is shown to further improve the BYY harmony learning, with a new mechanism that coordinates automatic model selection and sparse learning. In Sect. 3, we get further insights on this perspective of the BYY harmony learning via examples based on Eq. (4). In addition to updating typical algorithms of the FA and NFA families to share such a mechanism, we suggest a new parametrization that embeds a de-

noise nature to Gaussian mixture and variants, and an alternative formulation of graph Laplacian based linear manifold learning. Then, taking Eq. (9) also in consideration, we are led to algorithms for attributed graph matching and a co-decomposition of data and covariance. In Sect.4, we proceed to a general formulation of a BYY harmony learning with a hierarchy of several co-dim matrix pairs. The de-noise parametrization has been further extended to local FA, and the co-dim matrix pairing nature has been generalized to temporal FA and state space modeling. Moreover, with help of a co-dim matrix pair in Hadamard product, we are lead to a general formation of semi-blind learning. Finally, section 5 further addresses that these advances provide with new tools for network biology applications, including learning TRN, PPI network alignment, and network integration.

2 Co-dimensional matrix-pairing perspective of BYY harmony learning

2.1 Learning post bi-linear system and model selection

This subsection introduces the probabilistic structures for $q(X|\boldsymbol{\eta}^x, \Psi^x)$, $q(Y|\Theta^y)$ and $q(A|\boldsymbol{\eta}^a, \Psi^a)$, and related fundamental issues, including typical learning tasks, indeterminacy problems, and model selection issues.

Equivalently, $q(E|\Psi^x)$ in Eq. (3) can be rewritten into

$$\begin{aligned} X &= \boldsymbol{\eta}^x + E, \\ q(X|R) &= q(X|\boldsymbol{\eta}^x, \Psi^x) = q(X - \boldsymbol{\eta}^x|\Psi^x), \\ &= \textstyle\prod_t q(x_t - \eta_t^x|\Psi_t^x) = \prod_t q(x_t|\eta_t^x, \Psi_t^x), \\ \eta_t^x &= \mathcal{E}[x_t], \\ \Psi_t^x &= \mathcal{E}[(x_t - \eta_t^x)(x_t - \eta_t^x)^{\mathrm{T}}] = \mathcal{E}[e_t e_t^{\mathrm{T}}] \text{ by Eq.(4)}, \end{aligned} \tag{10}$$

where the nations $q(u|\boldsymbol{\eta}^u, \Psi^u)$ and $q(u - \boldsymbol{\eta}^u|\Psi^u)$ are used exchangeably for convenience. Typically, for $x_t = [x_{1,t}, \cdots, x_{d,t}]^{\mathrm{T}}$ we also have

$$q(x_t|\eta_t^x, \psi_t^x) = \textstyle\prod_j q(x_{j,t}|\eta_{j,t}^x, \psi_{j,t}^x). \tag{11}$$

From knowing that elements of the additive noise E have zero mean and are uncorrelated among all its elements and also with $\boldsymbol{\eta}^x$, we further have

$$\begin{aligned} \mathcal{E}[\mathrm{vec}(X)\mathrm{vec}^{\mathrm{T}}(X)] = \mathcal{E}[\mathrm{vec}(\boldsymbol{\eta}^x)\mathrm{vec}^{\mathrm{T}}(\boldsymbol{\eta}^x)] \\ + \mathcal{E}[\mathrm{vec}(E)\mathrm{vec}^{\mathrm{T}}(E)]. \end{aligned} \tag{12}$$

This is a problem of additive decomposition of a nonnegative definite matrix into a sum of two nonnegative definite matrices. Without knowing the noise covariance $\mathcal{E}[\mathrm{vec}(E)\mathrm{vec}^{\mathrm{T}}(E)]$, we have an additive indeterminacy that the decomposition is ill-posed since there are infinite number of possibilities. To reduce the indeterminacy, we may further impose some structure on a diagonal matrix $\mathcal{E}[\mathrm{vec}(E)\mathrm{vec}^{\mathrm{T}}(E)]$. E.g., we have

$$\begin{aligned} \mathcal{E}[vec(E)vec^{\mathrm{T}}(E)] = diag[\Psi^x, \cdots, \Psi^x], \\ or\ \Psi_t^x = \Psi^x, \end{aligned} \tag{13}$$

i.e., each row of E has a same covariance. At one extreme case, we even assume that all the elements of E shares a same covariance σ^2 as follows

$$\Psi^x = \sigma^2 I,\ or\ \psi_{j,t}^x = \sigma^2. \tag{14}$$

Even in this case, the additive indeterminacy is still not totally eliminated as long as $\mathcal{E}[\mathrm{vec}(\boldsymbol{\eta}^x)\mathrm{vec}^{\mathrm{T}}(\boldsymbol{\eta}^x)] + \gamma I$ and $\sigma^2 - \gamma$ both remain nonnegative for any scalar γ.

The other way to reduce this indeterminacy is taking the structure of $\boldsymbol{\eta}^x$ in consideration. For $\boldsymbol{\eta}^x = AY$ in Eq. (4), the above indeterminacy about a scalar γ may be eliminated by the maximum likelihood learning when the rank of AY is less than the full rank d. However, the indeterminacy still remains when either AY is full rank or Ψ^x is diagonal. Moreover, it follows that

$$\begin{aligned} AY = A\phi\phi^{-1}Y = A^*Y^*, \\ A^* = A\phi, Y^* = \phi^{-1}Y. \end{aligned} \tag{15}$$

i.e., AY suffers an indeterminacy of any nonsingular matrix ϕ.

To tackle the problem, we consider an appropriate structural constraint on Y. A typical structure is that its elements are independently distributed, that is, $q(Y - \boldsymbol{\eta}^y|\Psi^y)$ in Eq. (7) is given as follows:

$$\begin{aligned} q(Y|\boldsymbol{\eta}^y, \Psi^y) &= q(Y - \boldsymbol{\eta}^y|\Psi^y), \\ q(Y|\boldsymbol{\eta}^y, \Psi^y) &= \textstyle\prod_t q(y_t|\eta_t^y, \Psi_t^y), \\ q(y_t|\eta_t^y, \psi_t^y) &= \textstyle\prod_j q(y_{j,t}|\eta_{j,t}^y, \psi_{j,t}^y), \\ \eta_t^y &= \mathcal{E}[y_t], \\ \Psi_t^y &= \mathcal{E}[(y_t - \eta_t^x)(x_t - \eta_t^x)^{\mathrm{T}}] \\ &= \mathrm{diag}[\psi_1^y, \psi_2^y, \cdots, \psi_m^y], \\ Y &= [y_1, \cdots, y_N]^{\mathrm{T}},\ y_t = [y_{1,t}, \cdots, y_{m,t}]^{\mathrm{T}}. \end{aligned} \tag{16}$$

Similar to Eq. (13), in many problems [2–18], the columns of Y are independently and identically distributed (i.i.d.), from which we have

$$\Psi_t^y = \Psi^y. \tag{17}$$

Moreover, the counterpart of Eq. (14) is also encountered in some studies.

For $\boldsymbol{\eta}^x = AY$ in Eq. (4), $\boldsymbol{\eta}^x$ is regarded as generated from independent hidden factors, which makes the indeterminacy of any nonsingular matrix ϕ in Eq. (15) reduces to an indeterminacy that ϕ comes from the orthogonal matrix family. In this case, Eq. (10) covers several typical latent variable models shown in Fig. 1, with the following details :

- As illustrated by the Box ①, we are lead to the classic FA [2–4] for real valued Y featured with that each $y_{i,t}$ is the following Gaussian

$$\begin{aligned} q(y_{j,t}|\eta^y_{j,t},\psi^y_j) &= G(y_{j,t}|0,1),\\ \eta^y_{j,t}=0, \psi^y_j &= 1,\\ q(y_t|\eta^y_t,\psi^y_t) &= G(y_t|\mathbf{0},I), \end{aligned} \tag{18}$$

  where and hereafter $G(\boldsymbol{u}|\boldsymbol{\mu},\Sigma)$ denotes a Gaussian distribution with a mean μ and a covariance matrix Σ. The indeterminacy of any orthogonal matrix ϕ reduces to an orthonormal matrix since $\psi^y_j = 1$.
- As illustrated by the Box ③, we are lead to the binary FA (BFA) [5–8] with each $y_{i,t} = 0$ or $y_{i,t} = 1$ from

$$\begin{aligned} q(y_{j,t}\,|\eta^y_{j,t},\psi^y_j) &= q(y_{j,t}|\eta^y_{j,t})\\ = exp\{y_{j,t}\ln\eta^y_j &+(1-y_{j,t})\ln(1-\eta^y_j)\},\\ \eta^y_{j,t}=\eta^y_j,\quad \psi^y_j &= \eta^y_j(1-\eta^y_j), \end{aligned} \tag{19}$$

  where ψ^y_j is not a free parameter but a function of $\eta^y_{j,t}$ that need not to be put in the distribution. The indeterminacy by ϕ reduces to only any permutation, since $y_{i,t}$ takes only 1 or 0.
- The above BFA includes a special case that has one additional constraint

$$\begin{aligned} y_{i,t}=0,\ y_{i,t}=1,\ \textstyle\sum_i y_{i,t}=1,\\ q(y_t|\eta^y_t) = exp\{\textstyle\sum_j y_{j,t}\ln\eta^y_j\},\ \textstyle\sum_j \eta^y_j = 1. \end{aligned} \tag{20}$$

  That is, the hidden factors are not only binary but also exclusively taking 1 by only one factor. Also, to be further introduced in Sect. 3.1 that this exclusive BFA equivalently implements the classic least square error (MSE) clustering problem, as illustrated by the Box ⑤.
- As illustrated by the Box ②, we are lead to the non-Gaussian FA (NFA) [9–13] for real valued Y that at most one $y_{i,t}$ per column is Gaussian. In this case, we have a scale indeterminacy [86].
- As illustrated in the Boxes ④–⑥, the NFA includes a family featured by that both A and Y are nonnegative matrices, where a matrix is nonnegative if every element is nonnegative valued. Extensive studies has been widely made on this family under the term nonnegative matrix factorization (NMF) [19–28]. Moreover, BFA and exclusive BFA also lead to its NMF counterparts when A is a nonnegative matrix, as illustrated by the Box ⑦.

Moreover, both BFA and NFA closely relate to multiple cause mixture model [14,15], and generalized latent trait models or item response theory [16–18]. Particularly, Eq. (4) with Eq. (19) and Eq. (20) lead to two typical binary matrix factorization (BMF) models [28] when the matrix A comes from a distribution similar to Eq. (19) or a distribution similar to Eq. (20).

For a unified consideration on binary, real, and nonnegative valued $y_{i,t}$, we consider $q(y_{i,t}|\eta^y_{i,t},\psi^y_i)$ in Eq. (16) given by the following exponential family [87]:

$$q(u|\eta,\psi) = \begin{cases} \exp\{\frac{1}{\psi}[\eta u - a(\eta) - h(u)]\}, & \text{(a)}\\ G(u|\eta,\psi), & \text{(b)}. \end{cases} \tag{21}$$

Generally, η,ψ are called natural parameter and dispersion parameter, respectively. Corresponding to a specific distribution, the function $\eta(\cdot)$ is also a specific scalar function called the mean function while its inverse function $\eta^{-1}(r)$ is called the link function in the literature of generalized linear model (GLM) [88]. Some examples are shown in Table 1, e.g., we may consider Bernoulli and exponential distribution when $\psi = 1$ and u takes binary and nonnegative values, respectively.

Table 1 Link functions for several typical distributions in the exponential family

distribution	name	link function	mean function
Gaussian	identity	$\eta^{-1}(r) = r$	$\eta(\xi) = \xi$
exponential gamma	inverse	$\eta^{-1}(r) = r^{-1}$	$\eta(\xi) = \xi^{-1}$
binomial Bernoulli	logit	$\eta^{-1}(r) = \ln\frac{r}{1-r}$	$\eta(\xi) = \frac{1}{1+\exp(-\xi)}$

Similarly, we may consider $q(x_{i,t}|\eta^x_{i,t},\psi^x_i)$ coming from the exponential family by Eq. (10) together with Table 1, in order to cover that $x_{i,t}$ takes either of binary, real, and nonnegative types of values. Accordingly, we extend Eq. (4) into the following one:

$$\begin{aligned} \boldsymbol{\eta}^x = \begin{cases} AY & (a)\text{homogenous linear},\\ \eta(AY), & (b)\text{post-linear} \end{cases}\\ where\ \eta(V) = [\eta(v_{i,j})]\ \text{for a matrix}\ V = [v_{i,j}]\\ \text{and a monotonic scalar function}\ \eta(r), \end{aligned} \tag{22}$$

by which $X = \boldsymbol{\eta}^x + E$ becomes a post bi-linear system since it is an extension of the bi-linear system by Eq. (4) with the bilinear unit AY followed by an element-wise nonlinear scalar mapping $\eta(r)$. When both X and Y are given, the above model degenerates to the generalized linear model (GLM) for the linear regression [88].

A nonlinear scalar mapping $\eta(r)$ in Eq. (22)(b) also bring one additional favorable point. For Eq. (22)(a), the additive form by Eq. (12) gets the detailed form Eq. (9) for a fixed A. Observing $AS_YA^{\mathrm{T}} + \Sigma = AS_YA^{\mathrm{T}} + C + \Sigma - C$, there will be many values for C such that both $AS_YA^{\mathrm{T}}+C$ and $\Sigma-C$ remain nonnegative definite. Also, any nonnegative definite matrix can be rewritten in the form $A^*S^*_yA^{*\mathrm{T}}$. In other words, there is still an additive indeterminacy. For Eq. (22)(b), AS_YA^{T} becomes $E[\eta(AY)\eta^{\mathrm{T}}(AY)]$, while $E[\eta(AY)\eta^{\mathrm{T}}(AY)] + C$ usually may not be rewritten into the same format of $E[\eta(AY)\eta^{\mathrm{T}}(AY)]$. In other words, an additive indeterminacy has been eliminated.

A post bi-linear system is described by Eq. (10) and Eq. (11) in help with Eq. (16) plus a specific $q(y_{j,t}|\eta^y_{j,t},\psi^y_j)$, (e.g., either of Eq. (18), Eq. (19), and Eq. (20)) to meet a specific learning task. The task of estimating all the unknown parameters in the system is called parameter learning, which is typically implemented under the principle of the maximum likelihood (ML), that is

$$\begin{aligned}&\Theta^* = \text{argmax}_\Theta \ln q(X|\Theta),\ \Theta = \{A, \Psi^x, \Theta^y\},\\ &q(X|\Theta) = \int q(X|\eta(AY), \Psi^x) q(Y|\Theta^y)\text{d}Y.\end{aligned} \tag{23}$$

The maximization is usually implemented by the expectation maximization (EM) algorithm [3,5,10–12].

One major challenge for the ML learning is that the rank of AY needs to be given in advance, while giving an inappropriate rank will deteriorate the learning performances. This challenge is usually tackled by model selection, sparse learning, and controlling model complexity, which are three closely related concepts in the literature of machine learning and statistics. The concept of model selection came from several decades ago on the studies of linear regression for selecting the number of variables [89,90], of clustering analysis for the number of clusters [91], of times series modeling for the order of autoregressive model [92]. The studies of this stream all involve to select the best among a family of candidate models via enumerating the number or order, and thus usually referred by the term of model selection.

The concept of model complexity came from the efforts also started in the 1960's by Solomonoff [93] on what later called Kolmogorov complexity [94]. Being different from task dependent models, these efforts aimed at a general framework that is able to measure the complexity of any given model by the counts of a unit complexity by a universal gauge and then to build a mathematical relation between this model complexity and the model performance (i.e., generalization error). One difficulty is how to get a universal gauge. A popular example is the so-called VC dimension [95] based theory on structural risk minimization, while the other example is the so-called yardstick or universal model in the evolution of MDL studies [45,46]. The other challenge is that the resulted mathematical relation between measured complexity and model performance is conceptually and qualitatively important but difficult to be applied to a model selection task in practice. Efforts have also been made towards this purpose and usually lead to some rough estimates, among which useful ones typically coincide with the first stream.

Instead of measuring the complexity contributed from every unit complexity, sparse learning is a recent popular topic that came from efforts on Laplace prior based Bayesian learning, featured by pruning away each extra parameter for selecting variables in a regression or interpolation task [41–43]. Though sparse learning and controlling model complexity are both working for a task similar to model selection and often referred in certain confusion, three concepts actually have different levels of focuses.

For a post bi-linear system by Eq. (10) and Eq. (16), the concept of model selection is selecting the column dimension m of Y or the number of variables in y_t. It is also equivalent to selecting the row dimension of A, while model complexity considers measuring the complexity of the entire system by counting a total sum contributed from every unit complexity by a universal gauge. This model complexity is a function of m and thus can be used for model selection. However, as above mentioned, this model complexity not only is difficult to be computed accurately but also may contain an additional part that even blurs or weakens the sensitivity on selecting m. Without measuring the contributions from every unit complexity and also being different from model selection that prunes away extra individual columns of A, sparse learning focuses on pruning away individual parameters in A per element.

Instead of tackling the difficulty of counting every unit complexity by a universal gauge or considering whether each individual parameter should be pruned away, model selection works on an appropriate middle level on which a unit incremental is featured by a sub-model, e.g., one column of the matrix Y. This feature not only avoids wasting computing cost on useless details but also uses limited information collectively for estimating reliably an intrinsic scale that suits the learning tasks. For the system by Eq. (10) and Eq. (16), this intrinsic scale is the dimension m.

Most of the existing studies on both model selection and sparse learning rely on ceratin a priori $q(\Theta|\Xi)$. For a post bi-linear system by Eq. (10) and Eq. (16), the first important a priori is about A. Recalling that the columns of A form a coordinate system in the observation space, we let such a priori about A in a structure of column-wise independence as follows:

$$\begin{aligned}&q(A - \boldsymbol{\eta}^a|\Psi^a) = q(A|\boldsymbol{\eta}^a, \Psi^a) = \textstyle\prod_j q(a_j|\boldsymbol{\eta}^a_j, \Psi^a_j),\\ &A = [a_1, a_2, \cdots, a_m],\ \boldsymbol{\eta}^a = [\boldsymbol{\eta}^a_1, \boldsymbol{\eta}^a_2, \cdots, \boldsymbol{\eta}^a_m],\\ &\Psi^a = \text{diag}[\psi^a_1, \psi^a_2, \cdots, \psi^a_m],\end{aligned} \tag{24}$$

where $q(a_j|\boldsymbol{\eta}^a_j, \Psi^a)$ comes from the following extension of Eq. (21) for a multivariate vector u:

$$q(u|\eta,\psi) = \begin{cases} e^{\eta^{\text{T}}\psi^{-1}u - a(\eta,\psi) - h(u,\psi)}, & \text{(a)},\\ G(u|\eta,\psi), & \text{(b)},\\ ML(u|\eta,\psi), & \text{(c)},\end{cases}$$

where ψ is usually a diagonal matrix for the case (a), and ML$(u|\eta,\psi)$ denotes a multivariate Laplace extended from its counterpart multivariate Gaussian $G(u|\eta,\psi)$.

On one hand, extensive efforts have been made on learning based a priori with help of Bayesian approaches. That is, the ML learning by Eq. (23) is extended into maximizing Eq. (5) under the name of Bayesian learning or maximizing Eq. (6) under the name of marginal Bayes or its approximation under the name of variational Bayes. As outlined in Sect. 2 and especially Figs. 4&5 of Ref. [1], these studies all base on a priori $q(\Theta|\Xi)$, including the one by Eq. (24) to make model selection and sparse learning, while the role of $q(Y|\Theta^y)$ by Eq. (16) is hidden behind the integral in Eq. (23) without taking its role.

On the other hand, via $q(R)$ by Eq. (8) the BYY harmony learning by Eq. (2) considers $q(Y|\Theta^y)$ in a role that is not only equally important to $q(\Theta|\Xi)$ but also easy computing, while $q(\Theta|\Xi)$ is still handled in a way similar to Bayesian approaches. As addressed in Sect. 2.2 of Ref. [1], the BYY harmony learning on Eq. (4) leads to improved model selection via either or both of improved model selection criteria and Ying-Yang alternative learning with automatic model selection.

Conventionally, model selection is implemented in two stages. That is, enumerating a number of m and learning unknown parameters at each m, and then selecting a best m^* by a model selection criterion, such as AIC/BIC/MDL. An alternative road of efforts is referred as automatic model selection. An early effort made since 1992 is rival penalized competitive learning (RPCL) [96,97,34] for clustering analysis and Gaussian mixture, with the cluster number k automatically determined during learning. Also, sparse learning can be regarded as implementing a type of automatic model selection, e.g., it leads to model selection if the parameters of one entire column of A has been all pruned away. The above mentioned Bayesian approach may also be implemented in a way of automatic model selection, e.g., pruning extra clusters by a Dirichlet prior [40].

As outlined at the end of Sect. 2.1 in Ref. [1], automatic model selection is associated with a learning algorithm or principle with the following two features:

- there is an indicator $\psi(\theta_{\rm SR})$ on a subset $\theta_{\rm SR}$ of parameters that represents a particular structural component.
- during implementing this learning, there is an intrinsic mechanism that drives

$$\psi(\theta_{\rm SR}) \to 0, \text{ as } \theta_{\rm SR} \to \text{ a specific value}, \quad (25)$$

if the corresponding component is redundant.

Thus, automatic model selection gets in effect via checking $\psi(\theta_{\rm SR}) \to 0$ and then discarding its corresponding $\theta_{\rm SR}$. The simplest case is checking whether $\theta_{\rm SR} \to \mathbf{0}$, a typical scenario encountered in Ref. [1].

In the rest of this section, $q(A|\boldsymbol{\eta}^a, \Psi^a)$ is put into Eq. (7) and jointly considered with $q(Y|\Theta^y)$ such that the BYY harmony learning on Eq. (4) further improves model selection and sparse learning, with help of exploring the co-dimension nature of the matrix pair A, Y.

2.2 Co-dim matrix pair and BYY harmony learning

Two matrices A and Y are regarded as a co-dimensional (shortly co-dim) matrix pair if they share a same rank m. In the post bi-linear system by Eq. (10) and Eq. (22), a co-dim matrix pair A and Y forms a matrix product AY as a core building unit. Actually, the common rank m is an intrinsic dimension that is shared from two aspects:

$$m \text{ is shared by } \begin{cases} \text{all the columns of } Y, & (a) \\ \text{all the rows of } A. & (b) \end{cases} \quad (26)$$

That is, there are two sources of information that could be integrated for a reliable estimation on this intrinsic co-dimension m.

In the studies of linear regression that estimates A with both X and Y given, model selection is made on selecting the variables of each row in A with the help of either a criterion (e.g., Cp , AIC, BIC) [44,89,91] that takes the number of variables in consideration with help of a priori on A. In the studies of learning a post bi-linear system with X given and Y unknown, parameter estimation is made by the maximum likelihood on $q(X|\Theta)$ in Eq. (23), and model selection is made via Bayesian approach by Eq. (5) or Eq. (6) with help of a priori on A. In these studies, model selection uses only the information from A, while the information from Y has not been used for model selection though it is used for estimating A. In other words, the information of Y has been ignored or even not been noticed in those previous studies. In contrast, as addressed in Sect. 2.2 of Ref. [1], the BYY harmony learning considers the information of Y via $q(Y|\Theta^y)$ in a role that differs from that in Eq. (23) but is equally important to a priori on A, which leads to improved model selection on m.

Taking the studies on the classic FA [2–4] for further insights, it follow from Eq. (18) that $G(y_{j,t}|0,1)$ has been widely adopted in the literature of statistics for describing each hidden factor, without unknowns to be estimated. Equivalently, there is no free parameter within this parametrization for describing Y. In Ref. [76, Item 9.4], we considered a different FA parametrization by restricting the matrix A to be orthonormal matrix and relaxing the extreme case $G(y_{j,t}|0,1)$ to $G(y_{j,t}|0,\psi_j^y)$ with one unknown ψ_j^y. The two FA parameterizations make no difference on $q(X|\Theta)$ in Eq. (23) and thus are equivalent in term of the ML learning. In contrast, two FA parameterizations become different in term of the BYY harmony learning, as listed in Table 2 of Ref. [9]. Also, it was experimentally found that the FA with $G(y_{j,t}|0,\psi_j^y)$ outperforms considerably the FA with $G(y_{j,t}|0,1)$ [98],

which may be understood from observing that a unknown parameter ψ_j^y provides a room for a further improvement by the BYY harmony learning.

Though the BYY harmony learning takes the information of Y in consideration of model selection via $q(Y|\Theta^y)$ by Eq. (16), the previous studies on the BYY harmony learning uses the information from A in a way similar to Bayesian approach by Eq. (5) or Eq. (6), in lack of a good coordination with $q(Y|\Theta^y)$. For improvements, we need further examine the effects of scale indeterminacy.

As discussed in the previous subsection, this product AY suffers from the indeterminacy by Eq. (15), which is remedied via requiring the row independence of $q(Y|\Theta^y)$ by Eq. (16). For a diagonal matrix $\phi = D \neq \gamma I$, the indeterminacy by Eq. (15) is usually called the scalar indeterminacy, which can not be removed by Eq. (16) for a real valued Y. When each $y_{i,t}$ takes either 1 or 0, such a scale indeterminacy is not permitted, since $Y^* = D^{-1}Y$ could not remain to be either 1 or 0. Alternatively, if $y_{i,t}$ is allowed to be binary of any two values, we still have a scale indeterminacy.

The studies of maximum likelihood and Bayesian approach by Eq. (5) or Eq. (6) rely on $q(X|\Theta)$ in Eq. (23), which is insensitive to such a scale indeterminacy. Usually $\psi_j^y = 1$ is imposed to remove this scale indeterminacy, e.g., in Eq. (18) for the classic FA [2–4]. With $G(y_{j,t}|0,1)$ relaxed to $G(y_{j,t}|0,\psi_j^y)$, the BYY harmony learning searches an appropriate value for each ψ_j^y, with an improved model selection. Still, there is a scale indeterminacy that is removed by imposing the constraint $A^{\mathrm{T}}A = I$ [8,9,13].

In sequel, we seek a coordinated consideration of both $q(Y|\boldsymbol{\eta}^y, \Psi^y)$ by Eq. (16) and $q(A|\boldsymbol{\eta}^a, \Psi^a)$ by Eq. (24),

$$q(A|\mathbf{0}, \Psi^a) = q(A|\boldsymbol{\eta}^a, \Psi^a)|_{\boldsymbol{\eta}^a=0},$$
$$q(Y|\mathbf{0}, \Psi^y) = q(Y|\boldsymbol{\eta}^y, \Psi^y)|_{\boldsymbol{\eta}^y=0}.$$

We again focus on the FA with $G(y_{j,t}|0,1)$ and the FA with $G(y_{j,t}|0,\psi_j^y)$. Both the two FA parameterizations are special cases of a family of FA variants featured with a transform

$$A^* = AD^{-1},\ Y^* = DY,\ D \text{ is a diagonal}, \tag{27}$$

which is shortly called the FA-D family. As stated above, instances in the FA-D family are equivalent in term of the ML learning and Bayesian approach. In Ref. [1], the BYY harmony learning by Eq. (2) with $q(R)$ by Eq. (8) is considered with $q(A|\mathbf{0}, \Psi^a)$ buried in $q(\Theta)$, while $q(Y|\mathbf{0}, \Psi^y) = \prod_t G(y_t|0, D^2)$ with different diagonal matrices of D makes no difference on $q(X|\Theta)$ by Eq. (23) but indeed leads to a difference for the BYY harmony learning.

With help of $q(R)$ by Eq. (7) that jointly considers $q(Y|\mathbf{0}, \Psi^y)$ and $q(A|\mathbf{0}, \Psi^a)$, it follows from $q(Y^*|\mathbf{0}, \Psi^y) = q(Y|\mathbf{0}, \Psi^y)/|D|$ and $q(A^*|0, \Psi^a) = |D|q(A|0, \Psi^a)$ that the value of this $q(R)$ and thus the harmony measure $H(p||q)$ by Eq. (2) are invariant to Eq. (27). That is, all the variants in the FA-D family become equivalent to each other. Interestingly, the above two FA parameterizations become equivalent again. This coordinated nature of a paired $q(Y|\mathbf{0}, \Psi^y)$ and $q(A|0, \Psi^a)$ provides the following new insights.

(1) This variant nature is different from the previous one owned by the ML learning. Due to $q(X|\Theta)$ in Eq. (23), any value for D has no difference and also no help for model selection. Thus, $\psi_j^y = 1$ is simply imposed to remove such a scale indeterminacy, while the above BYY harmony learning is able to use the information of Y in consideration of model selection via $q(Y|\Theta^y)$. With $q(R)$ by Eq. (8), two FA parameterizations make the harmony measure $H(p||q)$ by Eq. (2) different. However, we still do not know which one is better though the FA with $G(y_{j,t}|0,\psi_j^y)$ is experimentally shown to outperform the FA with $G(y_{j,t}|0,1)$ [98]. In contrast, with $q(R)$ by Eq. (7), knowing that two FA parameterizations make $H(p||q)$ by Eq. (2) take the same value indicates that we need to compare two FA parameterizations from aspects other than from $H(p||q)$.

First, the FA with $G(y_{j,t}|0,\psi_j^y)$ is better than the FA with $G(y_{j,t}|0,1)$ in term of being able to use the information of Y for model selection. Particularly, automatic model selection can be made via discarding the j-th dimension as the BYY harmony learning drives

$$\psi_j^y \to 0, \tag{28}$$

as a simple example of Eq. (25). Second, in comparison with a priori $q(A|0, \Psi^a)$, $q(Y|\mathbf{0}, \Psi^y)$ is more reliable and easy to use (see Sect. 2.2 of Ref. [1]), while an inappropriate $q(A|\mathbf{0}, \Psi^a)$ will deteriorate the overall performance. Third, it follows from Eq. (26) that Y has N columns to contain the information about m while A has only d rows, where we usually have $N \geqslant d$ or even $N \gg d$.

(2) In those previous studies on the BYY harmony learning [1], the role of a priori $q(A|\mathbf{0}, \Psi^a)$ is buried in $q(\Theta|\Xi)$ that contributes to a model selection criterion roughly via the number of free parameters in Θ as a whole. A paired consideration of $q(Y|\mathbf{0}, \Psi^y)$ and $q(A|\mathbf{0}, \Psi^a)$ in Eq. (7) also motivates to put the contribution by $q(A|\mathbf{0}, \Psi^a)$ in a more detailed expression. E.g., the model selection criterion by Eq. (18) in Ref. [1] is modified into

$$\begin{aligned} 2J(m) = {} & \ln|\Psi^x| + h^2\mathrm{Tr}[\Psi^{x\ -1}] + m\ln(2\pi\mathrm{e}) + \\ & \ln|\Psi^y| + \tfrac{d}{N}[m\ln(2\pi\mathrm{e}) + \ln|\Psi^a|] + n_f(\Theta), \end{aligned} \tag{29}$$

where $\ln|\Psi^a| = \sum_{i,j} \ln \Psi^a_{i,j}$, and the notations Ψ^x, Ψ^y were changed from ones Σ and Λ in Sect. 3.2 of Ref. [1], adopting the notation system of this paper (see Fig. 2). Also, we may ignore $n_f(\Theta)$ if there is no appropriate priori. It is observed that the contribution from $q(A|\mathbf{0}, \Psi^a)$ is weighted by a ratio d/N, which echoes the discussion

made at the end of the above item (1). Similarly, we may also modify the model selection criterions by Eqs. (13) and (19) in Ref. [1].

(3) In the previous studies on the FA with $G(y_{j,t}|0,\psi_j^y)$ (e.g., see Sect. 3.2 in Ref. [1]), to normally make automatic model selection with help of checking Eq. (28), the BYY harmony learning need to be implemented under the constraint of requiring $A^{\mathrm{T}}A = I$ for removing a scale indeterminacy. It follows that Eq. (27) leads to the following scalar indeterminacy:

$$\psi_j^{y*} = d_j^2\psi_j^y, \quad \Psi_j^{a*} = d_j^{-2}\Psi_j^a. \tag{30}$$

That is, it may occur that one element of Ψ^{y*} tends to zero due to a unknown scaling $d_j^2 \to 0$ that may simultaneously make the counterparting Ψ^{a^*} tend to infinity. The constraint $A^{\mathrm{T}}A = I$ can avoid this scenario. Moreover, a paired consideration of $q(Y|\mathbf{0},\Psi^y)$ and $q(A|\mathbf{0},\Psi^a)$ in Eq. (7) motivate to find a better choice.

It can be observed that the above scalar indeterminacy can also be avoided by the following constraint

$$\mathrm{Tr}[\Psi_j^a] = const \text{ or simply } \mathrm{Tr}[\Psi_j^a] = 1, \tag{31}$$

which is a relaxation of $A^{\mathrm{T}}A = I$ at a diagonal case

$$\begin{aligned} \Psi_j^a &= diag[\psi_{1,j}^a, \cdots, \psi_{d,j}^a], \\ \psi_{i,j}^a &= \mathcal{E}[(a_{ij} - \eta_{ij}^a)^2]. \end{aligned} \tag{32}$$

Noticing that $A^{\mathrm{T}}A = I$ includes $a_j^{\mathrm{T}}a_j = 1$ and that $\psi_{i,j}^a$ is a variance of the random variable a_{ij}, we see that $\sum_j a_{ij}^2 = 1$ actually leads to $\sum_j \Psi_{i,j}^a = 1$ when $\mathcal{E}[a_{ij}] = 0$. Inversely, $\sum_j \Psi_{i,j}^a = 1$ does not necessarily lead to $\sum_j a_{ij}^2 = 1$.

The alternatives of $A^{\mathrm{T}}A = I$ also include

$$\{\sum_i |a_{ij}|^\gamma\}^{1/\gamma} = 1,\ 0 < \gamma < \infty, \text{for every } j, \tag{33}$$

where the case $\gamma = 2$ includes $\sum_j a_{ij}^2 = 1$ as a special case. Moreover, it can be observed that these cases are all within the FA-D family by Eq. (27) with

$$\begin{aligned} &D = diag[d_1, \cdots, d_m], \\ &d_j = 1/\mathrm{Tr}[\Psi_j^a], \text{ for Eq. (30)}, \\ &d_j = 1/\{\textstyle\sum_i |a_{ij}|^\gamma\}^{1/\gamma} \text{ for Eq. (33)}\ . \end{aligned} \tag{34}$$

(4) For the Bayesian approach by Eq. (5) or Eq. (6) with $q(X|\Theta)$ by Eq. (23), the constraint $\psi_j^y = 1$ already shut down the contribution of $q(Y|\mathbf{0},\Psi^y)$ to model selection. Actually, model selection is implemented via discarding the j-th column of A if the entire column $a_j \to 0$. Also, sparse learning is implemented via discarding one element a_{ij} if $a_{ij} \to 0$. In contrast, the BYY harmony learning improves model selection via either or both of Eq. (28) and Eq. (29), with help of one constraint by either of Eq. (31), Eq. (33) and Eq. (34). Still, sparse learning can be performed since such a constraint will not impede $a_{ij} \to 0$.

Moreover, checking $a_{ij} \to 0$ may also be improved by checking whether

$$\Psi_{i,j}^a \to 0. \tag{35}$$

A coordinated implementation of Eq. (28) and Eq. (35) under the constraint of Eq. (31) (or one of its alternatives) form a good mechanism that coordinates automatic model selection and sparse learning.

(5) Adding extra constraint to remove a scale indeterminacy has both a good side and a bad side. The good side is facilitating to make model selection and sparse learning by Eq. (28) and Eq. (35) and also reducing the targeted domain of solutions. The bad side is that externally forcing the targeted domain not only increasing computing cost but also make it easy to be stuck at some suboptimal solution. Thus, we consider how to make model selection and sparse learning without imposing those constraints by Eq. (31), Eq. (33) and Eq. (34). Instead of checking by Eq. (28) and Eq. (35), i.e., the simplest format $\theta_{\mathrm{SR}} \to \mathbf{0}$ in Eq. (25), we consider $\psi(\theta_{\mathrm{SR}}) \to 0$ with $\psi(\theta_{\mathrm{SR}})$ in a composite format. It follows from Eq. (30) that the scalar indeterminacy also disappears by considering the product $\psi_j^{y*}\Psi_j^{a*}$. Thus, we replace Eq. (28) with the help of

$$\begin{aligned} &(a)\ \text{Discard the } j\text{-th row of } Y \text{ and also the } j\text{-th} \\ &\quad \text{column of } A \text{ if } \psi(\theta_{\mathrm{SR}}) = \psi_j^y \mathrm{Tr}[\Psi_j^a] \to 0, \\ &(b)\ \text{Discard the } a_{ij} \text{ of } A \text{ if } \psi(\theta_{\mathrm{SR}}) = \psi_j^y\psi_{i,j}^a \to 0. \\ &\quad \text{for the special case by Eq. (32).} \end{aligned} \tag{36}$$

When the j-th dimension of y_t or equivalently the j-th row of Y is redundant, both Eqs. (36)(a) and (36)(b) will happen for every i. In this case, they equivalently prune away the j-th dimension. If Eq. (36)(b) happens only for some i, the corresponding elements in the j-th column of A will be pruned away, that is, we are lead to sparse learning. In other words, both automatic model selection and sparse learning are nicely coordinated. Also, it can be observed that Eq. (36)(a) returns back to Eq. (28) under the constraint by Eq. (31). In other words, Eq. (36)(a) is an integration of Eq. (28) and Eq. (31).

2.3 Apex approximation and alternation maximization

We further consider the Bayesian Ying Yang system shown in Fig. 2, with the following representation

$$\begin{aligned} q(R) &= q(Y|\boldsymbol{\eta}^y,\Psi^y)q(A|\boldsymbol{\eta}^a,\Psi^a)q(\Psi|\Xi), \\ \Psi &= \{\Psi^a,\Psi^y,\Psi^x\}, \end{aligned} \tag{37}$$

which is a special case of Eq.(7) with Υ consisting of only Ψ while the rest parameters (if any) ignored.

Together with $q(X|R)$ by Eq. (10), we get the Ying machine $q(X,R)=q(X|R)q(R)$. For the Yang machine, $p(X)$ usually comes from a simple estimate $p(X)=G(X|X_N,h^2I)$ based on a set X_N of N samples. Here, we set $h=0$ for simplicity. The structure of $p(R|X)$ is designed as a functional with $q(X|R)$, $q(R)$ as its arguments according to a Ying-Yang variety preservation principle (see Sect. 4.2 in Ref. [1]). Putting the BYY system into Eq. (2), we have

$$\begin{aligned} &H(p||q,m,\Xi)\\ &\quad=\int p(\Psi|X_N)\ln[Q_{X_N|\Psi}\ q(\Psi|\Xi)]\mathrm{d}\Psi,\\ &\ln Q_{X_N|\Psi}=H(p||q,X_N,\Psi)=\\ &\quad\int p(A|\Psi\ ,X_N)\ln[Q_{X_N,A|\Psi}\ q(A|\boldsymbol{\eta}^a,\Psi^a)]\mathrm{d}A,\\ &\ln Q_{X_N,A|\Psi}=H(p||q,A,X_N,\Psi)=\\ &\quad\times\ln[q(X_N|\boldsymbol{\eta}(AY),\Psi^x)q(Y|\boldsymbol{\eta}^y,\Psi^y)]\mathrm{d}Y, \end{aligned} \tag{38}$$

where and hereafter in this paper, we use the notation $Q=e^H$ and $\ln Q=H$ exchangeably for convenience.

For simplicity, we consider a data $X_N=\{x_t\}$ with zero mean. Otherwise, we need to remove it by

$$X_N=\{x_t^*\},\ x_t^*=x_t-\mu_x,\ \mu_x=\tfrac{1}{N}\textstyle\sum_t x_t.$$

Correspondingly, we let

$$\boldsymbol{\eta}^y=\mathbf{0},\boldsymbol{\eta}^a=\mathbf{0}. \tag{39}$$

Otherwise, there are two different scenarios. One is that $\boldsymbol{\eta}^y$, $\boldsymbol{\eta}^a$ are free parameters and determined by

$$\{\boldsymbol{\eta}^{y*},\boldsymbol{\eta}^{a*}\}=\arg\max_{\boldsymbol{\eta}^y,\boldsymbol{\eta}^a}H(p||q,\boldsymbol{\eta}^y,\boldsymbol{\eta}^a,m,\Xi). \tag{40}$$

The other scenario is that $\boldsymbol{\eta}^y$ and $\boldsymbol{\eta}^a$ are functions of other variables and thus learned via those variables. In such cases, $H(p||q,m,\Xi)$, $\ln Q_{X_N|\Psi}=H(p||q,X_N,\Psi)$, and $\ln Q_{X_N,A|\Psi}=H(p||q,A,X_N,\Psi)$ are functions of $\boldsymbol{\eta}^y,\boldsymbol{\eta}^a$, which will be further discussed in Sects. 4.1 and 4.2. In this section, we consider the cases by Eqs. (39) and (40), and thus omit to write out $\boldsymbol{\eta}^y,\boldsymbol{\eta}^a$ explicitly.

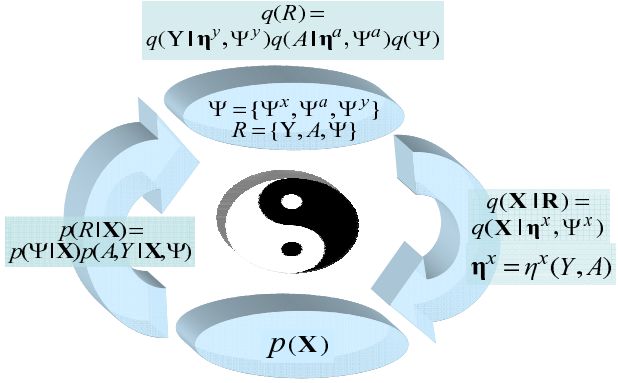

Fig. 2 Post bi-linear Bayesian Ying Yang system

It is usually difficult to handle those integrals in Eq. (38). We consider one integral approximately with help of a Taylor expansion up to the second order:

$$\begin{aligned} &\int p(u)Q(u)\mathrm{d}u\approx Q(u^*)-\tfrac{1}{2}[\varepsilon_u^{\mathrm{T}}\Omega_Q(u^*)\varepsilon_u+d_u],\\ &u^*=\mathrm{argmax}_uQ(u),\ \varepsilon_u=u^\mu-u^*,\\ &\Omega_Q(u)=-\tfrac{\partial^2Q(u)}{\partial u\partial u^{\mathrm{T}}},\ d_u=\mathrm{Tr}[\Gamma^u\Omega_Q(u^*)], \end{aligned} \tag{41}$$

which is called the apex approximation because it is made around the apex point u^*, where u^μ, Γ^u are the mean vector and covariance matrix of $p(u)$.

One key point is getting the apex point u^*, i.e., the difficulty of computing an integral is approximately by a task of maximization. Also, we typically consider

$$\Gamma^u=\Omega_Q^{-1}(u^*),\ \text{and thus}\ d_u=\mathrm{rank}[\Omega_Q(u^*)].$$

When $Q(u)$ is a quadratic function of u, not only this $\approx$ becomes $=$, but also Eq. (41) applies to the cases that u takes discrete values, with $\Omega_Q(u)$ obtained simply by regarding that the domain of u is expanded to a real domain.

The integrals over Ψ and A,Y in Eq. (38) are all in a format of $H(\omega)=\int p(\vartheta|\omega)\ln[Q(\vartheta)q(\vartheta|\omega)]\mathrm{d}\vartheta$, which may be partially integrable. Considering a partition $\vartheta=\zeta\cup\xi$, $\zeta\cap\xi=\emptyset$ such that $q(\vartheta|\omega)=q(\zeta|\omega_\zeta)q(\xi|\omega_\xi)$, we have

$$\begin{aligned} H(\omega)&=\int p(\vartheta|\omega)\ln[Q(\vartheta)q(\zeta|\omega_\zeta)q(\xi|\omega_\xi)]\mathrm{d}\zeta\mathrm{d}\xi\\ &=H_\zeta(\omega)+H_\xi(\omega),\\ H_\zeta(\omega)&=\int p(\zeta|\omega)\ln q(\zeta|\omega_\zeta)\mathrm{d}\zeta,\\ H_\xi(\omega)&=\int p(\vartheta|\omega)\ln[Q(\vartheta)q(\xi|\omega_\xi)]\mathrm{d}\vartheta,\\ &\approx\ln[Q(\vartheta^*)q(\xi|\omega_\xi)]-\tfrac{1}{2}[\varepsilon_u^{\mathrm{T}}\Omega_Q(\vartheta^*)\varepsilon_u+d_u],\\ \vartheta^*&=\mathrm{argmax}_\vartheta\ln[Q(\vartheta)q(\xi|\omega_\xi)],\\ \varepsilon_u&=\vartheta^\mu-\vartheta^*,\ d_u=\mathrm{Tr}[\Omega_Q(\vartheta^*)],\\ \Omega_Q(\vartheta)&=-\tfrac{\partial^2\ln[Q(\vartheta)q(\xi|\omega_\xi)]}{\partial\vartheta\partial\vartheta^{\mathrm{T}}}, \end{aligned} \tag{42}$$

where $H_\zeta(\omega)$ is integrable and thus is analytically handled, while $H_\xi(\omega)$ is approximated with help of Eq. (41). For a problem with an empty set $\zeta=\emptyset$, we have $q(\zeta|\omega_\zeta)=1$ and thus $H_\zeta(\omega)=0$.

With help of Eq. (42), the three integrals in Eq. (38) are handled with help of Eq. (41) as follows:

$$\begin{aligned} &(a)\ \text{Get}\ H(p||q,A,X_N,\Psi)\ \text{with}\ \textstyle\int\ldots\mathrm{d}Y\ \text{in two parts};\\ &(b)\ \text{Get}\ H(p||q,X_N,\Psi)\ \text{with}\ \textstyle\int\ldots\mathrm{d}A\ \text{in two parts};\\ &(c)\ \text{Get}\ H(p||q,m,\Xi)\ \text{with}\ \textstyle\int\ldots\mathrm{d}\Psi\ \text{in two parts};\\ &(d)\ \text{Discard the}\ j\text{-th row of}\ Y\ \text{via checking either}\\ &\quad\text{Eq. (28)(a) or Eq. (36)(a)}\\ &\quad\text{Prune}\ a_{i,j}\ \text{of}\ A\ \text{via checking either}\\ &\quad\text{Eq. (35) or Eq. (36)(b)}\\ &(e)\ \text{Update}\ \Xi^*=\arg\max_\Xi H(p||q,m,\Xi);\\ &(f)\ \text{Get}\ m^*=\arg\max_m H(p||q,m,\Xi^*). \end{aligned} \tag{43}$$

Each of the above steps is implemented with a part of unknowns that are estimated in other steps. Therefore, the procedure by Eq. (43) should be implemented iteratively, started from an initialization on all the unknown

parameters. The first four steps already composite one iterative learning algorithm with automatic model selection and sparse learning implemented at Step (d). Step (e) is involved only when there is a priori $q(\Psi|\Xi)$]. Also, Step (f) is made in a way similar to the conventional two stage implementation. That is, steps (a)-(e) are iterative until convergence at each m, after enumerating m for a number of value, a best m^* is selected by the criterion from $H(p||q, m, \Xi^*)$, e.g., the one in Eq. (29) for FA.

In Eqs. (43)(a)& (b) & (c), removing an integral in two parts as handled in Eq. (43). For some problems, the entire integral is analytically integrable, and thus there is no part that needs to make approximation. Also, there may be no analytically integrable part, for which the entire integral has to be tackled approximately. Usually, we need a trade-off between computing cost and accuracy when the entire integral is divided into two parts.

Taking $\int \ldots \mathrm{d}Y$ for an example, when each $y_{i,t}$ takes values by Eq. (20), the integral $\int \ldots \mathrm{d}Y$ becomes a simple summation, which definitely belongs to the part of analytically integrable. When each $y_{i,t}$ takes either 1 or 0, the integral $\int \ldots \mathrm{d}Y$ also becomes a summation, for which we encounter a trading-off scenario. If the dimension m is not high, this case may still be classified as the part of analytically integrable. However, the computational complexity of the summation becomes intractable for a big value m. Such a situation should be classified into the part to be handled approximately with help of Eq. (41).

In sequel, we introduce the detailed equations for the approximations in Eqs. (43)(a)& (b) & (c). Without losing generality and also for notation simplicity, we only consider how to handle the approximation part in Eq. (42), while the analytically integrable part is task dependent and handled manually.

We start from Step (a) to consider the integral $\int \ldots \mathrm{d}Y$ for getting $H(p||q, \boldsymbol{\eta}^y, \boldsymbol{\eta}^a, m, \Xi)$ in Eq. (38). It follows from Eq. (41) and Eq. (42) that we have

$$\begin{aligned}
\ln Q_{X_N,A|\Psi} &= H(p||q, A, X_N, \Psi) \\
&= \int p(Y|A, \Psi, X_N) \ln Q_{X_N,A,Y|\Psi} \mathrm{d}Y. \\
&= \ln Q_{X_N,A,Y^*|\Psi} \\
&\quad -\tfrac{1}{2}[\varepsilon_y^{\mathrm{T}} \Omega_{Y^*|A,\Psi} \varepsilon_y + d_{Y^*|A,\Psi}] \\
\ln Q_{X_N,A,Y^*|\Psi} &= \ln Q_{X_N,A,Y=Y^*|\Psi}, \\
Y^* &= \mathrm{argmax}_A \ln Q_{X_N,A,Y|\Psi}, \\
\ln Q_{X_N,A,Y|\Psi} &= \ln[q(X_N|\boldsymbol{\eta}(AY), \Psi^x) q(Y|\boldsymbol{\eta}^y, \Psi^y)], \\
\varepsilon_y &= \mathrm{vec}(Y_\mu - Y^*), \\
d_{Y^*|A,\Psi} &= \mathrm{rank}[\Omega_{Y^*|A,\Psi}], \\
\Omega_{Y|A,\Psi} &= -\tfrac{\partial^2 \ln Q_{X_N,A,Y|\Psi}}{\partial \mathrm{vec}(Y) \partial \mathrm{vec}(Y)^{\mathrm{T}}}.
\end{aligned} \tag{44}$$

We move to Step (b) to consider the integral $\int \ldots \mathrm{d}A$ for getting $H(p||q, X_N, \Psi)$ in Eq. (38).

$$\begin{aligned}
\ln Q_{X_N|\Psi} &= H(p||q, X_N, \Psi) \\
&= \ln[Q_{X_N,A^*|\Psi} q(A^*|\boldsymbol{\eta}^a, \Psi^a)] \\
&\quad -\tfrac{1}{2}[\varepsilon_a^{\mathrm{T}} \Omega_{A^*|Y^*,\Psi} \varepsilon_a + d_{A^*|Y^*,\Psi}], \\
A^* &= \mathrm{argmax}_A \ln[Q_{X_N,A|\Psi}\ q(A|\boldsymbol{\eta}^a, \Psi^a)], \\
\varepsilon_a &= \mathrm{vec}(A_\mu - A^*), \\
d_{A^*|Y^*,\Psi} &= \mathrm{rank}[\Omega_{A^*|Y^*,\Psi}], \\
\Omega_{A|Y,\Psi} &= -\tfrac{\partial^2 \ln[Q_{X_N,A|\Psi}\ q(A|\boldsymbol{\eta}^a, \Psi^a)]}{\partial \mathrm{vec}(A) \partial \mathrm{vec}(A)^{\mathrm{T}}}.
\end{aligned} \tag{45}$$

Next, we proceed to Step (c) to get the integral $\int \ldots \mathrm{d}\Psi$ for getting $H(p||q, m, \Xi)$ in Eq. (38) turned into

$$\begin{aligned}
H\ (p||q, m, \Xi) &= \int p(\Psi|X_N) \ln[Q_{X_N|\Psi}\ q(\Psi|\Xi)] \mathrm{d}\Psi \\
&= \ln[Q_{X_N|\Psi^*}\ q(\Psi^*|\Xi)] - \tfrac{1}{2}[\varepsilon_{\Psi^*}^{\mathrm{T}} \Omega_{\Psi^*} \varepsilon_{\Psi^*} + d_{\Psi^*}], \\
\Psi^* &= \mathrm{argmax}_\Psi \ln[Q_{X_N|\Psi}\ q(\Psi|\Xi)], \\
\varepsilon_\Psi &= \mathrm{vec}(\Psi_\mu - \Psi^*), \\
d_\Psi &= \mathrm{rank}[\Omega_\Psi], \\
\Omega_\Psi &= -\frac{\partial^2 \ln[Q_{X_N|\Psi}\ q(\Psi|\Xi)]}{\partial \mathrm{vec}(\Psi) \partial \mathrm{vec}(\Psi)^{\mathrm{T}}}.
\end{aligned} \tag{46}$$

In the above equations, the following computing issues need to be further addressed:

- Only for some special cases, the implementations of $\max_Y$, $\max_A$, and $\max_\Psi$ are analytically solvable. Generally, a maximization with respect to continuous variables is implemented by a gradient based searching algorithm, suffering a local maximization problem, while a maximization with respect to discrete variables, e.g., each $y_{i,t}$ takes either 1 or 0, involves a combinatorial optimization [99,100].
- The Hessian matrices $\Omega_{Y|A,\Psi}$, $\Omega_{A|Y,\Psi}$, and Ω_Ψ are typically assumed to be diagonal for avoiding tedious computation and unreliable estimation incurred from much parameters.
- For the learning tasks with the follow unknowns of the Yang machine

$$\begin{aligned}
\Psi_\mu &= \mathcal{E}_{p(\Psi|X_N)}[\Psi], \\
A_\mu &= \mathcal{E}_{p(A|\Psi,X_N)}[A], \\
Y_\mu &= \mathcal{E}_{p(Y|A,\Psi,X_N)}[Y],
\end{aligned}$$

being free to be determined via maximizing $H(p||q)$ by Eq. (2), we are led to $\Psi_\mu = \Psi^*, A_\mu = A^*, Y_\mu = Y^*$. When other parameters are still far away a convergence, enforcing $\varepsilon_\Psi = 0, \varepsilon_y = 0, \varepsilon_a = 0$ too early will make the entire learning process by Eq. (43) get stuck at local optimum. To balance the progresses of learning different parts in the entire maximization, one simple way is letting Ψ_μ, A_μ, Y_μ to be some previous Ψ^*, A^*, Y^* at a delayed time lag τ, that is,

$$\Psi_\mu = \Psi^*(\tau), A_\mu = A^*(\tau), Y_\mu = Y^*(\tau), \tag{47}$$

for which as the entire iteration converges we still get $\Psi_\mu = \Psi^*, A_\mu = A^*, Y_\mu = Y^*$.

Next, we provides further details by considering the following typical case:

$$\begin{aligned}
&q(X|\boldsymbol{\eta}^x, \Psi^x) \text{ by Eq. (10)}\\
&\quad q(Y|\Theta^y) \text{ by Eq. (16)}\\
&q(A|\boldsymbol{\eta}^a, \Psi^a) \text{ by Eq. (24) together with,}\\
&\boldsymbol{\eta}^x = AY,\ \boldsymbol{\eta}^a = [\boldsymbol{\eta}^a_1, \cdots, \boldsymbol{\eta}^a_m],\ \boldsymbol{\eta}^y = [\boldsymbol{\eta}^y_1, \cdots, \boldsymbol{\eta}^y_N],\\
&q(x_t|\eta^x_t, \Psi^x_t) = G(x_t|Ay_t, \Psi^x),\\
&q(y_t|\eta^y_t, \Psi^y_t) = G(y_t|\boldsymbol{\eta}^y, \Psi^y),\\
&q(a_j|\boldsymbol{\eta}^a_j, \Psi^a_j) = G(a_j|\boldsymbol{\eta}^a_j, \Psi^a_j).
\end{aligned} \tag{48}$$

We further get $\nabla_Y \ln Q_{X_N,A,Y|\Psi}$.

$$\begin{aligned}
&= A^{\mathrm{T}}_{\boldsymbol{\eta}} \Psi^{x\ -1} X - \Gamma_A Y_{\boldsymbol{\eta}},\\
Y^* &= \Gamma_A^{-1} A^{\mathrm{T}}_{\boldsymbol{\eta}} \Psi^{x\ -1} X + \boldsymbol{\eta}^y,\\
\Gamma_A &= A^{\mathrm{T}}_{\boldsymbol{\eta}} \Psi^{x\ -1} A_{\boldsymbol{\eta}} + \Psi^{y\ -1},\\
\Omega_{Y|A,\Psi} &= \Gamma_A \otimes I,\\
A_{\boldsymbol{\eta}} &= A - \boldsymbol{\eta}^a,\ Y_{\boldsymbol{\eta}} = Y - \boldsymbol{\eta}^y,
\end{aligned} \tag{49}$$

where $\otimes$ denotes the Hadamard product, and in this subsection we have $\boldsymbol{\eta}^y = \mathbf{0}$, $\boldsymbol{\eta}^a = \mathbf{0}$ and thus $A_{\boldsymbol{\eta}} = A$, $Y_{\boldsymbol{\eta}} = Y$. Still, we keep the notations $\boldsymbol{\eta}^y$, $\boldsymbol{\eta}^a$ here for the convenience of further discussions in Sect. 4.1.

Similarly, we also get

$$\begin{aligned}
&\nabla_A \ln[Q_{X_N,A|\Psi}\ q(A|\boldsymbol{\eta}^a, \Psi^a)]\\
&\quad = \Psi^{x\ -1} X Y^{\mathrm{T}}_{\boldsymbol{\eta}} - \Psi^{x\ -1} A_{\boldsymbol{\eta}} \Gamma_Y - A_{\boldsymbol{\eta}} \Psi^{a\ -1}\\
&\quad = \Psi^{x\ -1} [X Y^{\mathrm{T}}_{\boldsymbol{\eta}} - A_{\boldsymbol{\eta}} \Gamma_Y - \Psi^x A_{\boldsymbol{\eta}} \Psi^{a\ -1}]\\
\Gamma_Y &= Y_{\boldsymbol{\eta}} Y^{\mathrm{T}}_{\boldsymbol{\eta}} + (Y_\mu - Y^*)(Y_\mu - Y^*)^{\mathrm{T}},\\
\Omega_{A|Y,\Psi} &= \Psi^{x\ -1} \otimes \Gamma_Y + I \otimes \Psi^{a\ -1}.
\end{aligned} \tag{50}$$

When $\varepsilon_{\mathrm{y}} \neq 0$, it takes a regularization role via Γ_Y. Generally, the maximization with respect to A is reached at A^* that satisfies the following equation:

$$X Y^{\mathrm{T}}_{\boldsymbol{\eta}} - A^*_{\boldsymbol{\eta}} \Gamma_Y - \Psi^x A^*_{\boldsymbol{\eta}} \Psi^{a\ -1} = 0. \tag{51}$$

We can solve this A^* by a local searing algorithm based on the gradient $\nabla_A \ln[Q_{X_N,A|\Psi}\ q(A|\boldsymbol{\eta}^a, \Psi^a)]$. Moreover, when every element of Ψ^x is same, namely, $\Psi^x = \psi^x I$, we simply have

$$A^* = [\Gamma_Y + \psi^x \Psi^{a\ -1}]^{-1} X Y^{\mathrm{T}}_{\boldsymbol{\eta}} + \boldsymbol{\eta}^a. \tag{52}$$

Ignoring $q(\Psi|\Xi)$, from $\nabla_{\Psi^x,\Psi^y,\Psi^a_j} \ln[Q_{X_N|\Psi}\ q(\Psi|\Xi)] = \mathbf{0}$ we further get

$$\begin{aligned}
\Psi^{x*} &= \tfrac{1}{N} \textstyle\sum_t \mathrm{diag}[e^x_t e^{x\ \mathrm{T}}_t + \Delta\Psi^{x*}_t],\\
e^x_t &= x_t - A^* y^*_t,\ e^y_t = y_{t,\mu} - y^*_t,\ \delta A = A_\mu - A^*,\\
\Delta\Psi^{x*}_t &= A^* e^y_t e^{y\ \mathrm{T}}_t A^{*\ \mathrm{T}}\\
&\quad + \delta A (y^*_t - \boldsymbol{\eta}^y_t)(y^*_t - \boldsymbol{\eta}^y_t)^{*\ \mathrm{T}} \delta A^{\mathrm{T}},\\
\psi^{x*} &= \tfrac{1}{d} Tr[\Psi^{x*}],\ for\ \Psi^{x*} = \psi^{x*} I;\\
\Psi^{y*} &= \tfrac{1}{N} \textstyle\sum_t \mathrm{diag}[(y^*_t - \boldsymbol{\eta}^y_t)(y^*_t - \boldsymbol{\eta}^y_t)^{\mathrm{T}} + e^y_t e^{y\ \mathrm{T}}_t],\\
\Psi^{a*}_j &= \mathrm{diag}[(a^*_j - \boldsymbol{\eta}^a_j)(a^*_j - \boldsymbol{\eta}^a_j)^{\mathrm{T}}]\\
&\quad + \mathrm{diag}[(a^*_{j,\mu} - a^*_j)(a^*_{j,\mu} - a^*_j)^{\mathrm{T}}].
\end{aligned} \tag{53}$$

As discussed before Eq. (35), sparse learning prunes an element a_{ij} by checking $\Psi^a_{i,j} \to 0$. Also, there is an alternative choice on the order of considering Y, A, i.e., remove the integral over A first and then the integral over Y, with equations obtained by simply switching the position of Y and A.

3 Several typical learning tasks

3.1 De-noise Gaussian mixture

We start from one special case of $X = AY + E$ by Eq. (4) featured with $q(X|\boldsymbol{\eta}^x, \Psi^x)$ by Eq. (10) and Eq. (48) with $\Psi^x = \sigma^2 I$. We observe that a sample x_t is approximated by $Ay_t = \sum_j a_j y_{j,t}$ as a convex combination of the columns of A, weighted by the elements of each column y_t of Y. This approximation is made in term of minimizing the square error $\sum_t \|x_t - Ay_t\|^2$. With $q(Y|\eta^y, \Psi^y)$ given by Eq. (16) with $q(y_t|\eta^y_t, \psi^y_t) = q(y_t|\eta^y_t)$ by Eq. (20), it becomes equivalent to minimizing

$$\sum_t \|x_t - Ay_t\|^2 = \sum_t y_{j,t} \|x_t - a_j\|^2,$$

which has been widely studied under the name of the mean square error (MSE) clustering analysis. Echoing the discussions made after Eq. (20), we are led to the Box ⑤ in Fig. 1. Particularly, when samples of X are nonnegative and also A is nonnegative, we are further led to the Box ⑥ for those studies of making clustering analysis under the name of NMF [19–28].

Even interestingly, we further proceed to the Box ⑧ with Eq. (48). It follows from Eq. (7) that we have

$$\begin{aligned}
q(R) &= q(A|Y, \boldsymbol{\eta}^a, \Psi^a) q(Y|\boldsymbol{\eta}^y, \Psi^y)\\
&= \textstyle\prod_{t,j} [q(a_j|\boldsymbol{\eta}^a_j, \Psi^a) \eta^y_j]^{y_{j,t}},\\
q(A|Y, \boldsymbol{\eta}^a, \Psi^a) &= \textstyle\prod_{t,j} q(a_j|\boldsymbol{\eta}^a_j, \Psi^a)^{y_{j,t}}.
\end{aligned}$$

Being different from Eq. (24), here $q(A|Y, \boldsymbol{\eta}^a, \Psi^a)$ is considered under the condition of Y. It further follows that

$$\begin{aligned}
q(X_N|\theta) &= \textstyle\int q(X_N|\boldsymbol{\eta}^x, \Psi^x) q(A, Y|\theta) \mathrm{d}Y \mathrm{d}A\\
&= \textstyle\prod_t q(x_t|\theta)\\
q(x_t|\theta) &= \textstyle\sum_\ell \eta^y_\ell q(x_t|a_\ell, \Psi^x, \Psi^a_\ell),\\
q(x_t|a_\ell, \Psi^x, \Psi^a_\ell) &= \textstyle\int G(x_t|a_\ell, \Psi^x) q(a_\ell|\boldsymbol{\eta}^a_\ell, \Psi^a_\ell) \mathrm{d}a_\ell.
\end{aligned} \tag{54}$$

That is, X_N can be regarded as generated from a finite mixture distribution of $q(x_t|a_j, \Psi^x, \Psi^a_j)$.

Taking a multivariate Gaussian distribution as an example, we consider $q(a_j|\eta^a_j, \Psi^a_j) = G(a_j|\eta^a_j, \Psi^a_j)$ with the mean vector η^a_j and the covariance matrix Ψ^a_j that is not limited to be diagonal as in Eq. (24). We have

$$\begin{aligned}
q(R) = q(A, Y|\theta) &= \textstyle\prod_{t,j} [G(a_j|\eta^a_j, \Psi^a_j) \eta^y_j]^{y_{j,t}},\\
q(x_t|a_\ell, \Psi^x, \Psi^a_\ell) &= G(x_t|\eta^a_\ell, \Psi^x + \Psi^a_\ell)\\
&= \textstyle\int G(x_t|a_\ell, \Psi^x) G(a_\ell|\eta^a_\ell, \Psi^a_\ell) \mathrm{d}a_\ell.
\end{aligned} \tag{55}$$

That is, X_N can be regarded as generated from a Gaussian mixture with each Gaussian $G(x_t|\eta_j^a, \Psi^x + \Psi_j^a)$ in a proportion $\eta_j^y \geqslant 0$. Therefore, we are led to the Box ⑧ shown in Fig. 1.

When $\Psi^x = 0$ we return to a standard Gaussian mixture [29–31]. Here, the effect of adding a diagonal matrix Ψ^x to the covariance matrix Ψ_j^a is similar to that of data-smoothing learning for regularizing a small size of samples via a smoothing trick, i.e., each sample x_t is smoothed by a Gaussian kernel $G(x|x_t, h^2 I)$. Reader are referred to Eq. (7) of Ref [69], and a rather systematic elaboration in Ref. [36]. Here, Eq. (55) has two key differences from the previous data-smoothing learning. First, each sample is regularized by not just a scalar $h^2 I$ but a diagonal matrix Ψ^x that affects all the dimensions differently. Second, considering $G(x|x_t, h^2 I)$ externally is equivalent to adding a Gaussian white noise to samples, while Ψ^x interacts with $q(a_j|\eta_j^a, \Psi_j^a)$ in a way intrinsic to data and learning tasks.

It follows from Eq. (38) with the help of Eq. (55) for a generalized Gaussian mixture. It follows that

$$\begin{aligned}
&H\,(p||q, X_N, \theta)\\
&\quad = \textstyle\sum_t \int \sum_{y_{1,t},\cdots,y_{m,t}} \prod_j [p(j|x_t,\theta)p(a_j|x_t,\theta)]^{y_{j,t}}\\
&\quad \times \ln \textstyle\prod_j [G(x_t|a_j,\Psi^x)G(a_j|\eta_j^a,\Psi_j^a)\eta_j^y]^{y_{j,t}} \mathrm{d}A\\
&\quad = \textstyle\sum_t \sum_{j\in J_t} p(j|x_t,\theta) \int p(a_j|x_t,\theta) H_t(\theta_j, a_j) \mathrm{d}a_j\\
&\quad = \textstyle\sum_t \sum_{j\in J_t} p(j|x_t,\theta) H_t(\theta_j, a_{t,j}^*) - 0.5d, \qquad (56)\\
&\quad H_t(\theta_j, a_j)\\
&\quad = \ln\{G(x_t|a_j,\Psi^x)G(a_j|\eta_j^a,\Psi_j^a)\eta_j^y\},\\
&a_{t,j} = \mathrm{argmax}_{a_j} H_t(\theta_j, a_j)\\
&\quad = \left[\Psi^{x\ -1} + \Psi_j^{a\ -1}\right]^{-1} (\Psi^{x\ -1} x_t + \Psi_j^{a\ -1}\eta_j^a),
\end{aligned}$$

where the integral over a_j is made by Eq. (41), and J_t is a subset of indices as follows:

$$J_t = \begin{cases} \{1, 2, \cdots, m\}, & \text{(a) unsupervised,}\\ \text{teaching label } j_t^*, & \text{(b) supervised,}\\ \text{a winner subset,} & \text{(c) apex approximation.}\end{cases}$$

$$j_t^* = \begin{cases} j_t^* \text{ given in pair of } x_t, & \text{(a) supervised,}\\ \mathrm{argmax}_j H_t(\theta_j, a_{t,j}), & \text{(b) unsupervised,}\end{cases} \qquad (57)$$

where a winning subset consists of j_t^* and κ neighbors that corresponds the first κ largest values of $H_t(\theta_j, a_{t,j})$. The above J_t covers supervised learning for a teaching pair $\{x_t, j_t*\}$, unsupervised learning with no teaching label for each x_t, as well as semi-supervised learning if there are teaching labels for a subset of samples.

According to the variety preservation principle (see, Eq. (38) in Ref.[1]), $p(j|\theta, x_t)$ is designed from η_j^y, $G(x_t|a_j, \Psi^x)$, and $G(a_j|\eta_j^a, \Psi_j^a)$ as follows:

$$\begin{aligned}
p(j|\theta, x_t) &= \frac{\int exp\{H_t(\theta_j, a_j)\}\mathrm{d}a_j}{\sum_j \int exp\{H_t(\theta_j, a_j)\}\mathrm{d}a_j}\\
&= \frac{exp\{H_t(\theta_j, a_{t,j}) - 0.5\ln|\Psi^{x\ -1} + \Psi_j^{a\ -1}|\}}{\sum_j exp\{H_t(\theta_j, a_{t,j}) - 0.5\ln|\Psi^{x\ -1} + \Psi_j^{a\ -1}|\}}.
\end{aligned} \qquad (58)$$

Next, we examine $\nabla_{\theta_\ell} \sum_{j\in J_t} p(j|x_t,\theta) H_t(\theta_j, a_{t,j}) = p_{\ell,t}(\theta)\nabla_{\theta_\ell} H_t(\theta_\ell, a_{t,\ell}) - 0.5\Delta_{\ell,t}(\theta)\nabla_{\theta_\ell} \ln|\Gamma_\ell|$ and $\Gamma_\ell = \Psi^{x\ -1} + \Psi_\ell^{a\ -1}$, with help of considering $d\ln|\Gamma_j| = -\mathrm{Tr}[\Psi^{x\ -1}\Gamma_j^{-1}\Psi^{x\ -1} d\Psi^x] - \mathrm{Tr}[\Psi_j^{a\ -1}\Gamma_j^{-1}\Psi_j^{a\ -1} d\Psi_j^a]$, with the following details:

$$\begin{aligned}
&p_{\ell,t}(\theta) = \Delta_{\ell,t}(\theta)\\
&\quad + \begin{cases} p(\ell|\theta, x_t), & \text{(a) unsupervised,}\\ \sum_{j\in J_t} p(j|\theta, x_t)\delta_{\ell,j}, & \text{(b) in general.}\end{cases}\\
&\Delta_{\ell,t}(\theta)\\
&\quad = \begin{cases} p(\ell|\theta, x_t)\delta H_{\ell,t}(\theta), & \text{(a) unsupervised,}\\ \sum_{j\in J_t} p(j|\theta, x_t)\Delta_{\ell,t,j}(\theta), & \text{(b) in general;}\end{cases} \qquad (59)\\
&\quad \delta H_{\ell,t}(\theta) = H_t(\theta_\ell^x, a_{t,\ell}) - \textstyle\sum_j p(j|\theta, x_t) H_t(\theta_j^x, a_{t,j}),\\
&\quad \Delta_{\ell,t,j}(\theta) = H_t(\theta_j^x, a_{t,j})\left[\delta_{\ell,j} - p(\ell|\theta, x_t)\right],\\
&\quad \text{where } \delta_{\ell,j} = 1 \text{ if } \ell = j, \text{ otherwise } \delta_{\ell,j} = 0 \text{ if } \ell \neq j.
\end{aligned}$$

Let the above gradients to be zero, we get the following updating formulas:

$$\begin{aligned}
&\eta_j^{y*} = \tfrac{1}{N}\textstyle\sum_t p_{j,t}(\theta^{\mathrm{old}}),\ \eta_j^{a*} = \tfrac{1}{N\eta_j^{y*}}\textstyle\sum_t p_{j,t}(\theta^{\mathrm{old}}) a_{t,j}^{\mathrm{old}},\\
&\Psi^{x*} = \tfrac{1}{N}\textstyle\sum_{t,j} p_{j,t}(\theta^{\mathrm{old}})(x_t - a_{t,j}^{\mathrm{old}})(x_t - a_{t,j}^{\mathrm{old}})^{\mathrm{T}}\\
&\quad + \tfrac{1}{N}\textstyle\sum_{t,j} \Delta_{j,t}(\theta^{\mathrm{old}})(\Psi^{x\ -1} + \Psi_j^{a\ -1})^{-1}, \qquad (60)\\
&\Psi_j^{a\ *} = \tfrac{1}{N\eta_j^{y*}}\textstyle\sum_t p_{j,t}(\theta^{\mathrm{old}})(a_{t,j}^{\mathrm{old}} - \eta_j^{a*})(a_{t,j}^{\mathrm{old}} - \eta_j^{a*})^{\mathrm{T}}\\
&\quad + \tfrac{1}{N}\textstyle\sum_t \Delta_{j,t}(\theta^{\mathrm{old}})(\Psi^{x\ -1} + \Psi_j^{a-1})^{-1}.
\end{aligned}$$

Putting together Eqs. (57), (58), (59), and (60), we get an iterative algorithm for the BYY harmony learning on a generalized Gaussian mixture. Interestingly, it follows from Eq. (56) that each sample x_t is smoothed by its mean vector η_j^a proportional to their precision matrices $\Psi^{x\ -1}$ and $\Psi_j^{a\ -1}$. This generalized Gaussian mixture returns back to a standard Gaussian mixture when we force $\Psi^x = 0$, and the above algorithm returns to the BYY harmony learning algorithm in Sect. 3.1 of Ref. [1]. That is, we have $a_{t,j} = x_t$ and that Eqs. (57), (58), (59), and (60) are simplified as follows

$$\begin{aligned}
&H_t(\theta_j, a_j) = \ln\{G(x_t|\eta_j^a, \Psi_j^a)\eta_j^y\},\\
&p(j|\theta, x_t) = \tfrac{G(x_t|\varphi_j, \Psi_j^a)\eta_j^y}{\sum_j G(x_t|\eta_j^a, \Psi_j^a)\eta_j^y},\ \eta_j^{y*} = \tfrac{1}{N}\textstyle\sum_t p_{j,t}(\theta^{\mathrm{old}}),\\
&\eta_j^{a*} = \tfrac{1}{N\eta_j^{y*}}\textstyle\sum_t p_{j,t}(\theta^{\mathrm{old}}) x_t, \qquad (61)\\
&\Psi_j^{a\ *} = \tfrac{1}{N\eta_j^{y*}}\textstyle\sum_t p_{j,t}(\theta^{\mathrm{old}})(x_t - \eta_j^{a*})(x_t - \eta_j^{a*})^{\mathrm{T}},
\end{aligned}$$

where $p_{t,l}(\theta)$ is still given by Eq. (59). As a complementary to Sect. 3.1 of Ref. [1], $p_{t,l}(\theta)$ in Eq. (59) is featured with the sum over J_t given by Eq. (57) such that supervised learning, unsupervised learning and semi-supervised learning are covered in a unified formulation.

3.2 Independent factor analysis, manifold learning, and semi-blind learning

From $X = AY + E$ featured with $q(X|\boldsymbol{\eta}^x, \Psi^x)$ by Eq. (10) and $q(Y|\Theta^y)$ by Eq. (16), we get a typical family

of learning models, as illustrated by the Boxes ①–④ in Fig. 1 and introduced in Sect. 2.1, especially around Eq. (18), Eq. (19) and Eq. (20). Further improvements can be obtained via exploring the co-dim matrix pair nature with help of $q(A|\boldsymbol{\eta}^a, \Psi^a)$ by Eq. (24), for implementing automatic model selection by Eq. (28) or Eq. (36)(a) and sparse learning by Eq. (35) or Eq. (36)(b).

Specifically, the learning procedure by Eq. (43) is simplified for learning the co-dim matrix pair featured FA by Eq. (48) as follows:

$$\begin{array}{l}(a)\ \text{update } Y^* \text{ by Eq. (49);}\\ (b)\ \text{update } A^* \text{ by Eq. (52) or Eq. (51);}\\ (c)\ \text{update } \Psi^{x*}, \Psi^{y*}, \Psi^{a*}_j \text{ by Eq. (53);}\\ (d)\ \text{Discard the } j\text{-th row of } Y \text{ via checking}\\ \quad \text{either Eq. (28)(a) or Eq. (36)(a)}\\ \quad \text{Prune } a_{i,j} \text{ of } A \text{ via checking either}\\ \quad \text{Eq. (35) or Eq. (36)(b)}\\ (e)\ \textit{(optional)}\ \text{use Eq. (29) to select the best } m^*,\end{array} \tag{62}$$

which is implemented iteratively until converged. The algorithm may also be approximately used for NFA [9–13] for real valued Y featured with that at most one $y_{i,t}$ per column is Gaussian, e.g., $y_{i,t}$ comes from distributions of exponential, Gamma, a mixture of Gaussians. There are two points of modifications. First, Step (a) is modified with

$$Y^* = \text{argmax}_Y \ln[q(X_N|\boldsymbol{\eta}(AY), \Psi^x)q(Y|\boldsymbol{\eta}^y, \Psi^y)], \tag{63}$$

in Eq. (44) solved by an iteration. Second, we approximately let

$$\Psi^y \approx \frac{\partial^2 \ln q(y|\mathbf{0}, \Psi^y)}{\partial \boldsymbol{y} \partial \boldsymbol{y}^{\mathrm{T}}}.$$

Moreover, the above learning algorithm may be modified for implementing the BFA [5–8] with each $y_{i,t} = 0$ or $y_{i,t} = 1$ from Eq. (19). Again Step (a) is modified with Eq. (63), which is now a quadratic combinatorial optimization which can be effectively handled by the algorithms investigated in Ref. [99]. This optimization can be handled simply by enumeration for implementing exclusive binary FA as the binary factorization Eq. (19) becomes the multi-class problem by Eq. (20). If we let $q(a_j|\boldsymbol{\eta}^a_j, \Psi^a)$ in Eq. (24) and Eq. (48) replaced with

$$\begin{array}{l}q(a_j|\boldsymbol{\eta}^a_j, \Psi^a) = \prod_i (\eta^a_{i,j})^{a_{i,j}},\\ \eta^a_{i,j} \geqslant 0, \sum_i \eta^a_{i,j} = 1, \quad a_{i,j} \geqslant 0, \quad \sum_i a_{i,j} = 1,\end{array} \tag{64}$$

we are further lead to the binary matrix factorization (BMF) based bi-clustering [28], for which A^* is obtained in a way similar to get Y^*.

Moreover, instead of using Eq. (29), Step (e) use to select the following criterion for selecting the best m^*:

$$\begin{array}{l}J(m) = \frac{1}{2}\ln|\Psi^x| + \frac{h^2}{2}\text{Tr}[\Psi^{x\ -1}] - J^y_m + \frac{d}{N}J^a_m,\\ J^y_m = \begin{cases}\sum_j \eta^y_{j,t} \ln \eta^y_{j,t},\\ \sum_j (1-\eta^y_{j,t}) \ln (1-\eta^y_{j,t}), & \text{for Eq. (19),}\\ \sum_j \eta^y_{j,t} \ln \eta^y_{j,t}, & \text{for Eq. (20).}\end{cases}\\ J^a_m = \begin{cases}m\ln(2\pi \mathrm{e}) + \ln|\Psi^a|, & \text{for in Eq. (48),}\\ -\sum_i \eta^a_{i,j} \ln \eta^a_{i,j}, & \text{for Eq. (64).}\end{cases}\end{array} \tag{65}$$

Instead of $q(Y|\Theta^y)$ by Eq. (16), $X = AY + E$ featured with $q(X|\boldsymbol{\eta}^x, \Psi^x)$ by Eq. (10) may be used for modeling $q(Y|\boldsymbol{\eta}^y, \Psi^y)$ to preserve topological dependence among data, as illustrated in the Box ⑩ in Fig. 1.

One popular way to describe a local topology among a set of data (equivalently the columns of X) is to get a nearest neighbor graph G of N vertices with each vertex corresponding to a column of X. Define the edge matrix S as follows:

$$S_{ij} = \begin{cases}\frac{1}{\sqrt{2\pi}\gamma} \exp\{\frac{-0.5\|x_i - x_j\|^2}{\gamma^2}\},\\ \text{if } x_i \in N_k(x_j) \text{ or } x_j \in N_k(x_i),\\ 0, \text{ otherwise,}\end{cases}$$

for a pre-specific γ, where $N_k(x_i)$ denotes a set of k nearest neighbors of x_i. We have $\boldsymbol{L} = \boldsymbol{D} - \boldsymbol{S}$ that is called a graph Laplacian and positively definite [37,38,101–103], where $\boldsymbol{D}$ is a diagonal matrix whose entries are $\boldsymbol{D}_{ii} = \sum_j \boldsymbol{S}_{ij}$. Considering a mapping $Y \approx WX$, a locality preserving projection (LPP) attempts to minimize $\text{Tr}[WX^{\mathrm{T}}LWX]$, i.e., the sum of each distance between two mapped points on the graph G, subject to a unity L_2 norm of this projection WX.

Alternatively, we may regard that X is generated via $X = AY + E$ such that the topological dependence among Y is preserved, and thus handle this problem as one extension of FA, as shown from the center toward right to the box ⑩. To be specific, we consider $q(X|\boldsymbol{\eta}^x, \Psi^x)$ by Eq. (10) with $\boldsymbol{\eta}^x = AY$, as well as Eq. (24) with $q(a_j|\boldsymbol{\eta}^a_j, \Psi^a) = G(a_j|\mathbf{0}, \Psi^a)$. Instead of Eq. (16), we let $q(Y|\boldsymbol{\eta}^y, \Psi^y)$ to be

$$\begin{array}{l}q(Y) = \frac{1}{Z(L)} \exp\{-\frac{1}{2}\text{Tr}[YLY^{\mathrm{T}}]\},\\ Z(L) = \int \exp\{-\frac{1}{2}\text{Tr}[YLY^{\mathrm{T}}]\}\mathrm{d}Y,\end{array} \tag{66}$$

where the Laplacian L is known from a nearest neighbor graph G, and also $Z(L)$ is correspondingly known.

The learning is implemented again by modifying the above learning algorithm by Eq. (62). Instead of getting update Y^* by Eq. (49) in Step (a), Eq. (49) is modified into the following one for updating Y^*:

$$\begin{array}{l}\nabla_Y \ln Q_{X_N,A,Y|\Theta} = A^{\mathrm{T}}\Psi^{x\ -1}X - YL - \Gamma_a Y,\\ \Gamma_a = A^{\mathrm{T}}\Psi^{x\ -1}A,\\ \Omega_{Y|A,\Theta} = \Gamma_a \otimes I + L,\end{array} \tag{67}$$

from which Y^* is solved from the following equation

$$A^{\mathrm{T}}\Psi^{(1)\ -1}X - Y^*L - \Gamma_a Y^* = 0. \quad (68)$$

Alternatively, we may also update Y by a gradient based searching with the help of Eq. (67).

It is interesting to further experimentally and theoretically compare whether the above Y^* outperforms its counterpart obtained by the existing LPP approach for manifold learning, at least with two new features. First, the reduced dimension of the manifold of Y^* may be determined by automatic model selection. Second, regularization is made via $q(A|\boldsymbol{\eta}^a, \Psi^a)$ by Eq. (24).

As outlined in Sect. 1, semi-blind learning is a better name for efforts that put attention on the cases of knowing partially either or both of the system and its input, instead of just knowing partially the inputs by semi-supervised learning. E.g., we know not only X, but also partial knowledge about A, Y, and E for Eq. (4). For the above manifold learning, Y is unknown but it is assumed that its covariance information Ψ^y is given by the Laplacian L, that is, it is actually an example of semi-blind learning.

On the other hand, even for the problem of linear regression by $X = AY + E$ with both X and Y known, we may turn the problem in a semi-blind learning when concerning a small sample size (i.e., the column of X) or a unreliable relation given by a known pair X and $\hat{Y}$. In sequel, two methods are suggested.

- Semi-blind learning FA Instead of directly using the known Y^* in pairing with X, we let $q(y_t|\eta_t^y, \Psi_t^y)$ in Eq. (48) replaced with

$$q(y_t|\eta_t^y, \Psi_t^y) = G(y_t|\hat{y}_t, \Psi^y),\ \Psi^y = \hat{Y}\hat{Y}^{\mathrm{T}}, \quad (69)$$

  then, we use the algorithm by Eq. (62) for learning with $\hat{y}_t, \Psi^y$ fixed without updating.
- Semi-blind learning BFA as illustrated in the Box ⑮ in Fig. 1, with $\hat{Y}$ denoting a known instance of Y, we let a matrix of binary latent variables to take the position of Y, which leads to the following $\hat{Y}$ modulated binary FA:

$$\begin{aligned} &X = AY_H + E,\ Y_H = \hat{Y} \circ Y, \\ &\hat{Y} \circ Y = \begin{cases} [\hat{y}_1 \circ y, \cdots, \hat{y}_N^* \circ y], & \text{(a) } Type\ 1, \\ [\hat{y}_{j,t}^* y_{j,t}], & \text{(b) } Type\ 2, \end{cases} \end{aligned} \quad (70)$$

  where $A \circ B = [a_{ij}b_{ij}]$ and Y comes from the following distribution

$$\begin{aligned} &q(Y|\boldsymbol{\eta}^y, \Psi^y) = B(Y|\boldsymbol{\eta}^y), \\ &B(Y|\boldsymbol{\eta}^y) = \prod_{t,j}(\eta_j^y)^{y_{j,t}}(1-\eta_j^y)^{1-y_{j,t}}, \\ &B(Y|\boldsymbol{\eta}^y) = \prod_{t,j}(\eta_{j,t}^y)^{y_{j,t}}. \end{aligned} \quad (71)$$

  Still, we may use the algorithm by Eq. (62) for learning, with the above $q(Y|\boldsymbol{\eta}^y, \Psi^y)$ putting in Eq. (63) to modify Step (a) for getting Y^* via a quadratic combinatorial optimization algorithm [99]. Then, we use $Y_H = \hat{Y} \circ Y^*$ to take the place of Y^* in the rest steps in Eq. (62).

The above two methods are motivated for dealing with different uncertainty. Semi-blind learning FA considers that $\hat{Y}$ suffers Gaussian noises, while semi-blind learning BFA considers that some elements of $\hat{Y}$ are pseudo values and thus we need to remove their roles with $y_{i,t} = 0$.

3.3 Graph matching, covariance decomposition, and data-covariance co-decomposition

After an extensive investigation on Eq. (4) in Sects. 3.1 and 3.2, we move to consider Eq. (9), and then make a coordinated study on Eqs. (4) and (9).

We start at considering two attributed graphs X and Y described by two matrices S_X and S_Y. Each diagonal element of S_Y is a number as one attribute attached to one node in the graph Y, where each off-diagonal element of S_Y is a number as an attribute attached to the edge between two nodes in Y. Moreover, S_Y is a symmetric matrix if Y is a unidirectional graph. Every element in Y can even be nonnegative, e.g., for a network of protein-protein interaction in biology to be discussed in Sect. 5.3. Two graphs are said to be matched exactly or isomorphism, if S_X and S_Y become same after a permutation of the nodes of one graph, namely $S_X = AS_YA^{\mathrm{T}}$ by an appropriate permutation matrix A. For an arbitrary permutation matrix A, we have usually $\Sigma_X \neq AS_YA^{\mathrm{T}}$ or $\Sigma = S_X - AS_YA^{\mathrm{T}} \neq 0$. The problem becomes seeking one among all the possible decompositions $S_X = \Sigma + AS_YA^{\mathrm{T}}$ as in Eq. (9) such that $\Sigma = 0$. This solution can be obtained when the Frobenius norm of Σ or equivalently $\mathrm{Tr}[\Sigma\Sigma^{\mathrm{T}}]$ reaches its minimum. A match between X and Y is thus formulated as

$$\min_{A\in\Pi} \mathrm{Tr}[(S_X - AS_YA^{\mathrm{T}})(S_X - AS_YA^{\mathrm{T}})^{\mathrm{T}}], \quad (72)$$

where Π consists of all permutation matrices. If the minimum is 0, we have $\Sigma = 0$ or two graphs are matched exactly. This minimization involves searching all the possible permutations and is a well known NP-hard problem. The problem is usually tackled by a heuristic searching, e.g., a simulated annealing, with a permutation matrix A that gives $\mathrm{Tr}[\Sigma\Sigma^{\mathrm{T}}] \neq 0$, which is made under the name inexact graph matching, widely studied in the literature of pattern recognition in past decades [50–54].

One direction for approximation solution was started from Umeyama in 1988 [50]. The permutation set Π is only a small subset within the Stiefel manifold O_Π of orthonormal matrices. Considering the minimization with respect to an orthonormal matrix in O_Π, the solution A can be obtained by an eigen-analysis $S_XA = AS_Y$. Though this solution is too rough for an exact graph

matching, it may be still good enough for building structural pattern classifiers, as suggested in 1993 [53]. This was further verified by fast retrieval of structural patterns in databases [54]. Also, it was suggested in Ref. [53] that eigen-analysis is simply replaced by updating A along the direction $S_X A S_Y - A S_Y A S_X$, which performs a constrained gradient based searching for minimizing $\mathrm{Tr}[\Sigma\Sigma^{\mathrm{T}}]$ with respect to $A \in O_\Pi$.

The direct relaxation from a permutation to orthonormal matrix goes too far, violating both the nature that all elements are nonnegative and the nature that all rows and columns sum to one. Instead, a relaxation from a permutation matrix to a doubly stochastic matrix can still retain both the natures, which is also justified from a perspective that the minimization of a combinatorial cost is turned into a procedure of learning a simple distribution to approximate Gibs distribution induced from this cost [104]. From this new perspective, a general guideline was proposed for developing a combinatorial optimization approach, and the Lagrange-enforcing algorithms were developed in Refs. [105,106] with guaranteed convergence on a feasible solution that satisfies constraints.

During proving convergence property of one theorem about ICA, it was further found in Refs. [107,108] that turning a doubly stochastic matrix $V = [v_{ij}]$ into an orthostochastic matrix $[r_{ij}^2]$ facilitates to make optimization by a Stiefel gradient flow. Moreover, this orthostochastic matrix based implementation leads to a favorable spare nature that pushes v_{ij} to be 0 or 1 when we consider a combinatorial cost that consists terms of r_{ij} in higher than quadratic order, e.g., a TSP problem [100]. A brief overview is referred to Sect. 3 in Ref. [108]. Also, either the problem by Eq. (72) or combinatorial optimization in general has been examined from a Bayesian Ying-Yang (BYY) learning perspective in Ref. [100]. In the sequel, we further investigate graph matching with modified formulations and also the use of one priori $q(A|\boldsymbol{\eta}^a, \Psi^a)$ by Eq. (24) with the help of the BYY harmony learning.

As discussed after Eq. (72), two graphs are matched exactly only when $\Sigma = 0$, while minimizing $\mathrm{Tr}[\Sigma\Sigma^{\mathrm{T}}]$ means the sum of the square norms of all the elements in Σ is minimized as a whole. A heuristic solution with $\mathrm{Tr}[\Sigma\Sigma^{\mathrm{T}}] \neq 0$ means that the minimization of $\mathrm{Tr}[\Sigma\Sigma^{\mathrm{T}}]$ leads to a solution that may be far from the exact graph matching. To push $\mathrm{Tr}[\Sigma\Sigma^{\mathrm{T}}]$ toward zero, we further impose that Σ should be diagonal in order to enhance the match between the topologies of two graphs, which is implemented via minimizing the following error

$$\min_{A\in\Pi} J(S_X, A) = \sum_{i,j} \gamma_{i,j}(s_{i,j}^x - \psi_{i,j}^x)^2 \tag{73}$$

$$J(S_X, A) = (1-\chi)\mathrm{Tr}\{\mathrm{diag}[\Sigma]\mathrm{diag}[\Sigma]\} + \chi\mathrm{Tr}\{\Sigma\Sigma^{\mathrm{T}}\},$$

$$\Sigma = S_X - A S_Y A^{\mathrm{T}},\ S_X = [s_{i,j}^x],\ [\psi_{i,j}^x] = A S_Y A^{\mathrm{T}},$$

$$\gamma_{i,j} = \begin{cases} 1, & \text{if } i = j, \\ \chi, & \text{if } i \neq j; \end{cases}$$

where $\chi > 0$ is a pre-specified number. We are led back to Eq. (72) when $\chi = 1$. Its implementation may be made in one of the following choices:

1) Similar to Ref. [50], $A \in \Pi$ is relaxed to an orthonormal matrix $A \in O_\Pi$, Eq. (73) is solved by a generalized eigen-analysis.

2) Similar to Ref. [53], we get the gradient of $J(S_X, A)$ with respect to an orthonormal matrix $A \in O_\Pi$ and update A by a gradient descending searching.

3) Following Refs. [107,108], we replace A by an orthostochastic matrix, that is we consider

$$A = [r_{i,j}^2],\ \text{from } R = [r_{i,j}] \text{ with } RR^{\mathrm{T}} = I. \tag{74}$$

Then, we get the gradient of $J(S_X, A)$ with respect to an orthonormal matrix $R \in O_\Pi$ via the above Eq. (74) and update R by a gradient descending searching.

We may have an alternative of Eq. (73) as follows:

$$\begin{aligned} &\min_{A\in\Pi} J(S_X, A), \\ &J(S_X, A) = \textstyle\sum_{i,j} \gamma_{i,j} s_{i,j}^x \psi_{i,j}^x, \\ &S_X = [s_{i,j}^x],\ A S_Y A^{\mathrm{T}} = [\psi_{i,j}^x], \end{aligned} \tag{75}$$

which is equivalent to Eq. (73) when S_X and S_Y are given. However, Eq. (75) is preferred if S_X and S_Y differ with a unknown scale or have partially unknowns.

Moreover, we consider the BYY harmony learning for handling the problems. For the case by Eq. (9) with samples of X from Gaussian, we may consider that $S_X = XX^{\mathrm{T}}$ comes from the following Wishart distribution

$$q(S_X|A) = \frac{|S_X|^{\frac{(N-d-1)}{2}} exp\{-\frac{1}{2}\mathrm{Tr}[(A S_Y A^{\mathrm{T}} + \Sigma)^{-1} S_X]\}}{2^{Nd/2}|A S_Y A^{\mathrm{T}} + \Sigma|^{N/2}\Gamma_d(N/2)}, \tag{76}$$

where $\Gamma_d(\cdot)$ is the multivariate gamma function, and Σ is a unknown diagonal or even $\Sigma = \sigma^2 I$.

It can be observed that $\max_{A\in\Pi} q(S_X|A)$ makes $A S_Y A^{\mathrm{T}} + \Sigma$ tends to S_X, and thus it can be regarded as a general format of Eq. (72). To take Eq. (73) and also Eq. (75) in consideration, we may consider

$$q(S_X|A) = \frac{exp\{-0.5J(S_X, A)\}}{\int exp\{-0.5J(S_X, A)\}\mathrm{d}S_X}. \tag{77}$$

From Eq. (44), we get the harmony measure $H(p||q) = \int p(A|S_X)\ln[q(S_X|A)q(A|\eta^a, \Psi^a)]\mathrm{d}A$. During removing $\int \cdots \mathrm{d}A$, we perform

$$\begin{aligned} &A = R \circ R = [r_{ij}^2], \\ &R^* = \mathrm{argmax}_{RR^{\mathrm{T}}=I} \ln[q(S_X|A)q(A|\eta^a, \Psi^a)], \\ &\Sigma^* = \mathrm{argmax}_{\Sigma} \ln q(S_X|A), \end{aligned} \tag{78}$$

from which we obtain one solution $A^* = R \circ R = [r_{ij}^{*\,2}]$, where sparse learning is considered via $q(A|\boldsymbol{\eta}^a, \Psi^a)$ for

such that A^* is close to a permutation matrix, e.g., Eq. (24) with $q(a_j|\eta_j^a, \Psi^a)$ from a multivariate Gaussian or Laplace.

Similar to Eq. (69), we consider the following Wishart distribution

$$q(S_Y|\hat{S}_Y) = \frac{|\hat{S}_Y|^{\frac{(N-m-1)}{2}} exp\{-\frac{1}{2}\mathrm{Tr}[\hat{S}_Y^{-1}S_Y]\}}{2^{Nm/2}|AS_YA^{\mathrm{T}}+\Sigma|^{N/2}\Gamma_m(N/2)}, \quad (79)$$

by which matching S_X based on a given $\hat{S}_Y$ is relaxed to certain tolerance on an inaccurate S_Y.

Correspondingly, we extend Eq. (78) to take Eq. (79) in consideration as follows

$$\begin{aligned} H(p||q) &= \int p(A|S_X)\ln[Q_{X|A}q(A|\eta^a,\Psi^a)]\mathrm{d}A \\ \ln Q_{X|A} &= H(p||q,A) = \int p(S_Y|A,S_X)\times \\ &\qquad \ln[q(S_X|A)q(S_Y|\hat{S}_Y)]\mathrm{d}S_Y \\ A &= R\circ R = [r_{ij}^2], \\ R^* &= \mathrm{argmax}_{RR^{\mathrm{T}}=I}\ln[Q_{X|A}q(A|\eta^a,\Psi^a)], \\ \Sigma^* &= \mathrm{argmax}_{\Sigma}\ln q(S_X|A), \end{aligned} \quad (80)$$

where $p(S_Y|A,S_X)$ and $q(S_Y|\hat{S}_Y)$ can be a conjugate pair, from which we get R^* and obtain a solution $A^* = R\circ R = [r_{ij}^{*\,2}]$.

Strictly speaking, the above problem by Eq. (72) relates to the covariance decomposition problem by Eq. (9) but cannot be simply regarded as its special case that S_X and S_Y are known and A is a permutation matrix, because S_X and S_Y from unidirectional graphs X and Y are symmetric but may not meanwhile positive definite.

In some applications, there may be available both a partial information about data X and a partial information about its covariance S_X, which motivates to integrate two parts of information. Since both X and S_X are generated from the same system by Eq. (10), the key point is to model $q(S_X, X|\boldsymbol{\eta}^x, \Psi^x)$. There are two possible roads to proceed. One is consider

$$q(S_X, X|\boldsymbol{\eta}^x, \Psi^x) = q(S_X|X, \Psi^s)q(X|\boldsymbol{\eta}^x, \Psi^x), \quad (81)$$

with $q(X|\boldsymbol{\eta}^x, \Psi^x)$ by Eq. (10) and $q(S_X|X, \Psi^s)$ describing some uncertainty between S_X and X. If we are sure on $S_X = XX^{\mathrm{T}}$, we have $q(S_X|X, \Psi^s) = \delta(S_X - XX^{\mathrm{T}})$, S_X does not bring extra information, and thus there is no need for integration. Due to observation noise and a small sample size, we have $q(S_X|X, \Psi^s)$ described by a distribution, e.g., a Gaussian

$$q(S_X|X, \Psi^s) = G(S_X|XX^{\mathrm{T}}, \rho^2 I). \quad (82)$$

The other road is jointly considering Eq. (4) and Eq. (9). Both X and S_X comes the same system, and thus we have $q(X|\boldsymbol{\eta}^x, \Psi^x)$ for Eq. (4) by Eq. (10) and $q(S_X|\boldsymbol{\eta}^x, \Psi^x) = q(S_X|A)$ for Eq. (9) by either Eq. (76) and Eq. (77). The key point is how to combine the two ones to get $q(S_X, X|\boldsymbol{\eta}^x, \Psi^x)$, or equivalently, making a co-decomposition of data matrix by Eq. (4) and covariance matrix by Eq. (9). As discussed after Eq. (9), Eq. (4) implies Eq. (9) if the condition by Eq. (3) holds. However, Eq. (3) may not hold well in practice, and thus such a co-decomposition actually provides an alternative way to ensure Eq. (3) indirectly.

Generally, we may regard S_X as coming from X via some symmetry preserved transform, e.g., with help of an element-wise mapping $g(XX^{\mathrm{T}}|\Xi_g)$ by an unknown scalar monotonic parametric function $g(r|\Xi_g)$. We may get the distribution of S_X from one distribution of XX^{T} via the Jacobian matrix induced from this scalar monotonic function and also get the parameters Ξ_g estimated also through learning. That is, we have

$$\begin{aligned} q(S_X|X, \Psi^s) &= G(S_X|g(X|\Xi_g), \rho^2 I), \\ g(X|\Xi_g) &= g(XX^{\mathrm{T}}|\Xi_g) \\ or\ g(X|\Xi_g) &= g(X|\Xi_g)g^{\mathrm{T}}(X|\Xi_g). \end{aligned} \quad (83)$$

There could be different ways to combine two distributions into one joint distributions [109,110]. A typical one is the naïve Bayesian rule or called the product rule, that is , we have

$$q(S_X, X|\boldsymbol{\eta}^x, \Psi^x) = \frac{q(S_X|\boldsymbol{\eta}^x,\Psi^x)q(X|\boldsymbol{\eta}^x,\Psi^x)}{\int q(S_X|\boldsymbol{\eta}^x,\Psi^x)q(X|\boldsymbol{\eta}^x,\Psi^x)\mathrm{d}S_X\mathrm{d}X}. \quad (84)$$

With help of Eqs. (81) and (84), we put $q(S_X, X|\boldsymbol{\eta}^x, \Psi^x)$ into Eq. (38) to replace $q(X|\boldsymbol{\eta}^x, \Psi^x)$, we may implement the BYY harmony learning to perform co-decomposition of data by Eq. (4) and covariance by Eq. (9).

4 BYY harmony learning with hierarchy of co-dim matrix pairs

4.1 Hierarchy of co-dim matrix pairs and bidirectional propagation

According to the natures of learning tasks, the building unit by Eq. (4) and Eq. (10), as well as the corresponding BYY system shown in Fig. 2 may further get some top-down support. Such a support may come from getting a prior $q(\Theta|\Xi)$ to be put in Eq. (7), as previously discussed in Sect. 2. Moreover, each component of this building unit may be the output of another co-dim matrix pair. E.g., in Eq. (70) we have $X = AY_H + E,\ Y_H = \hat{Y}\circ Y$, with $\hat{Y}$ given a fixed value. In general, we have a co-dim matrix pair with both matrices unknown.

Moreover, either or both of $\boldsymbol{\eta}^y, \boldsymbol{\eta}^a$ may also itself be the output of another co-dim matrix pair, e.g., in a format of Eq. (4) or a degenerated version, which may be regarded as taking a role of a structural prior. As to be introduced in Sect. 4.2, with $\boldsymbol{\eta}^y = \eta^y(\boldsymbol{\varepsilon}, B)$ in a co-dim matrix pair, we are led to a generalization of temporal FA and state space model. Also, with each $\boldsymbol{\eta}_j^a = \eta^a(\boldsymbol{\zeta}, \Phi)$ in a co-dim matrix pair, the de-noise

Gaussian mixture in Sect. 3.1 is further extended to a de-noise version of local FA. So on and so forth, one new layer may have its next new layer. Whether we add a new upper layer depends on if there is some priors available or we want to stop for simplifying computation. As shown in Fig. 3, we generally get a BYY system featured by a hierarchy of co-dim matrix pairs.

The first layer is same as the BYY system in Fig. 2, with two differences in notations. First, the superscript "$^{(1)}$" is added to indicate the first layer. Second, the studies in the previous sections consider the BYY system in Fig. 2 usually with $\boldsymbol{\eta}^y, \boldsymbol{\eta}^a$ by Eq. (39) and Eq. (40); while the place of $q(\Psi|\Xi)$ in Eq. (37) or $q(\Psi)$ in Fig. 2 is taken by $q(\Psi|\Xi)q(R^{(2)})$ in the first layer in Fig. 3 as an entry from the second layer.

The second layer consists of two co-dim matrix pairs. One is featured by $\eta^y(\boldsymbol{\varepsilon}, B)$ as the input to $\boldsymbol{\eta}^y$ in the first layer while the other is featured by $\eta^a(\boldsymbol{\zeta}, \Phi)$ as the input to $\boldsymbol{\eta}_j^a$. Each co-dim matrix pair is expressed in a format similar to the first layer except the different notations. In typical learning tasks, both the pairs may not coexist. In the above examples, there is merely $\boldsymbol{\eta}^y = \eta^y(\boldsymbol{\varepsilon}, B)$ for a generalization of temporal FA and state space model, while merely $\boldsymbol{\eta}_j^a = \eta^a(\boldsymbol{\zeta}, \Phi)$ for a de-noise version of local FA. Furthermore, similar to the relation between Eqs. (4) and (9), either Ψ^y or Ψ^a may also itself be the output of a quadratic system in a format of Eq. (9), e.g., $B^{\mathrm{T}}\Psi_Y^{(2)}B + \Psi^y$. For clarity, we omitted this aspect in Fig. 3.

Similarly, each pair in the second layer may be supported by two co-dim matrix pairs from the third layers. Thus, the third layer consists of four co-dim matrix pairs, as sketched in Fig. 3. So on and so forth, we generally get a hierarchy of co-dim matrix pairs. For each pair, one or more component may be degenerated, e.g., we have either or both of $\boldsymbol{\eta}_j^a = 0$ and $\Psi^a = 0$ for the pair of the first layer.

On the right side of Fig. 3, the information flow is top-down, with an order one Markovian nature, i.e., one layer decouples its upper layers to its lower layers. On the left side of Fig. 3, the structure of $p(R|X)$ is designed as a functional with $q(X|R)$, $q(R)$ as its arguments according to a Ying-Yang variety preservation principle, see Sect. 4.2 in Ref. [1]. In contrast, the information flow is bottom up. As a Bayesian inverse of the Ying part, the Yang part on the jth layer depends the representation of $R^{(j-1)}$. For computational simplicity, we may approximately regard that the bottom up information flow has an order one Markovian nature, as shown on the left side of Fig. 3.

Additionally, there may be also a global support $q(\Xi)$ that provides a priori to every layer, especially to the parameters in Ψ. For simplicity, we may consider improper priories without hyper-parameters, e.g., a Jeffrey or IBC prior, see Sect. 4.2 in Ref. [1].

The implementation of the BYY harmony learning can be made per layer and per pair, bottom up in a decoupled manner. Taking $\boldsymbol{\eta}^y = \eta^y(\boldsymbol{\varepsilon}, B)$ as an example, after

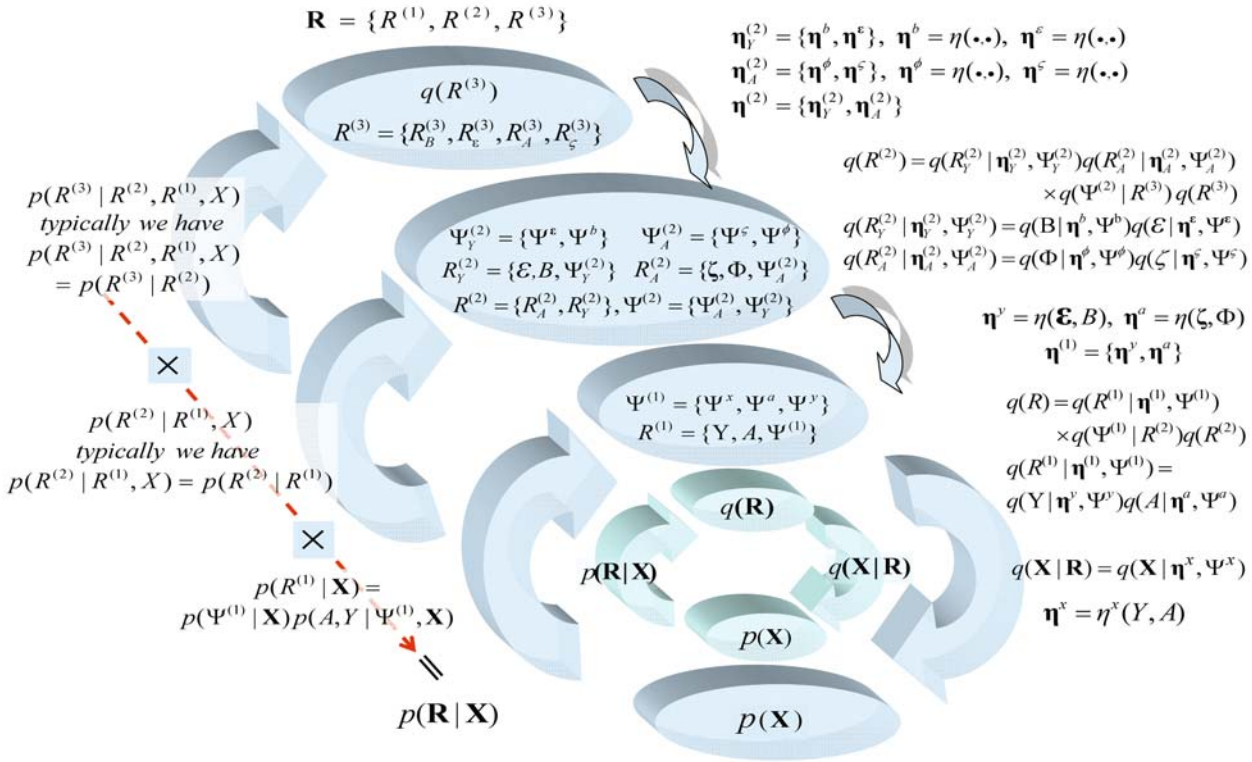

Fig. 3 BYY harmony learning with hierarchy of co-dim matrix pairs

running Eq. (43) for learning the first layer, we get Y^* available via Eq. (44). Considering the following correspondences:

$$\begin{aligned} Y^* &\Leftrightarrow X_N, \\ q(Y^*|\eta^y(\boldsymbol{\varepsilon}, B), \Psi^y) &\Leftrightarrow q(X_N|\boldsymbol{\eta}(AY), \Psi^x), \\ q(\boldsymbol{\varepsilon}|\boldsymbol{\eta}^\varepsilon, \Psi^\varepsilon) &\Leftrightarrow q(Y|\boldsymbol{\eta}^y, \Psi^y), \\ q(B|\boldsymbol{\eta}^b, \Psi^b) &\Leftrightarrow q(A|\boldsymbol{\eta}^a, \Psi^a), \end{aligned}$$

we get the counterpart of Eq. (38) as follows:

$$\begin{aligned} H^{(2)}(p||q, \kappa, \Xi^{(2)}) &= \int p(\Psi_Y^{(2)}|Y^*) \\ &\quad \times \ln[Q_{Y^*|\Psi_Y^{(2)}}\ q(\Psi_Y^{(2)}|\Xi)]\mathrm{d}\Psi_Y^{(2)}, \\ \ln Q_{Y^*|\Psi_Y^{(2)}} &= H(p||q, Y^*, \Psi_Y^{(2)}) \\ &= \int p(B|\Psi_Y^{(2)}, Y^*) \\ &\quad \times \ln[Q_{Y^*,B|\Psi_Y^{(2)}}\ q(B|\boldsymbol{\eta}^b, \Psi^b)]\mathrm{d}B, \\ \ln Q_{Y^*,B|\Psi_Y^{(2)}} &= H(p||q, B, Y^*, \Psi_Y^{(2)}) \\ &= \int p(\boldsymbol{\varepsilon}|B, \Psi_Y^{(2)}, Y^*)\times \\ &\quad \ln[q(Y^*|\ \eta^y(\boldsymbol{\varepsilon}, B), \Psi^y)q(\boldsymbol{\varepsilon}|\boldsymbol{\eta}^\varepsilon, \Psi^\varepsilon)]\mathrm{d}\boldsymbol{\varepsilon}. \end{aligned} \tag{85}$$

Thus, its learning can be iteratively implemented in the same procedure as introduced from Eq. (43) to Eq. (53) in Sect. 2.3. For simplicity, we simply use the phrase " the co-dim matrix pair learning procedure by Eq. (43)".

On one hand, learning on Eq. (85) reply on $Y*$ obtained from learning the first layer on Eq. (38). On the other hand, learning the first layer also relates to the value of $\boldsymbol{\eta}^y$. We can not simply let $\boldsymbol{\eta}^y$ given by Eq. (39) and Eq. (40), but given from the second layer via $\boldsymbol{\eta}^y = \eta^y(\boldsymbol{\varepsilon}, B)$. Thus, two layers of learning are coupled. In other words, learning is made via two iterative loops we get the counterpart of Eq. (38) as follows:

$$\begin{aligned} &\textit{Loop}\ 1: \text{learning on Eq. (38) at the current top down input } \boldsymbol{\eta}^y \text{ by the co-dim matrix pair learning procedure by Eq. (43), send out a bottom up output } Y^*; \\ &\textit{Loop}\ 2: \text{learning on Eq. (85) at the current bottom up input } Y^* \text{ by the co-dim matrix pair learning procedure by Eq. (43), send out a top down output } \boldsymbol{\eta}^y. \end{aligned} \tag{86}$$

The two loops are jointly iterated until a convergence is reached, featured by a bidirectional propagation of learning interaction. Similar to automatic model selection and sparse learning as discussed after Eq. (43), we may let Ψ^b to replace Ψ^a in Eq. (35), and Ψ^ε to replace Ψ^a in Eq. (36) for automatic model selection.

Similarly, we may also make learning on the co-dim matrix pair featured by $\boldsymbol{\eta}_j^a = \eta^a(\boldsymbol{\zeta}, \Phi)$ for a de-noise version of local FA, as to be further introduced in the next subsection. In the same way, two layers of learning may also be implemented by Eq. (86) between the second layer and the third layer, as well as between any the j-th layer and the $j+1$-th layer in general. We repeat such iterations downwards and upwards, until getting converged or stopped according to an external rule. Such a bidirectional propagation may be made either in a sequential way or a systolic way by which the renewed parameters are propagated upwards and downwards once each layer is updated.

Next, we further justify Eqs. (85) and (86). Putting $\ln\{q(X_N|\boldsymbol{\eta}(AY), \Psi^x)q(A|\eta^a, \Psi^a)q(Y|\eta^y, \Psi^y)q(\Psi^{(1)}|\Xi)$ $q(\Psi^{(2)}|\Xi)q(B|\boldsymbol{\eta}^b, \Psi^b)q(\boldsymbol{\varepsilon}|\boldsymbol{\eta}^\varepsilon, \Psi^\varepsilon)\}$ in the place of $\ln[q(X|R)q(R)]$ into Eq. (2), in a way similar to Eq. (38) we remove the integrals over $A, Y, \Psi^{(1)}$ and obtain $H(p||q) = \int p(\Psi_Y^{(2)}, \boldsymbol{\varepsilon}, B|X_N, A, Y, \Psi^{(1)}) \ln\{Q_{X_N|\Psi_Y^{(2)},\boldsymbol{\varepsilon},B}$ $\times q(\Psi_Y^{(2)}|\Xi)q(B|\boldsymbol{\eta}^b, \Psi^b)q(\boldsymbol{\varepsilon}|\boldsymbol{\eta}^\varepsilon, \Psi^\varepsilon)\}\mathrm{d}\Psi_Y^{(2)}\mathrm{d}\boldsymbol{\varepsilon}\mathrm{d}B$, where $\ln Q_{X_N|\Psi_Y^{(2)},\boldsymbol{\varepsilon},B}$ degenerates to $H(p||q, m, \Xi)$ in Eq. (38) if there is no the second layer and thus $A, Y, \Psi^{(1)}$ are discarded. In other words, the learning on the part of $\ln Q_{X_N|\Psi_Y^{(2)},\boldsymbol{\varepsilon},B}$ is same as the learning of $H(p||q, m, \Xi)$ in Eq. (38) and can also be implemented with help of Eq. (43). Moreover, it follows from Eq. (44), Eq. (45), and Eq. (46) that $\ln Q_{X_N|\Psi_Y^{(2)},\boldsymbol{\varepsilon},B} = Q_{X_N} + \ln q(Y^*|\eta^y, \Psi^y)$ with Q_{X_N} being able to be moved out of the integrals over $\Psi_Y^{(2)}, \boldsymbol{\varepsilon}, B$. Thus, we further have

$$\begin{aligned} &H(p||q) = Q_{X_N} + H^{(2)}(p||q, \kappa, \Xi^{(2)}) \\ &H^{(2)}(p||q, \kappa, \Xi^{(2)}) = \int p(\Psi_Y^{(2)}, \boldsymbol{\varepsilon}, B|X_N, A, Y, \Psi^{(1)}) \\ &\qquad \times \ln Q_{\Psi_Y^{(2)},\boldsymbol{\varepsilon},B}\mathrm{d}\Psi_Y^{(2)}\mathrm{d}\boldsymbol{\varepsilon}\mathrm{d}B, \\ &= \int p(\Psi_Y^{(2)}, \boldsymbol{\varepsilon}, B|X_N) \ln Q_{\Psi_Y^{(2)},\boldsymbol{\varepsilon},B}\mathrm{d}\Psi_Y^{(2)}\mathrm{d}\boldsymbol{\varepsilon}\mathrm{d}B, \\ &Q_{\Psi_Y^{(2)},\boldsymbol{\varepsilon},B} = \\ &\qquad q(Y^*|\eta^y, \Psi^y)q(\Psi_Y^{(2)}|\Xi)q(B|\boldsymbol{\eta}^b, \Psi^b)q(\boldsymbol{\varepsilon}|\boldsymbol{\eta}^\varepsilon, \Psi^\varepsilon), \end{aligned}$$

from which we see that $H^{(2)}(p||q, \kappa, \Xi^{(2)})$ is the one in Eq. (85) and is actually in the same expressions as the one in Eq. (38) for the first layer, and also that $p(\Psi_Y^{(2)}, \boldsymbol{\varepsilon}, B|X_N, A, Y, \Psi^{(1)})$ gets in effect in $H^{(2)}(p||q, \kappa, \Xi^{(2)})$ via $p(\Psi_Y^{(2)}, \boldsymbol{\varepsilon}, B|X_N)$, which justifies the simplifications of the Yang machine on the left side of Fig. 3.

4.2 Temporal FA and De-noise local FA

We continue the previous subsection with details about how the FA by Eq. (48) on the first layer is supported by the second layer via $\boldsymbol{\eta}^y = \eta^y(\boldsymbol{\varepsilon}, B)$ for temporal model. Specifically, we consider a special case of $\boldsymbol{\eta}^y = \eta^y(\boldsymbol{\varepsilon}, B)$ as follows:

$$\boldsymbol{\eta}^y = [\eta_1^y, \cdots, \eta_N^y], \quad \eta_t^y = \eta(\sum_{\tau=1}^{\kappa} B_\tau y_{t-\tau}). \tag{87}$$

That is, $\boldsymbol{\eta}^y$ is in a format similar to $\boldsymbol{\eta}^x$ in Eq. (22).

Particularly, when $\eta^{-1}(r) = r$ and $\kappa = 1$ we are led to the following formulation for temporal dependence among observations:

$$\begin{aligned} x_t &= Ay_t + e_t, \quad \mathcal{E}(y_t e_t^{\mathrm{T}}) = \mathbf{0}, \\ y_t &= By_{t-1} + \epsilon_t, \ \mathcal{E}(y_{t-1}\epsilon_t^{\mathrm{T}}) = \mathbf{0}, \\ e_t &\sim G(e|0, \Psi^x), \ \epsilon_t \sim G(\epsilon|0, \Psi^y), \end{aligned} \tag{88}$$

where Ψ^x and Ψ^y are usually diagonal. We are led to the box ⑫ in Fig. 1, i.e., Temporal FA (TFA) and state space model [68–75] for modeling temporal structure among data. The first equation is called observation equation and the second is called state equation.

In a standard SSM, not only we need to ensure that the parameters B to make the state equation stable, but also we need to know enough knowledge about A, B, Ψ^x and Ψ^y to infer y_t and all the remaining unknowns. This involves an important issue called identifiability, i.e., to ensure the SSM identifiable by imposing certain structures on either or both of A and B. It has been shown in Ref. [73–75] that adopting the basic nature of FA (i.e., dimensions of y_t are mutually uncorrelated or independent) will make the SSM by Eq. (88) become identifiable subject to an indeterminacy of any scaling. That is, it even removes the indeterminacy of any orthogonal matrix suffered by the classic FA as discussed after Eq. (18). Also, two FA parameterizations introduced around Eq. (27) become the following two identifiable structures:

$$\begin{aligned} &\text{Type A: } A \text{ is in general, while } B \text{ is diagonal} \\ &\qquad \text{and } \Psi^\epsilon = I; \\ &\text{Type B: } A^{\mathrm{T}}A \text{ is diagonal, } \Psi^\epsilon \text{ is diagonal,} \\ &\qquad \text{and for } B \text{ we have } B = \Phi D \Phi^{\mathrm{T}}, \\ &\text{where } D \text{ is diagonal, } \Phi \text{ is orthonormal.} \end{aligned} \tag{89}$$

The name TFA is used to refer the SSM by Eq. (88) under either of the above two constraints. Similar to the previous discussion on Eq. (15) in Sect. 2.1, the above two types are equivalent in term of the ML learning and Bayesian approach on $q(X|\Theta)$ in Eq. (23). Type A reduces the indeterminacy of Eq. (15) to an indeterminacy of any scaling due to the requirement that B, Ψ^ϵ are diagonal, while Type B transforms the condition of Type A via $C = \Phi$ in Eq. (15). With the help of the BYY harmony learning, Type B outperforms Type A on determining the unknown dimension of y_t via a diagonal $\Psi^\epsilon \neq I$, for a reason similar to the statement made between Eq. (27) and Eq. (29).

However, the constraint $A^{\mathrm{T}}A = I$ not only needs extra computing but also not good for sparse learning. Instead, recalling the statements made after Eq. (28) to the end of Sect. 2.1, the FA-D family by Eq. (27) with $A^{\mathrm{T}}A = I$ may be relaxed to the constraint that B, Ψ^ϵ are diagonal and A satisfies either of Eq. (31), Eq. (33) and Eq. (34), e.g., we consider

$$\text{Type C: } A \text{ by Eq. (31), and } B, \Psi^\epsilon \text{ are diagonal.} \tag{90}$$

That is, we consider the FA by Eq. (48) under the constraint $\mathrm{Tr}[\Psi_j^a] = 1$ together with

$$\begin{aligned} q(y_t|\eta_t^y, \Psi_t^y) &= G(y_t|By_{t-1}, \Psi_t^\epsilon), \\ q(y_{t-1}|\mathbf{0}, \Psi_{t-1}^y) &= G(y_{t-1}|\mathbf{0}, \Psi_{t-1}^y), \\ q(B|0, \Psi^b) &= \prod_j q(b_j|0, \Psi_j^b), \\ B = diag[b_1, \cdots, b_m], &\quad \text{subject to } |b_j| \leqslant 1, \\ each\ q(b_j|0, \Psi_j^b) &\quad \text{from a Beta-distribution,} \end{aligned} \tag{91}$$

where the constraint $|b_j| \leqslant 1$ comes from ensuring the system stability [73–75],and thus we may consider each $q(b_j|0, \Psi_j^b)$ with $b_j = 2(u_j - 0.5)$ and u_j from a Beta-distribution.

Putting the above setting into by Eq. (86) with Eq. (43) replaced by its detailed form by Eq. (62). The two loop learning procedure can be used for the BYY harmony learning directly. In the sequel, we omit the distribution $q(b_j|0, \Psi_j^b)$ and give the following detailed form of this double loop learning procedure:

$$\begin{aligned} &\underline{\textbf{Loop1 : updating of } \mathbf{x_t = Ay_t + e_t}} \\ \boldsymbol{\eta}_{new}^y &= By_{t-1}, \\ \Gamma_A &= \tfrac{1}{\psi_{old}^x} A_{old}^{\mathrm{T}} A_{old} + \Psi_{old}^{\epsilon\ -1}, \\ y_t &= \tfrac{1}{\psi_{old}^x} \Gamma_A^{-1} A_{old}^{\mathrm{T}} x_t + \boldsymbol{\eta}_{new}^y, \\ y_{t,\eta} &= y_t - \boldsymbol{\eta}_{new}^y, \ e_t^y = y_{t,\mu} - y_t, \\ A_{new} &= A_{old} + \\ &\qquad \gamma[x_t y_{t,\eta}^{\mathrm{T}} - A_{old}(\Psi_{old}^{\epsilon\ -1} + \psi_{old}^x \Psi_{old}^{a\ -1})], \\ e_t^x &= x_t - A_{new} y_t, \quad \delta A = A_\mu - A_{new}, \\ \delta\psi^x &= e_t^{y\ \mathrm{T}} A_{new}^{\mathrm{T}} A_{new} e_t^y + y_{t,\eta}^{\mathrm{T}} \delta A^{\mathrm{T}} \delta A y_{t,\eta}, \\ \psi_{new}^x &= (1-\gamma)\psi_{old}^x + \gamma(e_t^{xT} e_t^x + \delta\psi^x), \\ \Psi_{j,new}^{*a} &= (1-\gamma)\Psi_{j,old}^a + \\ &\quad \gamma\ \tfrac{\mathrm{diag}[a_{j,new} a_{j,new}^{\mathrm{T}} + (a_{j,\mu} - a_{j,new})(a_{j,\mu} - a_{j,new})^{\mathrm{T}}]}{a_{j,new}^{\mathrm{T}} a_{j,new} + (a_{j,\mu} - a_{j,new})^{\mathrm{T}}(a_{j,\mu} - a_{j,new})}. \\ &\quad where\ A = [a_1, \cdots, a_m], \\ &if\ \Psi_{j,new}^y \to 0, \text{discard the jth column of } A \\ &\quad \text{and discard dimension of } y_t. \end{aligned} \tag{92}$$

$$\begin{aligned} &\underline{\textbf{Loop2 : updating of } \mathbf{y_t = By_t + \epsilon_t}} \\ \Gamma_B &= B_{old} \Psi_{old}^{\epsilon\ -1} B_{old} + \Psi_{old}^{y\ -1}, \\ y^* &= \Gamma_B^{-1} B_{old}^{\mathrm{T}} \Psi_{old}^{\epsilon\ -1} y_t, \\ B_{new} &= B_{old} + \gamma diag[y^* y^{*\mathrm{T}} - B_{old} \Psi_{old}^y], \\ \Psi_{new}^y &= B_{new} \Psi_{old}^y B_{new} + \Psi_{old}^\epsilon, \\ \delta B_{new} &= B_\mu - B_{new}, \ \epsilon^{y*} = y_\mu^* - y^*, \\ \Delta\Psi_{old}^\epsilon &= B_{new} \epsilon^{y*} \epsilon^{y*\mathrm{T}} B_{new}^{\mathrm{T}} + \delta B_{new} y^* y^{*\mathrm{T}} \delta B_{new}^{\mathrm{T}}, \\ \Psi_{old}^\epsilon &= (1-\gamma)\Psi_{old}^\epsilon + \gamma \mathrm{diag}[y^* y^{*\mathrm{T}} + \Delta\Psi_{old}^\epsilon]. \end{aligned} \tag{93}$$

From a time series $\{x_t\}$, learning is made adaptively per sample. As a sample x_t comes, Loop 1 is implemented

equation by equation until its end, and next Loop 2 is implemented until its end, which composites an epoch. We iterate a number of epoches or until converged. Then, we get a new sample x_{t+1}, and so on and so forth.

Improving the previous TFA studies, the above learning has several new features. First, it makes TFA learning share the automatic model selection and sparse learning nature of the co-dim matrix pair based FA learning. Second, it provides a new algorithm for the BYY harmony learning on the second order TFA shown in Eq. (59) and Fig. 13 of Ref. [1]. Third, sparse learning may also be made on B with help of $q(b_j|0,\Psi_j^b)$ in a Beta-distribution. Still, similar to the role of $q(A|\boldsymbol{\eta}^a,\Psi^a)$ in Eq. (24) via $d_{A^*}+d_{Y^*}$, $q(B|0,\Psi^b)$ may also be taken in consideration for improving model selection criterion in Eq. (29).

Moreover, as elaborated in Fig. 8 of Ref. [67], a temporal dependence is described by a regression structure on $\boldsymbol{\eta}^y$ via the second equation in Eq. (88) or $\eta_t^y=\eta(\sum_{\tau=1}^{\kappa}B_\tau y_{t-\tau})$ in Eq. (87).

Alternatively, an equivalent temporal dependence may also be embedded in $q(y_t|\mathbf{0},\Psi^y)$ with Ψ^y given by an regression equation. E.g., $y_t=By_{t-1}+\boldsymbol{\varepsilon}_t$ is equivalently replaced by

$$\Psi_t^y=B^{\mathrm{T}}{\Psi_{t-1}^y}^{\mathrm{T}}B+\Psi^\epsilon,\ \mathcal{E}(y_t\boldsymbol{\varepsilon}_t^{\mathrm{T}})=\mathbf{0}. \tag{94}$$

Furthermore, we may also write the state equation in Eq. (88) in term of the entire data matrix Y as follows:

$$\begin{aligned}&\mathrm{vec}(Y)=B_b\mathrm{vec}(Y)+\boldsymbol{\varepsilon},\ \text{or}\\&\mathrm{vec}(Y)=B_L\boldsymbol{\varepsilon},\ B_L=(I-B_b)^{-1},\end{aligned} \tag{95}$$

where $\boldsymbol{\varepsilon}$ is stacked from $\epsilon_t, t=1,2,\cdots,N$, and B_L is a triangular with all the diagonal elements being 1, and B_b consists of $N\times m$ block rows with each block in a format $[\mathbf{0},_{\cdots},\mathbf{0},B_1,_{\cdots},B_\kappa,\mathbf{0},_{\cdots},\mathbf{0},]$, the first block row is $[\mathbf{0},B_1,_{\cdots},B_\kappa,\mathbf{0},_{\cdots},\mathbf{0},]$ with the first position being an $m\times m$ zero matrix $\mathbf{0}$ and the second position being B_1, while the next block row is one position circular shift toward right. Correspondingly, we have a Gaussian distribution

$$\begin{aligned}q(Y|\boldsymbol{\eta}^y,\Psi^y)&=q(Y|\mathbf{0},\Psi^y)\\&=G(\mathrm{vec}(Y)|\mathbf{0},B_L^{\mathrm{T}}(\Lambda\otimes I)B_L).\end{aligned} \tag{96}$$

That is, we are again led to a format similar to Eq. (66) that embeds temporal dependence into a constrained covariance matrix Ψ^y. Actually, it covers Eq. (66) since the Laplacian L is positively definite and we have $\mathrm{Tr}[YLY^{\mathrm{T}}]=\mathrm{vec}^{\mathrm{T}}(Y)(L^{-1}\otimes I)^{-1}\mathrm{vec}^{\mathrm{T}}(Y)=\mathrm{vec}^{\mathrm{T}}(Y)[B_L^{\mathrm{T}}(\Lambda\otimes I)B_L^{\mathrm{T}}]^{-1}\mathrm{vec}^{\mathrm{T}}(Y)$, with $B_L=B\otimes I, L^{-1}=B\Lambda B^{\mathrm{T}}$, where Λ is diagonal, and B is triangular with all the diagonal elements being 1. In other words, both a topological dependence and a temporal dependence can be considered via certain structure embedded in Ψ^y.

In the previous subsection, we also mentioned that the FA by Eq. (48) on the first layer may be supported by the second layer via $\boldsymbol{\eta}_j^a=\eta^a(\boldsymbol{\zeta},\Phi)$ for a denoise version of local FA. We consider each Gaussian $q(a_j|\eta_j^a,\Psi_j^a)=G(a_j|\eta_j^a,\Psi_j^a)$ is further described by a FA model $a_j=\varphi_j+\phi_j\zeta_t+\varepsilon_j$, with ε_j coming from $G(\varepsilon_j|0,\Psi_j^a)$, that is, we have $\eta_j^a=\varphi_j+\phi_j\zeta_t$ with a diagonal Ψ_j^a and $G(\zeta_t|0,\Lambda_j^\zeta)$ with a diagonal Λ_j^ζ. Accordingly we get

$$\begin{aligned}q(R|\theta)=q(A,Y,\zeta|\theta)=&\\\prod_{t,j}[G(a_j|\eta_j^a,\Psi_j^a)G(\zeta_t|0,\Lambda_j^\zeta)G(\phi_j|0,\Psi_j^\phi)\eta_j^y]^{y_{j,t}},&\\\eta_j^a=\varphi_j+\phi_j\zeta_t,&\end{aligned} \tag{97}$$

from which we are led to the Box ⑨ in Fig. 1, namely, a mixture of factor analysis, or local FA (including local PCA or local subspaces [1,32–36,48, 57–65]). The implementation can be handled in either of the following two choices:

- we let $q(a_j|\eta_j^a,\Psi_j^a)=G(a_j|\varphi_j,\Psi_j^a)$ in Eq. (55) replaced with $G(a_j|\varphi_j,\phi_j\Lambda_j^\zeta\phi_j^{\mathrm{T}}+\Psi_j^a)$ and similarly $G(x_t|\eta_j^a,\Psi^x+\Psi_j^a)$ in Eq. (55) becomes $G(a_j|\eta_j^a,\phi_j\Lambda_j^\zeta\phi_j^{\mathrm{T}}+\Psi^x+\Psi_j^a)$. In other words, the role Ψ^x applies to a standard local FA or a mixture of FA models [63–67].
- in order to make automatic model selection on the dimensions of ζ_t (i.e., hidden factors of local FA models), we can also implement the BYY harmony learning with $q(R|\theta)$ in Eq. (56) replaced by Eq. (97).

Last but not least, we may add on a common linear dimension reduction for tasks on a small size of high dimensional samples, with a common loading matrix C added to into Eqs. (55) and (97). That is,

$$\begin{aligned}&q(R^x|\Theta)=q(A,Y|\Theta)=\textstyle\prod_{t,j}[G(a_j|C\varphi_j,\Psi_j^a)\eta_j^y]^{y_{j,t}}\\&q(R^x|\Theta)=q(A,Y,\zeta|\Theta)=\\&\quad\textstyle\prod_{t,j}\,[G(a_j|\eta_j^a,\Psi_j^a)G(\zeta_t|0,\Lambda_j^\zeta)G(\phi_j|0,\Psi_j^\phi)\eta_j^y]^{y_{j,t}},\end{aligned} \tag{98}$$

where $\eta_j^a=C(\varphi_j+\phi_j\zeta_t)$ can be equivalently written as $\eta_j^a=C\varphi_j+\phi_j'\zeta_t$ since $\phi_j'=C\phi_j$. Also, we may consider the matrix C with the help of the singular value decomposition (SVD) $C=UDV^{\mathrm{T}}$.

4.3 General formulation, Hadamard matrix product, and semi-blind learning

Interestingly, $q(Y|\mathbf{0},\Psi^y)$ by Eq. (96) and $q(X|\eta(AY),\Psi^x)$ by Eqs. (10) and (14), jointly provide a general formulation that leads to the previous discussed examples, as each of four components $q(X|\boldsymbol{\eta}^x,\Psi^x)$, $q(Y|\mathbf{0},\Psi^y)$, $q(A|\boldsymbol{\eta}^a,\Psi^a)$, and $q(B|\boldsymbol{\eta}^b,\Psi^b)\}$ takes a specific structure. For example, with a Gaussian

$q(X|AY, \Psi^x)$ we have

$$X = AY + E, \quad \text{vec}[Y] = \mu^y + B_L\varepsilon. \tag{99}$$

That is, Y can be regarded as generated from ε via a linear mapping B_L by a sparse matrix with its nonzero parameters coming from nonzero parameters of B in a specific structure.

For an efficient implementation we usually return to the task motivated original models, such as Eq. (66) for manifold learning and Eq. (88) for TFA, in order to avoid a huge dimensional problem of B_L. Even so, the equation $\text{vec}(Y) = \mu^y + B_L\varepsilon$ not only provides conceptually a unified expression for variance dependence structures among Y, but also motivates further issues to investigate:

- Beyond a Gaussian $q(X|\eta^x, \Psi^x)$ and a Gaussian $q(Y|\eta^y, \Psi^y)$, Eq. (96) can be extended to an even general form with $q(X|\eta^x, \Psi^x)$, $q(Y|\eta^y, \Psi^y)$ given by Eq. (10) while η^x, η^y given by Eq. (22) with $\eta(r)$ taking one of typical choices shown in Table 1.
- The linear model by Eq. (4) is a common part of various data generating models, while the second part $\text{vec}(Y) = \mu^y + B_L\varepsilon$ should be considered based certain intrinsic properties of problems. Moreover, $B_L\varepsilon$ may also be extended to nonlinear mapping.
- This general guide is also helpful to making specific investigation. E.g., to check whether the Laplacian L from X by Eq. (66) preserves the topology underlying the one introduced by B_L, we may examine whether the cascaded mapping AB_L preserves this topology.

As discussed after Eq. (99), B_L is generally in a sparse structure. Recalling Eq. (70), we may encode a sparse matrix with help of binary variables that controls sparse degree and flexibly accommodates different matrix structures. Integrating Eq. (70) with the sate space model by Eq. (88), we further proceed to the following binary variable modulated state space model:

$$\begin{aligned} X &= (A \circ L_A)Y + E, \\ A \circ L_A &= [a_{ij}\ell^a_{ij}], \\ \text{vec}(Y) &= \mu^y + (B \circ L_B)\varepsilon, \\ B \circ L_B &= [b_{ij}\ell^b_{ij}]. \end{aligned} \tag{100}$$

As shown in Fig.4 and also illustrated by the Box ⑰ in Fig. 1, we get a general formulation for semi-blind learning. The first layer co-dim matrix pair $\eta^x = AY$ gets one second layer support $A \circ L_A$ in a Hadamard product for modulating A, and the other second layer support $\mu^y + B\varepsilon$ in an ordinary matrix product for modulating Y. Then, B in the second layer is further modulated by a third layer support $B \circ L_B$ in a Hadamard product. Following the two formations of semi-blind learning by Eq. (70) and Eq. (71), letting $B = 0$ in Eq. (100) leads to the third formation of semi-blind learning that can be regarded as an extension of NCA [76–79], as illustrated in the Box ⑯ in Fig. 1.

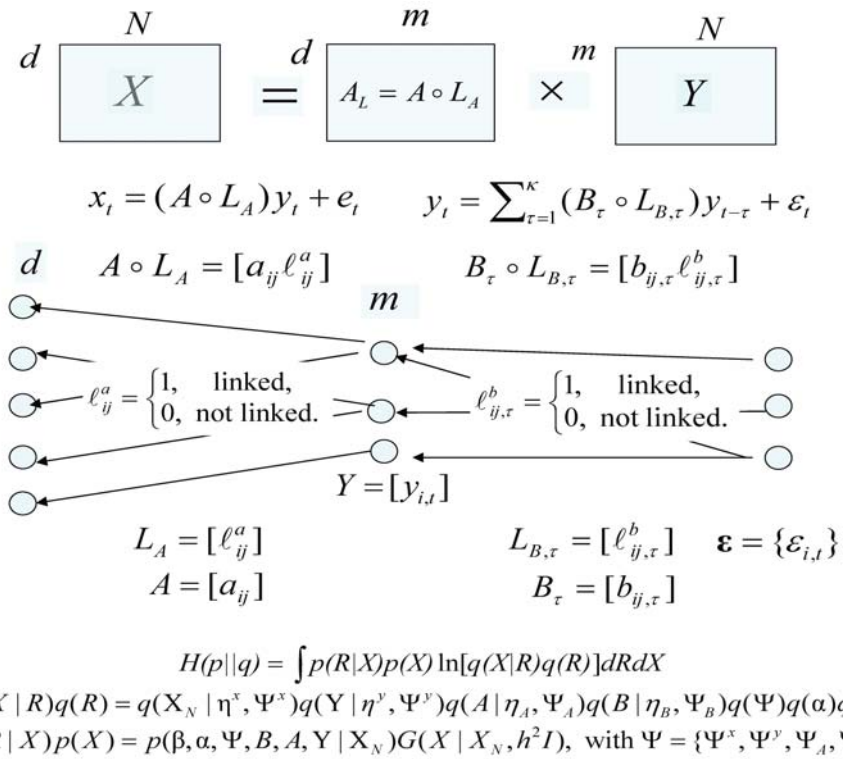

Fig. 4 General formulation, sparse learning, and semi-blind learning

All the binary random variables are mutually independent Bernoulli variables. That is, we have

$$q(L_A|\boldsymbol{\alpha}) = \prod_{i,j} \alpha_{i,j}^{\ell_{ij}^a}(1-\alpha_{i,j})^{1-\ell_{ij}^a},$$
$$q(L_B|\boldsymbol{\beta}) = \prod_{i,j} \beta_{i,j}^{\ell_{ij}^b}(1-\beta_{i,j})^{1-\ell_{ij}^b}. \quad (101)$$

In the special case that $\alpha_{ij} = 1$, we return back to Eq. (4). For $0 \leqslant \alpha_{ij} \leqslant 1$, its corresponding binary variable $\ell_{ij}^b = 0$ is switched on with a probability α_{ij}. Similarly, for $0 \leqslant \beta_{ij,\tau} \leqslant 1$, its corresponding binary variable ℓ_{ij}^b is switched on with a probability $\beta_{ij,\tau}$.

Specifically, we form the BYY system by Eq. (1) with

- $q(X|\eta(AY), \Psi^x)$ by Eqs. (10) and (14), and $\eta(r)$ takes one of typical choices shown in Table 1,
- $q(Y|\mathbf{0}, \Psi^y)$ by Eq. (96) or by Eq. (16) and Eq. (91),
- $q(A|\boldsymbol{\eta}^a, \Psi^a)$ by Eq. (24) and $q(B|\boldsymbol{\eta}^B, \Psi^B)$ by Eq. (91),
- $q(L_A|\boldsymbol{\alpha})$ and $q(L_B|\boldsymbol{\beta})$ by Eq. (101),

we implement the BYY harmony learning in a way similar to that introduced in the previous sections.

There are three major advantages for such a structure.

- It facilitates sparse learning that makes the matrix A flexibly take various stochastic topologies via driving $\alpha_{ij} \to 0$ if the corresponding element is extra.
- It provides a convenient way for incorporating partial structural information available in addition to knowing partially training samples of X and Y. If we know that it is high likely that some link does not exist or the corresponding a_{ij} takes zero or an ignorable small value, we force the corresponding $\alpha_{ij} \to 0$ or $\beta_{ij} \to 0$ or to be upper bounded by a small value.
- As those extra $\beta_{ij} \to 0$, the matrix B is pushed as sparse as possible by the least complexity nature of the BYY harmony learning.

5 Network biology applications

5.1 Network biology tasks

From the perspective of biological functions, we have transcriptional regulatory networks, protein interaction networks, metabolic networks, and signal transduction pathways [85]. From the perspective of network topology, these networks can be classified as undirectional networks (e.g., protein interaction networks) and directional networks (e.g., the other three biological networks). For the latter, we consider transcriptional regulatory and metabolic networks simply by bipartite networks. Studies on networks with a same topological type may be extended from one biological function to another, e.g., methods of learning bipartite transcriptional regulatory networks can be used to learn bipartite metabolic networks without difficulty [79]. Studies on building biological networks from data progress according to biological data available. Typically network biology tasks work on data that can be mainly expressed in the matrix forms. Shown in Fig. 5 are a number of data types encountered in various studies of networks biology.

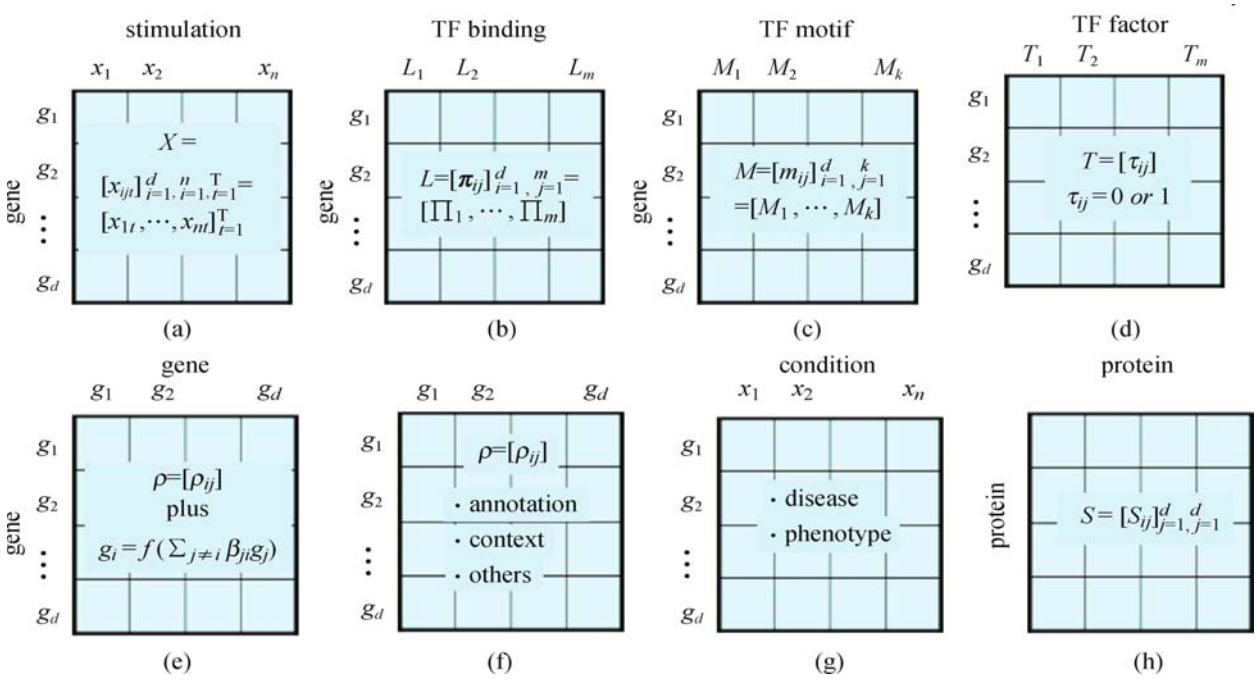

Fig. 5 Typical data matrices used in network biology tasks. (a) Gene expression; (b) CHIP-chip; (c) motifs in upstream region; (d) transcriptional network; (e) genetic association; (f) association network; (f) gene-X type; (h) PPI interaction

Though advances on high throughput technologies for DNA sequencing and genomics, gene expression data are still the most essential and widely available type for various studies. Expression data are given in a matrix in which each column is an observation on expressions of all the genes in one simulated experiment under one condition. Precisely, this type is called stead state data. In the past decade, efforts are made on time course gene expressions obtained from simulated experiments across a number of time points, which leads to matrices along the time coordinate, i.e., a cubic array in Fig. 5(a).

Various network biology tasks base on gene expression data, which can be roughly classified into three types. The most essential one consists of studies of transcriptome mechanisms, e.g., genetic association between genes as shown in Fig. 5(e) with each element describing the degree of association or relevance between a pair of genes, and transcriptional regulatory networks (TRN) as shown in Fig. 5(d) with the elements of "1" indicating TF-gene regulation. The other types of gene-gene association for various purposes are shown in Fig. 5(e), induced from or with the help of gene expression data. The third involves various studies and applications of gene-disease and genotype-phenotype relations as shown in Fig. 5(e).

However, gene expression alone is not enough for obtaining TRN shown in Fig. 5(d). Before transcription of a specific gene is initiated, its regulatory region is bound by one or more transcription factors (TFs), which recognize their targets through specific sequences, called binding motifs. Many efforts have been made in the last decade via binding motifs and CHIP-chip/ChIP-seq data for binding location [111], where CHIP stands for chromatin immunoprecipitation that acts as a protocol to separate the truncated DNA sequences that bind to specific protein from DNA suspensions.

Each CHIP-chip or CHIP-seq experiment involves one TF, and the results give information about the binding locations in the genome for this specific TF. As shown in Fig. 5(b), location data is a matrix with each element indicating a statistical significance level for binding between one TF and the promoter region of a gene, or an intensity level that quantifies the strength of binding. However, the observed binding information does not imply regulatory relationship between a TF and a gene that it is binding to, even when the results are highly significant [112]. Also, there are certain noises in CHIP-chip data collection. Moreover, CHIP-chip data only involves one TF per chip, while TF regulations involve combinatorial effects of multiple TFs that need to be inferred indirectly. Furthermore, TF binding is a dynamic process, and a TF can have different targets at different time points and/or under different conditions. If we draw conclusions on the regulatory targets for a TF based on one or a few CHIP-chip experiments, we will miss many true targets and also include many false targets for this TF under other conditions even if the experiments are done perfectly.

Shown in Fig. 5(c) is a matrix for TF binding motifs. If the binding motif for a TF is known, a gene whose regulatory region contains one or more instance of this motif is more likely to be the regulatory target of this TF. Each element of the matrix in Fig. 5(c) is proportional to the number of such motifs within this regulatory region. Motif data provide less direct evidence for the relation between TFs and genes than location data because motifs indicate merely potential binding sites which may not be bound by TFs and also many regulatory targets do not have known binding motifs in their regulatory regions. However, motif sequence does provide valuable information complementary to CHIP-chip and expression data, many efforts have been made in the past decade on integrating these data types for inferring TRN shown in Fig. 5(d).

Last but not least, shown in Fig. 5(h) is another data type that has a undirectional or symmetrical topology, describing protein-to-protein interactions (PPIs). Edges of PPI networks are determined by a measure technique on proteome-wide physical connections, indicating by a binary number 1 or 0 or a real degree between 0 and 1. Getting the PPIs mapping is regarded as a critical step towards unraveling the complex molecular relationships in living systems. In recent years, new large-scale technologies become available to measure proteome-wide physical connections between protein pairs. Similar to the way "genome" projects were a driving force of molecular biology 20 years ago, PPIs related studies has become one of the main scopes of current biological research, featured by the term interactome [113].

Though not entirely, a number of network biology tasks can be formulated as decomposition and integration of matrices in Fig. 5. One example is decomposing data matrix into a product of two matrices of lowered rank. Principally, it applies to each data matrix in Fig. 5, with different constraints added on two factorized matrices. Especially, it takes a major role in the studies of learning transcriptional networks as shown in Fig. 5(d), and also in the noncording RNA studies. Another example is featured by Eq. (9) and Eq. (72), which matches a unidirectional graph into another unidirectional graph via a permutation matrix [50,53,54], which can be used for comparison of two PPI networks [114]. Another example is decomposing a unidirectional graph into cliques or a PPI network into functional modules [115–118].

5.2 Learning transcriptional networks: past studies and future progresses

The earliest attempts [111] were made on gene expres-

sion data in Fig. 5(a) alone. The inferred networks consider general relationships among genes, and thus should be more appropriately named as gene regulatory networks. A fundamental limitation is the assumption that the expression levels of genes depend on the expression levels of the TFs regulating those genes. Expression data only measures the mRNA abundances, while it is the TF proteins that are directly involved in the regulation of genes. The mRNA levels of the TFs may not be highly correlated with those of the genes they regulate. In the past decade, progresses have been obtained on studying transcriptional regulation networks (TRN) with the help of location data shown in Fig. 4(b). The task is featured by finding the bipartite networks expressed by the binary matrix in Fig. 5(d). The effort starts from getting bipartite networks for TRN, which may also be backtracked by three streams.

One stream is featured by bi-clustering that groups genes and their corresponding TFs into functional modules. Genes in the same module exhibit similar expression profiles and tend to have similar biological functions. Also, these genes should share similar regulation patterns and thus are more likely to be under the control of the similar TFs. This bi-clustering bases both the similarities of expression profiles and the activities of TFs. In the absence of information for TF activities, Segal et al. [119] infers key TFs of each module by correlating the gene expression levels with expression patterns of genes in the module across a large number of experimental conditions, and then the obtained TFs activities and gene expression data are jointly used to accurately assign each gene to its corresponding module. Inferring key TFs and making clustering are made iteratively. However, this procedure is limited by the possibly poor correspondence between gene expression levels and TF activities. Reiss et al. [120] categorized genes into co-regulated groups across a subset of all observed experimental conditions with the help of bi-clustering (genes and conditions) instead of a standard clustering that participates genes into co-expressed groups. Co-regulated genes are often functionally associated and easy to be incorporated with a priori such as common cis-regulatory motifs.

Bar-Joseph et al. [112] proposed a procedure called GRAM that uses the CHIP-chip location data in Fig. 5(b) also in a two step iteration. First, a stringent criterion infers the binding targets only for those TF-gene pairs that have high statistical significance. Second, gene expression are used to define a core expression profile for a set of genes sharing a common set of TFs as their regulators. After the core expression profiles are defined, other genes are included in a transcription module if their expression profiles are similar to the core profiles. Sharing a similar structure, the ReModiscovery algorithm by Lemmens et al. [121] uses both the location data in Fig. 5(b) and the motif data in Fig. 5(c) to jointly detect modules of tightly co-expressed genes that share common subsets of TFs and motifs that exceed thresholds. Recently, LeTICE by Youn et al. [122] further extends these studies into a probabilistic model for a binary binding matrix for integrating the expression and location data in Fig. 5(a) and 5(b). Without requiring thresholds, LeTICE generates all gene modules simultaneously using the entire set of TFs, instead of step-wisely getting gene modules via subsets of TFs.

The second stream considers the matrix decomposition by Eq. (4) as bipartite networks, which is justified from the fact that the equilibrium state of a nonlinear kinetics model leads to that the log-ratio of gene expression levels between two conditions is related to the additive effects from a set of TFs through the log-ratio of the TF activities to the regulatory strength. Again, early studies began with only gene expression data as X in Eq. (4), one is the singular value decomposition (SVD) [123–127], another is the independent component analysis (ICA) [128,129] on the assumption of $E = 0$. However, the interpretability of SVD and ICA solutions are a concern. This stems from the fact that orthogonality and statistical independence lack physical meaning. Also, both SVD and ICA assume that the bipartite network topology is fully connected, and each source signal contributes to every output. This is an inappropriate assumption for transcriptional regulation where it is accepted that transcription networks are generally sparse.

Similarly, this stream also moved to considering the sequence data and CHIP-chip location data in Fig. 5(b) and 5(c). Liao et al. [76] considered a prior knowledge on the connectivity between TFs and genes but does not need knowing regulatory strengths or TF activities, from which each element of A is set to be zero if there is no connectivity that corresponds to this element. With this constraint, the decomposition of Eq. (4) is made via minimizing the Frobenius norm of E to determine those unknown regulatory strengths, under the name of network component analysis (NCA). This idea is followed and further extended, by Boulesteix and Strimmer [77] with the help of the partial least squares to reduce the dimensionality of the space spanned by the TFs, by Brynildsen et al. [78] with the help of the Gibs sampler to screen for genes with consistent expression and CHIP-chip derived connectivity data, and also by Brynildsen et al. [79] with the assumption relaxed to be nothing about the nature of the source signals beyond linear independency. Being different from directly using the binding intensities b_{ij} from CHIP-chip data, i.e., $a_{ij} = b_{ij}$, Sun et al. [130] assumed that $a_{ij} = b_{ij}c_{ij}$, where c_{ij} is the unknown but desired regulatory relationship between TF j and gene i. There are also several other efforts on learning TRN by integrating gene expression, CHIP-chip, and sequence information [131–133]. Despite different motivations, these methods all

share the same general modeling form. Pournara and Wernisch [134] compared some methods in this context.

The third stream consists of state space model (SSM) based studies for time course gene expressions across a number of time points. Gene expression in a general situation as shown in Fig. 5(a) is a cubic array that varies along three coordinates. It is known that network functions are determined not only by their static structures but also by their dynamic structural rearrangements [135]. In recent years, SSM has been adopted for time course gene expressions. Most of existing studies [136–145] treat data $x_{j,t}$ from $j = 1, 2, \cdots, n$ as multiple observation series from one same underlying SSM. In Ref. [146], apparently it considers multiple conditions with one subscript corresponding explicitly to each different condition, actually it still considers a same set of parameters for their SSM across all the conditions and thus bring no difference. In these situations, $x_{j,t}, j = 1, 2, \cdots, n$ can be equivalently regarded as segments of a same stochastic process. For notation clarity, we ignore the subscript $j = 1, 2, \cdots, n$ and simply consider a time series $X = \{x_t, t = 1, 2, \cdots, T\}$ with each vector x_t that consists of expressions of all the genes.

Two early studies [136,137] proposed to use a so-called input driven SSM, i.e., adding an extra term Cx_{t-1} to both the state equation and observation equation in a standard SSM by Eq. (88). Later on, most of efforts [138–146] turned back to a standard SSM because it has been shown that the additional term Cx_{t-1} basically has no help and may even cause instability [143]. Likely, such an input driven term has not been included in a standard SSM during the past extensive studies [68] just because of an awareness of its useless. Generally, one major SSM problem is the lack of identifiability, for which extra structural constraints have been imposed to solve the problem. In addition to the usual assumption on e_t and ε_t, extra constraints are imposed also on either or both of A and B. Mostly, A is imposed to be an identity matrix $\boldsymbol{I}$ [144] or its permutation [138,141] or a diagonal matrix [143]. These over-simplified SSM studies actually work as a filter to removing noises in gene expression data.

Also, this stream recently turns to integrating CHIP-chip and sequence data into the SSM modeling. Sanguinetti et al. [138] adopted the sparsity constraint on A in a same way as made in NCA [76], which can be regarded as an extension of NCA into SSM. Similarly but with more assumptions, constrained SSMs are suggested in [140] and [144], also based on a known network structure. If a pair of genes is represented by two states that is known to have no interaction, the corresponding entry in the matrix B are all set to be zero. Similarly, if an input has no influence on a gene that is represented as a state, the corresponding entry in A should be zero.

The above overview sketched an outline of frontier tasks, which motivates to apply the general formulation shown in Fig. 4 for modeling TRN with the help of the BYY harmony learning, with the following advantages and improvements:

1) As addressed above, imposing that $A = I$ [145] or its permutation [138,141] or that A is diagonal [144] actually makes the role of SSM degenerated to a filter for observation noises. This limitation can be removed by considering the TFA by Eq. (88), which has been shown to be identifiable with the help of either of three structural constraints given by Eq. (89) and Eq. (90).

2) Sanguinetti et al.[138] handled the above limitation by adopting the sparsity on A as used by NCA [76]. But it is unreliable to simply decide whether or not remove an edge based on the information from sequence data and CHIP-chip location data, as previously introduced. The formulation in Fig. 4 differs in not only considering either of the above two types of structural constraints, but also providing a flexible venue to accommodate stochastic topologies to adopt an appropriate one, controlled by the probability α_{ij} obtained from combining the data in Fig. 5(b) and 5(c), e.g.,

$$\alpha_{ij} = (1 - \gamma)\pi_{ij} + \gamma/(1 + \mathrm{e}^{-m_{ij}}), \quad 0 \leqslant \gamma \leqslant 1,$$

where γ controls the proportions of two types of data, which may be pre-specified or obtained via learning.

3) Being different from the existing SSM studies, the probabilistic sparsity is also imposed on both the observation equation and the state equation by the formulation in Fig. 4. With the help of learning each probability $0 \leqslant \beta_{ij} \leqslant 1$, the sparsity of the matrix B is determined by the least complexity nature of the BYY harmony learning, in order to learn both the underlying dynamics of each TF and the relationship across different TFs.

4) Though the recently proposed LeTICE [122] improves the previous studies by using a probabilistic sparsity on the matrix A also by the Bernoulli binary variable, it actually considers merely the first layer in Fig. 4, without considering temporal dependence by the state equation. That is, it not a SSM modeling, but just a bipartite TRN network that uses this probabilistic sparsity to improve the one used in NCA [76]. In addition, the formulation in Fig. 4 also differs in not only implementing the BYY harmony learning but also considering both CHIP-chip and sequence information in Fig. 5(b) and 5(c), while LeTICE only considers the CHIP-chip data in Fig. 5(b).

5) With the help of automatic model selection, the lower rank of the matrix A or equivalently an appropriate number of TF factors are determined during the BYY harmony learning, instead of pre-specifying this number or obtaining it by a conventional two stage model selection via a criterion such as AIC,BIC/MDL.

Readers are referred to Sect. 2 of Ref. [1] to see a number of advantages of this automatic model selection.

6) Most of the above mentioned advantages are applicable to those simplified studies that the matrix A is assumed to be binary, e.g., for finding TF modules by clustering analysis with a pre-specified number of clusters. In this case, the formulation in Fig. 4 will degenerate to nonnegative matrix factorization (NMF), and particularly to the binary matrix factorization (BMF) bi-clustering [28] for the cases of Eq. (64).

7) There have been some efforts that treating TRN modeling as a regression model, e.g., considering a linear regression $X = AY + E$ with a known pair X and Y. However, there are at least two places to be improved. First, a known pair X and Y actually provides a unreliable relation due to noises and also pseudo values taken by some elements of Y, for which we may consider the Semi-blind learning FA by Eq. (69) and Semi-blind learning BFA by Eq. (70). Second, AY is extended to be given by Eq. (22) to cover that $x_{i,t}$ takes either of binary, real, and nonnegative types of values. E.g., we further extend $X = AY_H + E$ in Eq. (70) into $X = \eta(AY_H) + E$ together with the third choice in Tab. 1 for performing a generalized Cox type regression. Actually, there are other types of regression tasks in bioinformatics studies, for which such extended regression models may also be considered.

5.3 Analyzing PPI networks: network alignment and network integration

As gene expression data in Fig. 5(a) takes essential roles in those studies of transcriptome mechanisms, the PPI data in Fig. 5(h) is also essentially important in the studies of interactome mechanisms [113]. Among various efforts made on the PPI data, many studies are made on functional modular analysis or module detection, i.e., decomposing a unidirectional graph into cliques or clusters [115–118]. Together with functional annotations and genomic data, extracted modules facilitate prediction of protein functions, and discovery of novel biomarkers, as well as identification of novel drug targets. The existing methods can be classified into three categories [117]. One utilizes direct connections to incrementally enlarge each module through local detection, mainly by heuristic approaches. The second performs graph clustering algorithms that consider extracted clusters/modules jointly under one global guide. Recently, a BYY harmony learning based bi-clustering algorithm has also been developed and shown favorable performances in comparison with several well known clustering algorithms [28]. The third considers the functioning of one PPI networks as a dynamic stable system.

Another topic on PPI data is finding the similarities and differences in two networks or called network alignment, which can directly be applied for analyzing signal pathways, detecting conserved regions, discovering new biological functions or understanding the evolution of protein interactions. As addressed after Eq. (72), an exact match between two graphs involves combining the scores of node-to-node matching into a global matching score and also searching all the combining possibilities, which is a well known NP-hard problem. To tackle this difficulty, one way is an approximate process of forming the global matching score from the local scores such that the global process of searching all the combinations become tractable or even analytically solvable [114]. The other way is seeking some heuristic searching that approximately neglects a part of all the possible combinations. Both ways lead to an inexact graph matching, as widely studied in the literature of pattern recognition in past decades [50–54]. Following the steps of Ref. [114], several efforts have recently been made along the first way. Here we explore the second way with the help of the techniques introduced in Sect. 3.3.

Given two PPI networks denoted by S_X and S_Y, we match the PPI networks by Eq. (72) or Eq. (73), with a permutation matrix relaxed to a doubly stochastic matrix as justified in Ref. [20]. Then, we tackle the problem under the general guideline by the Lagrange-enforcing algorithms proposed in Refs. [105,106], with a guaranteed convergence on a feasible solution that satisfies constraints. Moreover, we may further handle the problem by enforcing a doubly stochastic matrix into an orthostochastic matrix and implementing the optimization by Stiefel gradient flow [107,108]. Moreover, we consider the BYY harmony learning by Eq. (78) with $q(A|\boldsymbol{\eta}^a, \Psi^a)$ by Eq. (24) and $q(a_j|\eta_j^a, \Psi^a)$ from a multivariate Gaussian or Laplace distribution.

In addition to the matching costs by Eq. (72), Eq. (73) and Eq. (75), we may have another one that replaces each local score between two edges by a sum of local scores between not only the two corresponding edges but also a subset of neighbor edges. That is, we consider

$$\min_{A\in\Pi} J(S_X, A), \quad J(S_X, A) = \sum_{i,j} \gamma_{i,j} H_{i,j}, \qquad (102)$$

$$H_{i,j} = \begin{cases} \sum_{k\in N_i, \ell\in N_j} (s_{i,k}^x - \psi_{\ell,j}^x)^2, & choice(a) \\ -\sum_{k\in N_i, \ell\in N_j} s_{i,k}^x \psi_{\ell,j}^x, & choice(b) \end{cases}$$

where N_i denotes a subset of neighbor edges of the i-th node in S_X, while N_j denotes a subset of neighbor edges of the j-th node in S_Y. Sharing a sprit similar to [114], it puts more emphases on the matching between the densely connected parts.

Graph matching or network alignment also takes an essential role in network integration, i.e., integrating several types of data in Fig. 5, an intensively addressed important topic in the network biology literature

[85,111,117,147]. Efforts range from simple intersection analysis to sophisticated probability-based scoring systems, where Bayesian probabilities are derived based on the strength of evidence associated with an edge, e.g., referred to a summary in Figure 3 of Ref. [85]. However, most of these studies proceed with a given correspondence between different data matrices, integration is made between the corresponding elements without considering other possible correspondences. For those problems that need to take different permutations in consideration, we need to handle network alignment before making network integration.

Moreover, matching two PPI networks S_X and S_Y may also be directly used for an integration purpose. After networks matched, two corresponding edges with a high matching score supports each other, while two corresponding edges with a high mismatching degree may be removed. Furthermore, we may even consider matching two PPI networks S_X and S_Y with each having some unknown edges, with the help of defining an appropriate score for a correspondence between one known edge and one unknown edge, and also for a correspondence between two unknown edges. The unknown edges may be recovered or discarded according to the resulted graph matching.

Before network integration, another helpful process is making each network data contain as least redundancy as possible. One technique is to detect whether an edge describes a direct link or a duplicated indirect link. For data matrices for association relationship, e.g., ones in Fig. 5(e) and 5(h), one way to handle this problem is to examine association between two nodes in a presence of one or more other nodes. If two nodes i, j are linked to a third node w with the correlation coefficients ρ_{iw} and ρ_{jw}, it follows from Eqs. (14) and (15) in Refs. [148,149] that we can remove the link i, j if its correlation coefficient ρ_{ij} fails to satisfy Theorems 2 and 3 in Refs. [148,149], i.e.,

$$\rho_{ij} > \rho_{iw}\rho_{wj}. \tag{103}$$

Otherwise we may either choose to keep the link i, j, or let three nodes to be linked to a newly added node and then remove all the original links among three nodes.

There could be various tricks for integration of already matched or aligned matrices, e.g., taking an average, choosing the best, picking the most reliable one, etc. One key issue is to calibrate the values measured in different environments and with different uncertainties, which is actually a common topic of those studies on information fusion and classifier combination. One typical direction is turning different values into the probabilities under an assumption of mutual independence and then making a combination by the naïve Bayesian rule [85] or called the product rule. Readers are referred to the details and other combining strategies [109,110].

Another challenge is that data matrices to be integrated could be different types, as shown in Fig. 5. Here, we propose a direction to tackle this challenge. We consider two typical groups of data types, namely gene-gene and gene-X (disease, condition, etc) or generally $X - X$ type and $Y - X$ type. In a rough handling, we may treat the $Y - X$ type in Eq. (4) and the $X - X$ type as $S_X = XX^{\mathrm{T}}$ in Eq. (9), and make integration with the help of the BYY harmony learning by Eq. (2) together with Eqs. (81), (82), (83) and (84).

6 Concluding remarks

Further insights on the Bayesian Ying-Yang (BYY) harmony learning have been provided from a co-dimensional matrix-pairing perspective. A BYY system is featured with a hierarchy of several co-dim matrix pairs, and best harmony learning is further improved via exploring the co-dim matrix pairing nature, with refined model selection criteria and a modified mechanism that coordinates automatic model selection and sparse learning. Particularly, typical learning tasks based on $X = AY + E$ and its post-linear extensions have been re-examined. Not only learning algorithms for FA, BFA, BMF, and NFA have been updated from this new perspective, but also the following new advances have been introduced:

- A new parametrization that embeds a de-noise nature to Gaussian mixture and local FA;
- An alternative formulation of graph Laplacian based linear manifold learning;
- Algorithms for attributed graph matching, and co-decomposition of data and covariance;
- A co-dim matrix pair based generalization of temporal FA and state space model;
- A semi-supervised formation for regression analysis and a semi-blind learning formation for temporal FA and state space model.

Moreover, these advances provide with new tools for network biology studies, including learning transcriptional regulatory, Protein-Protein Interaction network alignment, and network integration.

Acknowledgements This work was supported by the General Research Fund from Research Grant Council of Hong Kong (Project No. CUHK4180/10E), and the National Basic Research Program of China (973 Program) (No. 2009CB825404).

References

1. Xu L. Bayesian Ying-Yang system, best harmony learning, and five action circling. *A special issue on Emerging Themes on Information Theory and Bayesian Approach*, Frontiers of Electrical and Electronic Engineering in China,

2010, 5(3): 281–328
2. Anderson T W, Rubin H. Statistical inference in factor analysis. In: Proceedings of the Third Berkeley Symposium on Mathematical Statistics and Probability. 1956, 5: 111–150
3. Rubi D, Thayer D. EM algorithm for ML factor analysis. Psychometrika, 1976, 57: 69–76
4. Bozdogan H, Ramirez D E. FACAIC: model selection algorithm for the orthogonal factor model using AIC and FACAIC. Psychometrika, 1988, 53(3): 407–415
5. Belouchrani A, Cardoso J. Maximum likelihood source separation by the expectation maximization technique: deterministic and stochastic implementation. In: Proceedings of NOLTA95. 1995, 49–53
6. Xu L. Bayesian Kullback Ying-Yang dependence reduction theory. Neurocomputing, 1998, 22(1–3): 81–111
7. Xu L. BYY learning, regularized implementation, and model selection on modular networks with one hidden layer of binary units, Neurocomputing, 2003, 51:277–301
8. Xu L. Advances on BYY harmony learning: Information theoretic perspective, generalized projection geometry, and independent factor auto-determination. IEEE Transactions on Neural Networks, 2004, 15(4): 885-902
9. Xu L. Independent component analysis and extensions with noise and time: a Bayesian Ying-Yang learning perspective. Neural Information Processing-Letters and Reviews, 2003, 1(1): 1–52
10. Moulines E, Cardoso J, Gassiat E. Maximum likelihood for blind separation and deconvolution of noisy signals using mixture models. In: Proc. ICASSP97. 1997, 3617–3620
11. Attias H. Independent factor analysis. Neural Computation, 1999, 11(4): 803–851
12. Liu Z Y, Chiu K C, Xu L. Investigations on non-Gaussian factor analysis. IEEE Signal Processing Letters, 2004, 11(7): 597–600
13. Xu L. Independent subspaces. In: Ramón J, Dopico R, Dorado J, Pazos A, eds. Encyclopedia of Artificial Intelligence, Hershey (PA): IGI Global. 2008, 903–912
14. Saund E. A multiple cause mixture model for unsupervised learning. Neural Computation, 1995, 7(1): 51–71
15. Zhang B L, Xu L, Fu M Y. Learning multiple causes by competition enhanced least mean square error reconstruction. International Journal of Neural Systems, 1996, 7(3): 223–236
16. Reckase M D. The past and future of multidimensional item response theory. Applied Psychological Measurement, 1997, 21(1): 25–36
17. Moustaki I, Knott M. Generalized latent trait models. Psychometrika, 2000, 65(3): 391–411
18. Bartholomew D J, Knott M. Latent variable models and factor analysis, Kendalls, Library of Statistics, Vol. 7. New York: Oxford University Press, 1999
19. Paatero P, Tapper U. Positive matrix factorization: a non-negative factor model with optimal utilization of error estimates of data values. Environmetrics, 1994, 5(2): 111–126
20. Lee D D, Seung H S. Learning the parts of objects by non-negative matrix factorization. Nature, 1999, 401(6755): 788–791
21. Lee D D, Seung H S. Algorithms for non-negative matrix factorization. Adv. Neural Inf. Process, 2001, 13: 556–562
22. Kim H, Park H. Nonnegative matrix factorization based on alternating nonnegativity constrained least squares and active set method. SIAM Journal on Matrix Analysis and Applications, 2008, 30(2): 713–730
23. Kim H, Park H. Sparse non-negative matrix factorizations via alternating non-negativity-constrained least squares for microarray data analysis. Bioinformatics (Oxford, England), 2007, 23(12): 1495–1502
24. Chen Y, Rege M, Dong M, Hua J. Non-negative matrix factorization for semi-supervised data clustering. Knowledge and Information Systems, 2008, 17(3): 355–379
25. Ho N, Vandooren P. Non-negative matrix factorization with fixed row and column sums. Linear Algebra and Its Applications, 2008, 429(5-6): 1020–1025
26. Cemgil A T. Bayesian Inference for Nonnegative Matrix Factorisation Models, Computational Intelligence and Neuroscience, 2009
27. Yang Z, Zhu Z, Oja E. Automatic rank determination in projective nonnegative matrix factorization. Lecture Notes in Computer Science: Latent Variable Analysis and Signal Separation, 2010, (6365): 514–521
28. Tu S, Chen R, Xu L. A binary matrix factorization algorithm for protein complex prediction. In: Proceedings of the BIBM 2010 International Workshop on Computational Proteomics, Hong Kong, December 18–21, 2010
29. Redner R A, Walker H F. Mixture densities, maximum likelihood, and the EM algorithm. SIAM Review, 1984, 26(2): 195–239
30. Xu L, Jordan M I. On convergence properties of the EM algorithm for Gaussian mixtures. Neural Computation, 1996, 8(1): 129–151
31. McLachlan G J, Geoffrey J. The EM Algorithms and Extensions. Wiley, 1997
32. Xu L. Multisets modeling learning: a unified theory for supervised and unsupervised learning. In: Proceedings of IEEE ICNN94. 1994, I: 315–320
33. Xu L. A unified learning framework: multisets modeling learning. In: Proceedings of WCNN95. 1995, 1: 35–42
34. Xu L. Rival penalized competitive learning, finite mixture, and multisets clustering. In: Proceedings of IEEE-INNS IJCNN98, Anchorage, Alaska, vol. II. 1998, 2525–2530
35. Xu L. BYY harmony learning, structural RPCL, and topological self-organizing on unsupervised and supervised mixture models. Neural Networks, 2002, (8-9): 1125–1151
36. Xu L. Data smoothing regularization, multi-sets-learning, and problem solving strategies. Neural Networks, 2003, 16(5-6): 817–825
37. Belkin M, Niyogi P. Laplacian eigenmaps for dimensionality reduction and data representation. Neural Computation, 2003, 15(6): 1373–1396
38. He X, Niyogi P. Locality Preserving Projections. In: Advances in Neural Information Processing Systems 16. Cambridge, MA: MIT Press, 2003, 152-160
39. Wallace C S, Dowe D R. Minimum message length and Kolmogorov complexity. Computer Journal, 1999, 42(4): 270–283

40. Figueiredo M A F, Jain A K. Unsupervised learning of finite mixture models. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2002, 24(3): 381–396
41. Williams P M. Bayesian regularization and pruning using a Laplace prior. Neural Computation, 1995, 7(1): 117–143
42. Tibshirani R. Regression shrinkage and selection via the lasso. J. Royal. Statist. Soc B., 1996, 58(1): 267-288
43. Hansen L K, Goutte C. Regularization with a pruning prior. Neural Networks, 1997, 10(6): 1053–1059
44. Schwarz G. Estimating the dimension of a model. Annals of Statistics, 1978, 6(2): 461–464
45. Rissanen J. Modeling by shortest data description. Automatica, 1978, 14: 465-471
46. Rissanen J. Basics of estimation. Frontiers of Electrical and Electronic Engineering in China, 2010, 5(3): 274–280
47. Corduneanu A, Bishop C M. Variational Bayesian model selection for mixture distributions. In: Jaakkola T, Richardson T, eds. Artificial Intelligence and Statistics, Morgan Kaufmann. 2001, 27–34
48. Choudrey R A, Roberts S J. Variational mixture of Bayesian independent component analyzers. Neural Computation, 2003, 15(1): 213–252
49. McGrory C A, Titterington D M. Variational approximations in Bayesian model selection for finite mixture distributions. Computational Statistics & Data Analysis, 2007, 51(11): 5352-5367
50. Umeyama S. An eigendecomposition approach to weighted graph matching problems. IEEE Transactions on Pattern Analysis and Machine Intelligence, 1988, 10(5): 695–703
51. Xu L, Oja E. Improved Simulated Annealing, Boltzmann Machine and Attributed Graph Matching. In: Goos G, Hartmanis J, eds. Lecture Notes in Computer Sciences, Springer-Verlag, 1989, 412: 151–160
52. Conte D, Foggiay P, Sansoney C, Vento M. Thirty years of Graph Matching in Pattern Recognition. International Journal of Pattern Recognition and Artificial Intelligence, 2004, 18(3): 265–298
53. Xu L, Klasa S. A PCA like rule for pattern classification based on attributed graph. In: Proceedings of 1993 International Joint Conference on Neural Networks (IJCNN93), Nagoya. 1993, 1281–1284
54. Xu L, King I. A PCA approach for fast retrieval of structural patterns in attributed graphs. IEEE Transactions on Systems, Man and Cybernetics, Part B, 2001, 31(5): 812–817
55. Li H B, Stoica P, Li J. Computationally efficient maximum likelihood estimation of structured covariance matrices. IEEE Transactions on Signal Processing, 1999, 47(5): 1314–1323
56. Burg J, Luenberger D, Wenger D. Estimation of structured covariance matrices. Proceedings of the IEEE, 1982, 70(9): 963–974
57. Xu L. Beyond PCA learning: from linear to nonlinear and from global representation to local representation. In: Proceedings of ICONIP94. 1994, 2: 943–949
58. Xu L. Vector quantization by local and hierarchical LMSER. In: Proceedings of 1995 Intl Conf.on Artificial Neural Networks (ICANN95), Paris. 1995, II: 575–579
59. Hinton G E, Dayan P, Revow M. Modeling the manifolds of images of handwritten digits. IEEE Transactions on Neural Networks, 1997, 8(1): 65–74
60. Liu Z Y, Chiu K C, Xu L. Strip line detection and thinning by RPCL-based local PCA. Pattern Recognition Letters, 2003, 24(14): 2335–2344
61. Liu Z Y, Xu L. Topological local principal component analysis. Neurocomputing, 2003, 55(3–4): 739–745
62. Tipping M E, Bishop C M. Mixtures of probabilistic principal component analyzers. Neural Computation, 1999, 11(2): 443–482
63. Salah A A, Alpaydin E. Incremental mixtures of factor analyzers. In: Proceedings of the 17th International Conference on Pattern Recognition. Cambridge: IEEE Press, 2004, 1: 276–279
64. Utsugi A, Kumagai T. Bayesian analysis of mixtures of factor analyzers. Neural Computation, 2001, 13(5): 993–1002
65. Ghahramani Z, Beal M. Variational inference for Bayesian mixtures of factor analysers, Advances in neural information processing systems 12. Cambridge, MA: MIT Press, 2000, 449–455
66. Xu L, Bayesian Ying Yang System, Best Harmony Learning, and Gaussian Manifold Based Family. In: Zurada et al, eds. Computational Intelligence: Research Frontiers (WCCI2008 Plenary/Invited Lectures), LNCS5050, 2008, 48–78
67. Xu L. Learning algorithms for RBF functions and subspace based functions. In: Olivas E S, et al, eds. Handbook of Research on Machine Learning, Applications and Trends: Algorithms, Methods and Techniques, Hershey (PA): IGI Global. 2009, 60–94
68. Brown R G, Hwang P Y C. Introduction to random signals and applied Kalman filtering. John Wiley & Sons, Inc., 1997
69. Xu L. Bayesian Ying Yang System and Theory as a Unified Statistical Learning Approach (II): From Unsupervised Learning to Supervised Learning and Temporal Modeling. In: Wong K M, Yeung D Y, King I, et al, eds. Theoretical Aspects of Neural Computation: A Multidisciplinary Perspective. Berlin: Springer-Verlag, 1997, 25–60
70. Xu L. Temporal BYY learning and its applications to extended Kalman filtering, hidden Markov model, and sensormotor integration. In: Proceedings of IEEE-INNS 1999 Intl J. Conf on Neural Networks, Washington. 1999, 2: 949–954
71. Xu L. Bayesian Ying-Yang system and theory as a unified statistical learning approach:(V) temporal modeling for temporal perception and control. In: Proceedings of ICONIP98, Kitakyushu. 1998, 2: 877–884
72. Ghahramani Z, Hinton G E. Variational learning for switching state-space models. Neural Computation, 2000, 12(4): 831–864
73. Xu L. Temporal BYY learning for state space approach, hidden Markov model and blind source separation. IEEE Transactions on Signal Processing, 2000, 48(7): 2132–2144
74. Xu L. BYY harmony learning, independent state space, and generalized APT financial analyses. IEEE Transactions on Neural Networks, 2001, 12(4): 822–849
75. Xu L. Temporal BYY encoding, Markovian state spaces,

and space dimension determination. IEEE Transactions on Neural Networks, 2004, 15(5): 1276–1295

76. Liao J C, Boscolo R, Yang Y L, Tran L M, Sabatti C, Roychowdhury V P. Network component analysis: reconstruction of regulatory signals in biological systems. Proceedings of the National Academy of Sciences of the United States of America, 2003, 100(26): 15522–15527
77. Boulesteix A L, Strimmer K. Predicting transcription factor activities from combined analysis of microarray and ChIP data: a partial least squares approach. Theoretical Biology & Medical Modelling, 2005, 2(1): 23
78. Brynildsen M P, Tran L M, Liao J C. A Gibbs sampler for the identification of gene expression and network connectivity consistency. Bioinformatics (Oxford, England), 2006, 22(24): 3040–3046
79. Brynildsen M P, Wu T Y, Jang S S, Liao J C. Biological network mapping and source signal deduction. Bioinformatics (Oxford, England), 2007, 23(14): 1783–1791
80. Stockham T G, Cannon T M, Ingebretsen R B. Blind deconvolution through digital signal processing. Proceedings of the IEEE, 1975, 63(4): 678–692
81. Kundur D, Hatzinakos D. Blind image deconvolution revisited. IEEE Signal Processing Magazine, 1996, 13(6): 61–63
82. Xu L, Yan P F, Chang T. Semi-blind deconvolution of finite length sequence: (I) linear problem & (II). Nonlinear Problem, SCIENTIA SINICA, Series A, 1987, (12): 1318–1344
83. Zhou Z H. When semi-supervised learning meets ensemble learning. Frontiers of Electrical and Electronic Engineering in China, 2011, 6(1): 6–16
84. De Las Rivas J, Fontanillo C. Protein–protein interactions essentials: key concepts to building and analyzing interactome networks. PLoS Comput Biol, 2010, 6(6): e1000807
85. Han J D. Understanding biological functions through molecular networks. Cell Research, 2008, 18(2): 224–237
86. Davies M. Identifiability Issues in Noisy ICA. IEEE SIGNAL PROCESSING LETTERS, 2004, 11(5): 470–473
87. Morris C. Natural exponential families with quadratic variance functions. Annals of Statistics, 1982, 10(1): 65–80
88. McCullagh P, Nelder J. Generalized Linear Models. 2nd ed. Boca Raton: Chapman and Hall/CRC, 1989
89. Gorman J W, Toman R J. Selection of variables for fitting equations to data. Technometrics, 1966, 8: 27-51
90. Mallows C L. Some comments on Cp. Technometrics, 1973, 15: 661-675
91. Wallace C S, Boulton D M. An information measure for classification. Computer Journal, 1968, 11(2): 185-194
92. Akaike H. A new look at the statistical model identification. IEEE Transactions on Automatic Control, 1974, 19(6): 714-723
93. Solomonoff R J. A formal theory of inductive inference. Part I. Information and Control, 1964, 7(1): 1-22
94. Kolmogorov A N. Three approaches to the quantitative definition of information. Problems of Information Transmission, 1965, 1(1): 1-11
95. Vapnik V. The Nature of Statistical Learning Theory, Springer-Verlag, New York, 1995
96. Xu L, Krzyzak A, Oja E. Rival penalized competitive learning for clustering analysis, RBF net and curve detection. IEEE Transactions on Neural Networks, 1993, 4(4): 636–649
97. Xu L, Krzyzak A, Oja E. Unsupervised and supervised classifications by rival penalized competitive learning. In: Proceedings of the 11th International Conference on Pattern Recognition. 1992, I: 672–675
98. Tu S K, Xu L. Parameterizations make different model selections: empirical findings from factor analysis, to appear on Frontiers of Electrical and Electronic Engineering in China, 2011
99. Sun K, Tu S, Gao D Y, Xu L. Canonical dual approach to binary factor analysis. In: Adali T, Jutten C, Romano J M T, Barros A K, eds. Independent Component Analysis and Signal Separation. Lecture Notes in Computer Science, 2009, 5441: 346-353
100. Xu L. Machine learning problems from optimization perspective. Journal of Global Optimization, 2010, 47(3): 369–401
101. He X F, Lin B B. Tangent space learning and generalization. Frontiers of Electrical and Electronic Engineering in China, 2011, 6(1): 27–42
102. Luxburg U. A tutorial on spectral clustering. Statistics and Computing, 2007, 17(4): 395–416
103. Chung F R. Spectral Graph Theory. Amer.Math. Soc., Providence, RI. MR1421568, 1997
104. Xu L. Distribution approximation, combinatorial optimization, and Lagrange-Barrier. In: Proceedings of International Joint Conference on Neural Networks 2003 (IJCNN 03), Jantzen Beach, Portland. 2003, 2354–2359
105. Xu L. Combinatorial optimization neural nets based on a hybrid of Lagrange and transformation approaches. In: Proceedings Of World Congress on Neural Networks. San Diego, CA. 1994, 399–404
106. Xu L. On the hybrid LT combinatorial optimization: new U-shape barrier, sigmoid activation, least leaking energy and maximum entropy. In: Proceedings of Intl. Conf. on Neural Information Processing, Beijing. 1995, 309–312
107. Xu L. One-bit-matching ICA theorem, convex-concave programming, and combinatorial optimization. In: Advances in neural networks: ISNN 2005, LNCS 3496. Berlin: Springer-Verlag, 2005, 5–20
108. Xu L. One-bit-matching theorem for ICA, convex-concave programming on polyhedral set, and distribution approximation for combinatorics. Neural Computation, 2007, 19(2): 546–569
109. Xu L, Amari S I. Combining Classifiers and Learning Mixture-of-Experts, In: Ramón J, Dopico R, Dorado J, Pazos A, eds. Encyclopedia of Artificial Intelligence. IGI Global (IGI) publishing company, 2008, 318–326
110. Xu L. A unified perspective and new results on RHT computing, mixture based learning, and multi-learner based problem solving. Pattern Recognition, 2007, 40(8): 2129–2153
111. Sun N, Zhao H Y. Reconstructing transcriptional regulatory networks through genomics data. Statistical Methods in Medical Research, 2009, 18(6): 595–617
112. Bar-Joseph Z, Gerber G K, Lee T I, Rinaldi N J, Yoo J Y,

Robert F, Gordon D B, Fraenkel E, Jaakkola T S, Young R A, Gifford D K. Computational discovery of gene modules and regulatory networks. Nature Biotechnology, 2003, 21(11): 1337–1342

113. De Las Rivas J, Fontanillo C. Protein–protein interactions essentials: key concepts to building and analyzing interactome networks. PLoS Comput Biol, 2010, 6(6): e1000807
114. Singh R, Xu J B, Berger B. Global alignment of multiple protein interaction networks with application to functional orthology detection. Proceedings of the National Academy of Sciences of the United States of America, 2008, 105(35): 12763–12768
115. Spirin V, Mirny L A. Protein complexes and functional modules in molecular networks. Proceedings of the National Academy of Sciences of the United States of America, 2003, 100(21): 12123–12128
116. Bu D, Zhao Y, Cai L, Xue H, Zhu X, Lu H, Zhang J, Sun S, Ling L, Zhang N, Li G, Chen R. Topological structure analysis of the protein-protein interaction network in budding yeast. Nucleic Acids Research, 2003, 31(9): 2443–2450
117. Sharan R, Ulitsky I, Shamir R. Network-based prediction of protein function. Molecular Systems Biology, 2007, 3: 88
118. Pinkert S, Schultz J, Reichardt J. Protein interaction networks more than mere modules. PLoS Computational Biology, 2010, 6(1): e1000659
119. Segal E, Shapira M, Regev A, Peer D, Botstein D, Koller D, Friedman N. Module networks: identifying regulatory modules and their condition-specific regulators from gene expression data. Nature Genetics, 2003, 34(2): 166–176
120. Reiss D J, Baliga N S, Bonneau R. Integrated biclustering of heterogeneous genome-wide datasets for the inference of global regulatory networks. BMC Bioinformatics, 2006, 7(1): 280
121. Lemmens K, Dhollander T, De Bie T, Monsieurs P, Engelen K, Smets B, Winderickx J, De Moor B, Marchal K. Inferring transcriptional modules from ChIP-chip, motif and microarray data. Genome Biology, 2006, 7(5): R37 (1–14)
122. Youn A, Reiss D J, Stuetzle W. Learning transcriptional networks from the integration of ChIP-chip and expression data in a non-parametric model. Bioinformatics (Oxford, England), 2010, 26(15): 1879–1886
123. Holter N S, Mitra M, Maritan A, Cieplak M, Banavar J R, Fedoroff N V. Fundamental patterns underlying gene expression profiles: simplicity from complexity. Proceedings of the National Academy of Sciences of the United States of America, 2000, 97(15): 8409–8414
124. Yeung M K, Tegnr J, Collins J J. Reverse engineering gene networks using singular value decomposition and robust regression. Proceedings of the National Academy of Sciences of the United States of America, 2002, 99(9): 6163–6168
125. Alter O, Brown P O, Botstein D. Singular value decomposition for genome-wide expression data processing and modeling. Proceedings of the National Academy of Sciences of the United States of America, 2000, 97(18): 10101–10106
126. Alter O, Brown P O, Botstein D. Generalized singular value decomposition for comparative analysis of genome-scale expression data sets of two different organisms. Proceedings of the National Academy of Sciences of the United States of America, 2003, 100(6): 3351–3356
127. Bussemaker H J, Li H, Siggia E D. Regulatory element detection using correlation with expression. Nature Genetics, 2001, 27(2): 167–174
128. Lee S I, Batzoglou S. Application of independent component analysis to microarrays. Genome Biology, 2003, 4(11): R76
129. Liebermeister W. Linear modes of gene expression determined by independent component analysis. Bioinformatics (Oxford, England), 2002, 18(1): 51–60
130. Sun N, Carroll R J, Zhao H. Bayesian error analysis model for reconstructing transcriptional regulatory networks. Proceedings of the National Academy of Sciences of the United States of America, 2006, 103(21): 7988–7993
131. Sabatti C, James G M. Bayesian sparse hidden components analysis for transcription regulation networks. Bioinformatics, 2006, 22(6): 739–746
132. Liu X, Jessen W J, Sivaganesan S, Aronow B J, Medvedovic M. Bayesian hierarchical model for transcriptional module discovery by jointly modeling gene expression and ChIP-chip data. BMC Bioinformatics, 2007, 8(1): 283
133. Xing B, van der Laan M J. A statistical method for constructing transcriptional regulatory networks using gene expression and sequence data. Journal of Computational Biology, 2005, 12(2): 229–246
134. Pournara I, Wernisch L. Factor analysis for gene regulatory networks and transcription factor activity profiles. BMC Bioinformatics, 2007, 8(1): 61
135. Gardner T S, di Bernardo D, Lorenz D, Collins J J. Inferring genetic networks and identifying compound mode of action via expression profiling. Science, 2003, 301(5629): 102–105
136. Rangel C, Angus J, Ghahramani Z, Lioumi M, Sotheran E, Gaiba A, Wild D L, Falciani F. Modeling T-cell activation using gene expression profiling and state-space models. Bioinformatics (Oxford, England), 2004, 20(9): 1361–1372
137. Beal M J, Falciani F, Ghahramani Z, Rangel C, Wild D L. A Bayesian approach to reconstructing genetic regulatory networks with hidden factors. Bioinformatics (Oxford, England), 2005, 21(3): 349–356
138. Sanguinetti G, Lawrence N D, Rattray M. Probabilistic inference of transcription factor concentrations and gene-specific regulatory activities. Bioinformatics (Oxford, England), 2006, 22(22): 2775–2781
139. Yamaguchi R, Higuchi T. State-space approach with the maximum likelihood principle to identify the system generating time-course gene expression data of yeast. International Journal of Data Mining and Bioinformatics, 2006, 1(1): 77–87
140. Li Z, Shaw S M, Yedwabnick M J, Chan C. Using a state-space model with hidden variables to infer transcription factor activities. Bioinformatics (Oxford, England), 2006, 22(6): 747–754
141. Inoue L Y, Neira M, Nelson C, Gleave M, Etzioni R. Cluster-based network model for time-course gene expression data. Biostatistics (Oxford, England), 2007, 8(3): 507–525
142. Martin S, Zhang Z, Martino A, Faulon J L. Boolean dynam-

ics of genetic regulatory networks inferred from microarray time series data. Bioinformatics (Oxford, England), 2007, 23(7): 866–874

143. Hirose O, Yoshida R, Imoto S, Yamaguchi R, Higuchi T, Charnock-Jones D S, Print C, Miyano S. Statistical inference of transcriptional module-based gene networks from time course gene expression profiles by using state space models. Bioinformatics (Oxford, England), 2008, 24(7): 932–942
144. Xiong H, Choe Y. Structural systems identification of genetic regulatory networks. Bioinformatics (Oxford, England), 2008, 24(4): 553–560
145. Wu F X, Zhang W J, Kusalik A J. State-space model with time delays for gene regulatory networks. Journal of Biological System, 2004, 12(4): 483–500
146. Shiraishi Y, Kimura S, Okada M. Inferring cluster-based networks from differently stimulated multiple time-course gene expression data. Bioinformatics (Oxford, England), 2010, 26(8): 1073–1081
147. Kim T Y, Kim H U, Lee S Y. Data integration and analysis of biological networks. Current Opinion in Biotechnology, 2010, 21(1): 78–84
148. Xu L, Pearl J. Structuring causal tree models with continuous variables. In: Proceedings of the 3rd Annual Conference on Uncertainty in Artificial Intelligence. 1987, 170–179
149. Xu L, Pearl J. Structuring Causal Tree Models with Continuous Variables. In: Kanal L N, Levitt T S, Lemmer J F, eds. Uncertainty in Artificial Intelligence 3. North Holland, Amsterdam, 1989, 209–219

Lei Xu is a chair professor of Chinese University of Hong Kong (CUHK), a Chang Jiang Chair Professor of Peking University, a guest Professor of Institute of Biophysics, Chinese Academy of Sciences, an honorary Professor of Xidian University. He graduated from Harbin Institute of Technology by the end of 1981, and completed his master and Ph.D thesis at Tsinghua University during 1982-86.Then, he joined Department Mathematics, Peking University in 1987 first as a postdoc and then exceptionally promoted to associate professor in 1988 and 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 senior lecturer, became professor in 1996 and took the current position since 2002. Prof. Xu has published dozens of journal papers and also many papers in conference proceedings and edited books, covering the areas of statistical learning, neural networks, and pattern recognition, with a number of well-cited papers, e.g., his papers got over 3200 citations according to SCI Expended (SCI-E) and over 6000 citations according to Google Scholar (GS), and over 2100 (SCI-E) and 3800 (GS) for his 10 most frequently cited papers. He served as associate editor for several journals, including Neural Networks (1995–present) and IEEE Transactions on Neural Networks (1994-98), and as general chair or program committee chair of a number of international conferences. Moreover, Prof. Xu has served on governing board of International Neural Networks Society (INNS) (2001-03), INNS Award Committee (2002-03), and Fellow Committee of IEEE Computational Intelligence Society (2006, 2008), chair of Computational Finance Technical Committee of IEEE Computational Intelligence Society (2001-03), a past president of Asian-Pacific Neural Networks Assembly (APNNA) (1995–96), and APNNA Award Committee (2007-09). He has also served as an engineering panel member of Hong Kong RGC Research Committee (2001-06), a selection committee member of Chinese NSFC/HK RGC Joint Research Scheme (2002-05), external expert for Chinese NSFC Information Science (IS) Panel (2004-06, 2008), external expert for Chinese NSFC IS Panel for distinguished young scholars (2009-10), and a nominator for the prestigious Kyoto Prize (2003, 2007). Prof. Xu has received several Chinese national academic awards (including 1993 National Nature Science Award) and international awards (including 1995 INNS Leadership Award and the 2006 APNNA Outstanding Achievement Award). He has been elected to an IEEE Fellow since 2001 and a Fellow of International Association for Pattern Recognition and a member of European Academy of Sciences since 2002.