Hybrid Method of Boolean Factor Analysis

Húsek, Dušan
2011

Institute of Computer Science
Academy of Sciences of the Czech Republic

Hybrid Method of Boolean Factor Analysis

Húsek Dušan, Frolov Alexander, Polyakov Pavel

Technical report No. 1115

5.5.2011

Pod Vodárenskou věží 2, 182 07 Prague 8, phone: +420 266 053 230, fax: +420 286 585 789,
e-mail:dusan@cs.cas.cz

Institute of Computer Science
Academy of Sciences of the Czech Republic

Hybrid Method of Boolean Factor Analysis [1]

Húsek Dušan, Frolov Alexander, Polyakov Pavel

Technical report No. 1115

5.5.2011

Abstract:

Usual task in large data set analysis is the search for their appropriate representation in the space of less dimension. One of the most efficient methods to solve this task is a factor analysis. In this study we suggest a new approach to Boolean factor analysis, which is extension of the previously proposed Boolean factor analysis method: Hopfield-like Attractor Neural Network with Increasing Activity. We increased its functionality when complementing this method by a maximization of the learning set likelihood function defined according to the proposed Boolean factor analysis generative model. Such a way we can obtain as a result full set of generative model parameters. We demonstrated the efficiency of the new method using the artificial signals generated according the generative model. Successful application of the method to the real data is shown when analyzing the data from Kyoto Encyclopedia of Genes and Genomes database which contains full genome sequencing for 1368 organisms.

Keywords:
Recurrent neural network, associative memory, neural network application, statistics, likelihood maximization, Boolean factor analysis, information gain, expectation-maximization, dendritic inhibition, Boolean matrix factorization, bars problem

[1] **Acknowledgement:** This work has been partly elaborated not only in the framework of the institutional project AV0Z10300504,but we have also thank to other the institutions that provided financing of our projects under co-denames GACR P202/10/0262, GACR 205/09/1079, and IT4Innovations Centre of Excellence project, reg. no. CZ.1.05/1.1.00/02.0070.

1 Introduction

Factor analysis is one of the most powerful statistical methods to reveal and reduce information redundancy in high dimensional signals. Boolean Factor Analysis (BFA) implies that all: the components of the signals, factor loadings and factor scores are binary variables. Each binary component of the signal can be interpreted as a representation of the presence or the absence of an attribute in the observation (pattern). The number of considered attributes is the dimension of the signal space, the presence of an attribute is encoded as One, and its absence as Zero. The patterns are assumed to be composed of many "objects". We define an object as a collection of highly correlated attributes and suppose that objects are relatively independent of one another. Hence the attributes of different objects are only slightly correlated. In terms of BFA, objects are factors, attributes constituting the object are factor loadings, and the presence or absence of an object in the pattern is identified by the value of the factor score (One or Zero). Correlations between the attributes constituting each factor can be revealed by statistics over the large data set constituted by patterns that contain each factor many times in different combinations with other factors. The aim of BFA is to detect this hidden structure of the signal space and to form a representation in which these independent objects are presented explicitly. A factor may also be interpreted as a hidden cause resulting in the sets of observations [1, 2]. For example in medical research, a cause is a syndrome and an observation is a symptom [3, 4].

In spite of the fact that binary data representations are typical in many fields, including social science, marketing, zoology, genetics, and medicine, BFA methods have only been rather moderately developed. We proposed earlier [5] a BFA method that is based on the Hopfield-like attractor neural network. This Attractor Neural Network with Increasing Activity is referred here as ANNIA. The method builds on the well known property of Hopfield network to create attractors of the network dynamics by assemblies of tightly connected neurons. Since the neurons representing a factor are activated simultaneously each time when the factor appears in the patterns of the data set, and neurons representing different factors are rather seldom activated simultaneously, then - due to the Hebbian learning rule - the factor neurons become more tightly connected than the other neurons. So factors can be revealed as attractors of the network dynamics. In our previous papers [6, 7, 8], we demonstrated the method performance using artificial data set and sets of the real data such as the mushroom, parliament voting and textual data sets. Although the method showed high ability to reveal the hidden factor structure in the artificial data, it was impossible to evaluate its performance in analyzing real data. The reason was the absence of a general blind measure of BFA efficiency that would allow, on one hand, to compare different BFA methods and, on the other hand, to estimate whether given data set is appropriate for BFA at all. To overcome this lack we recently proposed a general information theoretic measure of BFA efficiency which is the difference of two entropies. The first is the entropy of a data set when its hidden factor structure is unknown, and the second is the entropy when it is revealed and taken into account [9]. Thus, this measure is the information gain provided by BFA. Estimation of the entropy of the data set when its hidden structure is revealed is based on the supposed generative model of signals, adequate for BFA. We have shown on artificial signals that the information gain is sensitive to both the noise in the signals and the errors in the BFA results. Thus, information gain seems to be a reliable basis for comparing different BFA methods and for detecting the presence of hidden factor structures in a given data set as well.

The offered BFA generative model is not only the base for the new measure of BFA efficiency but it also provides the possibility to improve ANNIA complementing it by the procedure of the Likelihood Maximization (LM). In the experimental part of the paper, we estimate the efficiency of hybrid ANNIA & LM method, called LANNIA ("Likelihood Attractor Neural Network with Increasing Activity") in the next, using artificially generated signals and Kyoto Encyclopedia of Genes and Genomes (KEGG) [10] data set containing full genome sequencing for 1368 organisms.

The paper is organized as follows. A general BFA generative model and information gain are proposed in Sections 2 and 3. The procedure for likelihood maximization is described in Section 4. The hybrid method composed of ANNIA and LM methods (LANNIA) is described in Section 5. The performance of LANNIA on KEGG data set is described in Section 6.

2 Generative model of signals appropriate for BFA

In formulating a generative model of signals appropriate for BFA, we follow the ideas of Barlow [11], Marr [12], Kussul [13] and others who assumed that the search for a hidden factor structure in incoming sensory signals is one of the main brain functions. Explaining Barlow's ideas, Foldiak [14] writes: "According to Barlow [11] objects (and also features, concepts or anything that deserves a name) are collections of highly correlated properties. For instance, the properties 'furry', 'shorter than a meter', 'has a tail', 'moves', 'animal', 'barks', etc. are highly correlated, that is the combination of these properties is much more frequent than it would be if they were independent (the probability of the conjunction is higher than the product of individual probabilities of the component features). It is these non-independent, redundant features, the 'suspicious coincidences' that define objects, features, concepts, categories, and these are what we should be detecting. While components of objects can be highly correlated, objects are relatively independent of one another... The goal of the sensory system might be to detect these redundant features and to form a representation in which these redundancies are reduced and the independent features and objects are represented explicitly".

In terms of BFA, each pattern of a signal space is defined by a binary row vector $\mathbf{x} = [x_1, \ldots, x_N]$, where N is the total number of attributes. Every component of $\mathbf{x}$ takes value One or Zero, depending on the presence or absence of the related attribute. Each factor $\mathbf{f}_i = [f_{i1}, \ldots, f_{iN}]$ is a binary row vector of dimension N whose entries with values equal to One correspond to highly correlated attributes of the i^{th} object. Although the probability of the presence of attribute of an object in an observation simultaneously with its other attributes is high, it is not necessarily equal to 1. For example, the attribute 'has a tail' is not always present in an observation when the object 'dog' is present. We denote this probability by p_{ij}, where j is the index of the attribute and i is the index of the factor. For attributes constituting the factor, i.e. for attributes with $f_{ij} = 1$, the probability p_{ij} is high, and for the other attributes (with $f_{ij} = 0$), it is zero.

As in linear factor analysis, we suppose that in addition to common factors $\mathbf{f}_i$ that influence more than one attribute, each signal also contains N specific or unique factors that influence only particular attributes. The contribution of specific factors is defined by a binary row vector $\boldsymbol{\eta} = [\eta_1, \ldots, \eta_N]$. Each specific factor η_j is characterized by the probability q_j with which η_j takes on the value One.

As a result, any vector $\mathbf{x}$ can be presented in the form

$$x_j = [\bigvee_{i=1}^{L} s_i \wedge f'_{ij}] \vee \eta_j, \tag{2.1}$$

where $\mathbf{s} = [s_1, \ldots, s_L]$ is a binary row vector of factor scores of dimension L, L being the total number of factors, $\mathbf{f}'_i = [f'_{i1}, \ldots, f'_{iN}]$ is a distorted version of factor loadings $\mathbf{f}_i$ and $\boldsymbol{\eta}$ is a vector of specific factors. Factor distortion implies that entries of $\mathbf{f}_i$ having value equal to One can change their values to Zero with probability $1 - p_{ij}$ before mixing in the observed pattern but none of the entries of $\mathbf{f}_i$ equal to Zero can can change value to One in the distorted version of the factor because the probability for them to transform to One is zero ($p_{ij} = 0$). We assume that factors appear in patterns (that is related scores s_i take Ones) independently with probabilities π_i ($i = 1, \ldots, L$), factors are distorted independently of other factors and specific factors, factor components are distorted independently of other components, and specific factors are independent of each other and of the common factors.

The aim of Boolean factor analysis is to find the parameters of a generative model $\boldsymbol{\Theta} = (p_{ij}, q_j, \pi_i, i = 1, \ldots, L, j = 1, \ldots, N)$ and factor scores $\mathbf{s}_m$ ($m = 1, \ldots, M$) for all M patterns $\mathbf{x}_m$ of the observed data set. However, it is supposed that the factors found could also be detected in any arbitrary pattern $\mathbf{x}$, if generated by the same BFA model. Note that the finding of p_{ij} implies the finding of factor loadings f_{ij} since $f_{ij} = sgn(p_{ij})$.

The procedure of generation of signals according to this generative model is given in Algorithm 2.

3 Informational gain

The proposed generative model allows to define the BFA information gain. If factor structure of the signal space is unknown, then representing the j^{th} component of vector $\mathbf{x}$ requires $h(p_j)$ bits of

information, where $h(x) = -x\log_2 x - (1-x)\log_2(1-x)$ is Shannon function and p_j is the probability of the j^{th} component's taking One. Representing the whole data set requires

$$H_0 = M\sum_{j=1}^{N} h(p_j) \tag{3.1}$$

bits of information. If the hidden factor structure of the signal space is detected, that is all the generative model parameters and all factor scores in the data set are found, then representing the whole data set requires

$$H = H_1 + H_2 \tag{3.2}$$

bits of information. Here

$$H_1 = M\sum_{i=1}^{L} h(\pi_i) \tag{3.3}$$

is the information required to represent the factor scores and

$$H_2 = \sum_{m=1}^{M}\sum_{j=1}^{N} h(P(x_{mj}|\mathbf{s}_m, \mathbf{\Theta})) \tag{3.4}$$

is the information required to represent all patterns of the data set when factor scores are given. In (3.4)

$$P(x_{mj}|\mathbf{s}_m, \mathbf{\Theta}) = x_{mj} - (2x_{mj}-1)(1-q_j)\prod_{i=1}^{L}(1-p_{ij})^{s_{mi}} \tag{3.5}$$

is the probability of the j^{th} component of m^{th} signal $\mathbf{x}_m$ to take the value x_{mj}.

The information gain is determined by the difference between H_0 and H. We define the relative information gain as

$$G = (H_0 - H)/H_0. \tag{3.6}$$

From a practical point of view BFA is meaningful only if $G > 0$.

The information gain decreases when both the noise in data set increases and the errors in BFA solution increases [9]. Thus it is a reliable measure of the BFA quality and the adequacy of BFA to a given data set at all.

4 Likelihood maximization (LM)

If we have a generative model in terms of probabilities (as we did above) it is clear that we can set up likelihood function and solve the task by using likelihood maximization.

$$\Lambda = \sum_{m=1}^{M} \Lambda_m, \tag{4.1}$$

where

$$\Lambda_m = \log[P(\mathbf{s}_m|\mathbf{\Theta})P(\mathbf{x}_m|\mathbf{s}_m, \mathbf{\Theta})], \tag{4.2}$$

$$P(\mathbf{x}_m|\mathbf{s}_m, \mathbf{\Theta}) = \prod_{j=1}^{N} P(x_{mj}|\mathbf{s}_m, \mathbf{\Theta}), \tag{4.3}$$

$$P(\mathbf{s}_m|\mathbf{\Theta}) = \prod_{i=1}^{L} \pi_i^{s_{mi}} (1-\pi_i)^{1-s_{mi}}, \tag{4.4}$$

and $P(x_{mj}|\mathbf{s}_m, \mathbf{\Theta})$ is given by (3.5).

To maximize Λ we use the iterative procedure that is the usual method for likelihood function maximization [15]. The iterations alternatively increase Λ with respect to a set of factor scores s_{mi} ($m = 1, \ldots, M$, $i = 1, \ldots, L$), while holding $\mathbf{\Theta}$ fixed (the E-step), and with respect to parameters of the model $\mathbf{\Theta}$, while holding s_{mi} fixed (the M-step).

At the M-step, when scores are fixed, p_{ij} and q_j can be found by maximization of Λ according to the following iterative procedure:

$$\Delta p_{ij} = \gamma_{ij} \frac{\partial \Lambda}{\partial p_{ij}}, \qquad \Delta q_j = \gamma_j \frac{\partial \Lambda}{\partial q_j}, \tag{4.5}$$

where γ_{ij} and γ_j are positive learning rates and according to (4.1), (4.2) and (4.3)

$$\begin{aligned} \frac{\partial \Lambda}{\partial p_{ij}} &= \sum_{m=1}^{M} P(x_{mj}|\mathbf{s}_m, \mathbf{\Theta})^{-1} \frac{\partial P(x_{mj}|\mathbf{s}_m, \mathbf{\Theta})}{\partial p_{ij}} \\ \frac{\partial \Lambda}{\partial q_j} &= \sum_{m=1}^{M} P(x_{mj}|\mathbf{s}_m, \mathbf{\Theta})^{-1} \frac{\partial P(x_{mj}|\mathbf{s}_m, \mathbf{\Theta})}{\partial q_j}, \end{aligned}$$

and

$$\begin{aligned} \frac{\partial P(x_{mj}|\mathbf{s}_m, \mathbf{\Theta})}{\partial p_{ij}} &= (x_{mj} - P(x_{mj}|\mathbf{s}_m, \mathbf{\Theta})) \frac{s_{mi}}{1-p_{ij}} \\ \frac{\partial P(x_{mj}|\mathbf{s}_m, \mathbf{\Theta})}{\partial q_j} &= (x_{mj} - P(x_{mj}|\mathbf{s}_m, \mathbf{\Theta})) \frac{1}{1-q_j}. \end{aligned} \tag{4.6}$$

As we assume that the probabilities p_{ij} are sufficiently high for the components constituting the i^{th} factor ($f_{ij} = 1$) and equal to zero for the other components ($f_{ij} = 0$), at each iteration cycle of step M we put $p_{ij} = 0$, if

$$p_{ij} < 1 - \prod_{l \neq i} (1 - \pi_l p_{lj}), \tag{4.7}$$

where the right side of the inequality is the probability that the j-th attribute appears in the pattern due to other factors except $\mathbf{f}_i$. It is worth noting that without threshold truncation of p_{ij}, LM does not converge at all because of uncertainties arising from the competition between common and specific factors. For example one of the common factors can contain all Ones, then it will be functionally indistinguishable from specific factors because there is no preference to prescribe Ones of the observed signals to this common factor or to specific factors. To eliminate this uncertainty most of the components of each factor are put to be zero.

In our computer experiments we set the learning rates in (4.5) to be

$$\gamma_{ij} = p_{ij}(1-p_{ij})/(M\pi_i), \qquad \gamma_j = q_j(1-q_j)/M. \tag{4.8}$$

We terminated the search for p_{ij} and q_j at the M-step by the condition $\sum_{i,j} |\Delta p_{ij}| / \sum_{i,j} p_{ij} < 10^{-5}$.

The generative model parameters p_{ij} and q_j obtained at the M-step are used as the input for the next E-step to find factor scores. For each individual signal $\mathbf{x}_m$ of the data set, factor scores $\mathbf{s}_m$ can be found as those maximizing Λ_m. The global maximum of Λ_m can be provided only by the exhaustive search. However, the number of possible $\mathbf{s}_m$ is usually large (equal to 2^L), then to use some iterative procedure, even if it provides local maximum, is more reasonable. One of the possible procedures is following.

At each iterative step the values $\Lambda_m|_{s_{mi}=1}$ and $\Lambda_m|_{s_{mi}=0}$ obtained by substituting $s_{mi} = 1$ and $s_{mi} = 0$ into (4.2) are compared. The value of s_{mi} that provides the greater $\Lambda_m|_{s_{mi}}$ is chosen and the procedure goes to another i until it converges. In our experiments, we used a two-run iterative

procedure to compute $\mathbf{s}_m$. At each external cycle of the procedure all components of $\mathbf{s}_m$ were processed to maximize Λ_m. The sequence of their processing was randomly permuted at each cycle. The procedure was terminated when $\mathbf{s}_m$ remained the same at the next cycle. The procedure converges because at each iterative step the likelihood function does not decrease. The procedure starts with all $s_{mi} = 0$. After computing $\mathbf{s}_m$ the procedure is applied to the next signal $\mathbf{x}_{m+1}$ until the data set is exhausted.

The scores found at the E-step are used as input to the next M-step. If for some factors, all found loadings or scores are zeros, these factors are excluded from the list of found factors.

The LM-iterative procedure terminates when the increment of the Λ at the next step does not exceed $10^{-6}MN$ or the number of steps in the LM procedure reaches 10.

5 Hybrid ANNIA & LM method

As described in our previous papers [5, 8] our neural network based BFA method ANNIA itself does not solve the task in its entirety. It provides an accurate estimation of the factor loadings, an approximate estimation of factor scores, and no estimation of the parameters of the generative model p_{ij} and q_j. A way to overcome this drawback is to combine ANNIA with LM. We will show in the this section that the LM procedure itself is able to provide complete solution of the BFA problem but requires an appropriate initial approximation. If it starts from the random initial parameters it almost always fails. In the combination of ANNIA and LM the role of ANNIA is to provide LM with the initial approximation, particularly an initial set of the scores $\mathbf{s}_m$ $(m = 1, \ldots, M)$. Another aspect of the ANNIA and LM interaction is a suppression of the dominant attractors in ANNIA using the data provided by LM. Both these aspects of the hybrid LANNIA procedure are discussed in this Section.

The LANNIA performance is illustrated below when solving the so called Bars Problem (BP) [14]. The BP in various modifications has been considered in many papers (see [1], for references) as a benchmark for the learning of objects from complex patterns. In this problem, each pattern of the data set is an n by n binary pixel image containing several of $L = 2n$ possible (one-pixel wide) horizontal and vertical bars (Fig. 5.1). Pixels constituting a bar take the values 1 and pixels not constituting it take the values 0. For each image, each bar could be chosen with the probability C/L, where C is the mean number of bars mixed in an image. At the point of intersection of a vertical and a horizontal bar, the pixel takes the value 1. This Boolean summation of pixels belonging to different bars simulates the occlusion of objects. The task is to recognize all bars as individual objects on the basis of a data set containing M images consisting of bar mixtures. In most papers where the BP was used as benchmark, C was set to 2 and $n = 8$.

In terms of BFA, bars are factors. Factor loadings f_{ij} $(j = 1, \ldots, N)$ take value One for pixels constituting the i^{th} bar and value Zero for pixels not constituting it. Each image is a Boolean superposition of factors, and the factor score takes the value One or Zero depending on the presence or absence of a bar in the image. Thus, the bars problem is a special case of BFA. We consider the case of the homogeneously distributed noise in the images, both in the form of the factor distortion and in the form of the specific factors. Particularly, we put $p_{ij} = pf_{ij}$ and $q_j = q$. This means that pixels constituting a bar can take Zero with the equal probabilities $1 - p$ and any pixel can take One with the probability q due to the specific factor.

Initially the LANNIA performance is illustrated in the absence of noise, that is, when $p = 1$ and $q = 0$ and then for the case of the rather noisy data ($p = 0.7$ and $q = 0.2$) when the hidden factor structure of the input signals becomes practically invisible (Fig. 5.1**C**).

The method ANNIA is based on the network of N neurons corresponding to N binary coordinates of a signal space. All patterns of the data set are stored in the network by the Hebbian learning rule:

$$J'_{ij} = \sum_{m=1}^{M} (x_{mi} - a_m)(x_{mj} - a_m), \qquad i, j = 1, \ldots, N, \; i \neq j, \qquad J'_{ii} = 0, \tag{5.1}$$

where $a_m = \sum_{i=1}^{N} x_{mi}/N$ is the total activity of the m-th pattern.

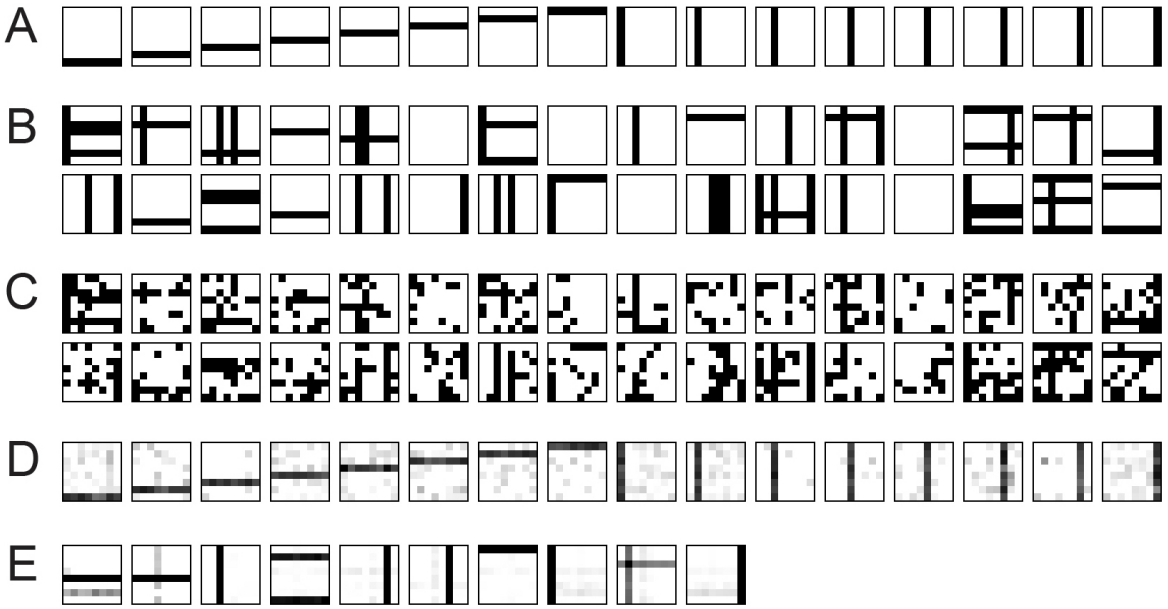


Figure 5.1: **A** Sixteen vertical and horizontal bars in 8 by 8 pixel images. **B** Examples of images in the standard bars problem. Each image contains two bars on average. **C** Examples of noisy images ($p = 0.7$, $q = 0.2$). **D** Factors found by LANNIA when solving BP with the noisy images, the number of patterns in the data set is $M = 800$. **E** Factors found by LM in the absence of noise in the images, $M = 800$. In **D** and **E**, the black pixels correspond to $p_{ij} = 1$, the white pixels correspond to $p_{ij} = 0$, and the grey pixels correspond to the intermediate values of p_{ij}.

Under conditions discussed in our papers [16, 5, 8, 17], two global spurious attractors dominate in the network dynamics. To suppress their dominance the matrix J''_{ij} is subtracted from the connection matrix $\mathbf{J}'$ where

$$J''_{ij} = M(p_i - \bar{p})(p_j - \bar{p}), \qquad i, j = 1, \ldots, N,\ i \neq j, \qquad J''_{ii} = 0, \tag{5.2}$$

p_j is a frequency taking One in the data set for j^{th} signal component and $\bar{p}$ is p_j averaged over all components. Although these global attractors do not appear during solving BP, we use the general procedure which guarantees their absence. Thus, both BP in this section and genome data set KEGG in the next section are analyzed by ANNIA when they are eliminated, that is, when the connection matrix is transformed to

$$\mathbf{J} = \mathbf{J}' - \mathbf{J}''. \tag{5.3}$$

ANNIA reveals factors as attractors of the network dynamics in a two-run recall procedure. Its initialization starts by the presentation of a random initial pattern $\mathbf{x}^{in}$ with k_{in} active neurons (k_{in} is supposed to be smaller than the number of active neurons in any factor). On the presentation of $\mathbf{x}^{in}$ the network activity $\mathbf{x}$ evolves to an attractor according to synchronous discrete time dynamics. At each time step, k_{in} winners with the highest synaptic excitations are activated. The excitations are calculated as $\mathbf{xJ}$, where $\mathbf{x}$ is the network state at the previous time step. When activity stabilizes at the initial level of activity k_{in}, then a neuron with the maximal excitation $T(k_{in})$ is selected over all not active neurons, and added to already active k_{in} neurons of the attractor. In fact, $T(k_{in})$ is a threshold of excitation for non-active neurons to activate only one of them. The obtained pattern with $k_{in} + 1$ active neurons is treated as the initial network state for the next iteration step, and the network activity evolves to an attractor at the new level of activity $k_{in} + 1$. The level of activity then increases to $k_{in} + 2$, and so on, until the number of active neurons reaches the final level k_{fin} which is supposed to be higher than the number of active neurons in any factor. Thus, one trial of the

recall procedure contains $k_{fin} - k_{in}$ external steps and several internal steps (usually 2-3) inside each external step to reach an attractor for a given level of activity.

At the end of each external step, when the network activity stabilizes at the level of k active neurons, a Lyapunov function is calculated:

$$\lambda(k) = \mathbf{x}(t+1)\mathbf{J}\mathbf{x}^T(t)/k, \tag{5.4}$$

where $\mathbf{J}$ is the matrix of synaptic connections and $\mathbf{x}(t+1)$ and $\mathbf{x}(t)$ are the two network states in a possible cyclic attractor of length 2 (for a point attractor, $\mathbf{x}(t+1) = \mathbf{x}(t)$). In fact, $\lambda(k)$ is the mean synaptic excitation of k active neurons. As suggested in [5], the identification of factors is based on the analysis of the dynamics of the Lyapunov function $\lambda(k)$ and the activation threshold $T(k)$ in the recall procedure. At the initial part of the recall trajectory, when $k < n_f$ ($n_f = 8$ is the number of active neurons in the factor), stable states are factor fragments and their neurons are tightly connected. Hence, their mean synaptic excitation increases almost proportionally to k, but when k reaches n_f, $\lambda(k)$ sharply breaks (Fig. 5.2 a) because the neurons added to attractors when $k > n_f$ do not belong to the factor and thus only slightly excite its neurons. For the same reason, the activation threshold $T(k)$ also increases proportionally to k when $k < n_f$, and then jumps down at the point $k = n_f$ (Fig. 2(b)). As shown in Fig. 2(d), the derivative $R'(k) = R(k) - R(k-1)$ of the function $R(k) = \lambda(k)/(k-1) - T(k)/k$ has a distinctly exposed peak at this point. Thus, the maximum of $R'(k)$ was used as an indicator of the factor on each recall trajectory. The state of the neural network activity at the maximum gives the factor loadings $\mathbf{f}$ for the found factor.

Some trajectories in Fig. 2(d) have a second, weaker, peak of $R'(k)$ at the points $k = 15$ or $k = 16$. These peaks correspond to pairs of factors most often appearing together in the data set images. The point $k = 15$ corresponds to the crossing bars, and $k = 16$ corresponds to the parallel bars. The similar but weaker peaks appear also at points $k = 22$, $k = 23$ and $k = 24$ corresponding to images containing three bars that most often appeared together in the data set, and so on. Thus, the method is able to extract some additional information on the data set besides revealing its factor structure.

As shown in Fig. 2(a), sometimes Lyapunov function jumps up from one to another continuous trajectory. In this step, the network activity transitions to an attractor far from the attractor at the previous step. As shown in Fig. 2(d), such a transition could also produce a peak of R'. To avoid falsely treating such transition as factors, we calculated at each point of each trajectory the similarity $Sim(k)$ between the patterns of the network activity in the current attractor $\mathbf{x}^{at}(k)$ and in the previous attractor $\mathbf{x}^{at}(k-1)$ as

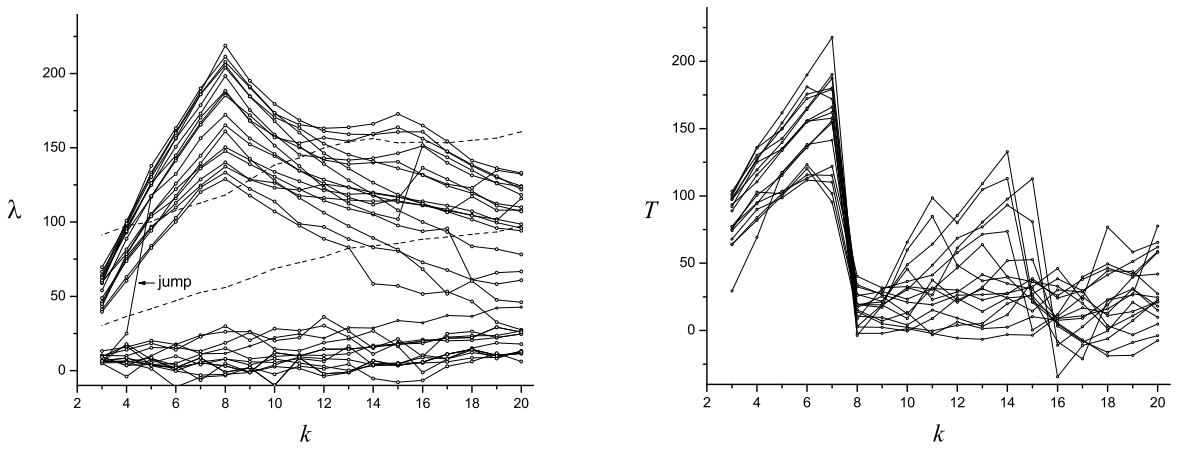
$$Sim(k) = \frac{a - (k-1)k/N}{(k-1)(1-k/N)}, \tag{5.5}$$

where a is the number of common Ones in $\mathbf{x}^{at}(k)$ and $\mathbf{x}^{at}(k-1)$. If $\mathbf{x}^{at}(k)$ contains $\mathbf{x}^{at}(k-1)$, then $Sim(k) = 1$. If $\mathbf{x}^{at}(k)$ and $\mathbf{x}^{at}(k-1)$ are independent, then $Sim(k)$ is equal to zero on average. We assumed that the pattern of the network activity changes smoothly along the trajectory if $Sim(k) \geq Sim_{thr}$, where $Sim_{thr} = 0.8$. In the opposite case, we treated the transition from $\mathbf{x}^{at}(k-1)$ to $\mathbf{x}^{at}(k)$ as a jump. Thus, the point on the trajectory with the largest peak for R' could be considered as related to the factor only if there was no jump at this point.

The sizes of attraction basins around factors are distributed in a large range. They are proportional to the values of the Lyapunov function of factors, which, in turn, is proportional to the frequency of their appearance in the patterns of the data set [5]. When the initial network states are chosen randomly, as in the procedure described above, the network activity tends to converge to the factors with the largest attraction basins. To suppress the dominance of these factors and make the search of new ones possible, we unlearned them from the network memory by subtracting the matrix $\Delta\mathbf{J}^i$ from the matrix of synaptic connections $\mathbf{J}$ for each found factor where

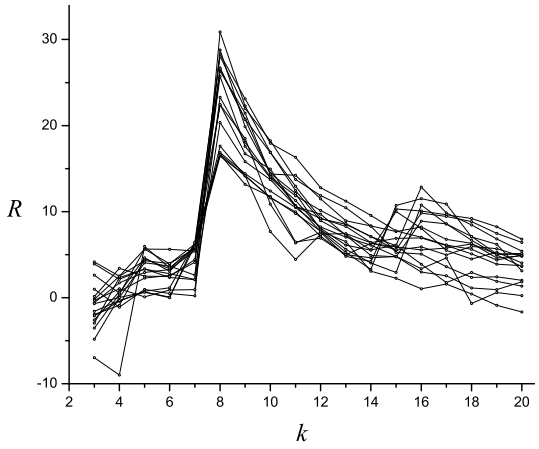
$$\Delta J^i_{jk} = M\pi_i(1-\pi_i)p_{ij}(1-p^0_{ij})p_{ik}(1-p^0_{ik}), \quad k \neq j, \qquad \Delta J^i_{jj} = 0, \tag{5.6}$$

p_{ij} is a probability that the j^{th} component takes One in the i^{th} found factor, p^0_{ij} is a probability that the j^{th} component takes One in signals not containing i^{th} found factors. Probability p_{ij} is a parameter of the generative model, p^0_{ij} can be estimated as frequency of the j^{th} component taking One in signals

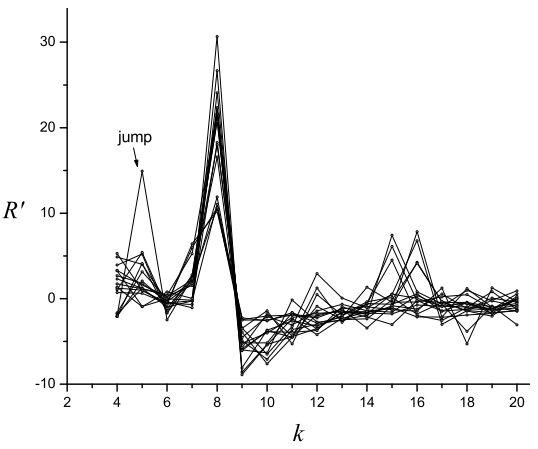


(a) a

(b) b



(c) c



(d) d

Figure 5.2: Lyapunov function λ (a), activation threshold T (b), function $R = \lambda/(k-1) - T/k$ (c) and its derivative R' (d) in dependence on the number of active neurons k. Dashed lines in (a) are thresholds for separating true and spurious trajectories at the beginning (upper line) and at the end (lower line) of the recall procedure. The example of a jump from one to another continuous trajectory is marked in (a) and (d). The results were obtained for the artificial BP data set containing $M = 400$ patterns without noise ($p = 1$, $q = 0$).

of the data set not containing the i^{th} factor. Substantiation of the unlearning rule (5.6) is given in Appendix 1. Since ANNIA does not provide direct estimates of generative model parameters required for the unlearning rule (5.6), they are obtained by using the LM procedure.

The LM procedure is initiated from E-step. To estimate the probabilities p_{ij} required to start LM procedure we use

$$p_{ij} = \begin{cases} h_{ij} / \max\limits_{j'=1\ldots N} \{h_{ij'}\} & \text{if } h_{ij} > 0 \\ 0 & \text{if } h_{ij} \le 0 \end{cases}, \tag{5.7}$$

where $\mathbf{h}_i = \mathbf{J}\mathbf{f}_i$ is a vector of postsynaptic excitations of neurons when the i^{th} found factor is activated in the network. This estimation seems to be reasonable because h_{ij} is proportional to p_{ij} [8]. After the LM convergence, the obtained probabilities p_{ij} and p^0_{ij} are used to suppress attractors for found factors. This allows for searching the other factors with the lower Lyapunov function. The probabilities p_{ij} and q_j calculated at the current cycle of the LANNIA procedure are used as input to LM procedure at its next cycle. At the first cycle of LANNIA the probabilities q_j are taken to be zero.

The parameters of the generative model and the factor scores obtained by LM are used to calculate the information gain provided by all found factors. The LANNIA procedure continues until G stops to increase due to adding new found factors. This is the first criterion to terminate LANNIA.

The network dynamics can converge not only to the true attractors corresponding to the factors, but also to spurious attractors far from all factors [5]. The Lyapunov function for the spurious attractors is smaller than that for factors (Fig. 2(a)). To separate the true attractors from the spurious ones, we use the following heuristic method. For each k we activate a random set of k neurons and find the maximal synaptic excitation over all neurons of the network. We repeat this procedure 100 times and calculate mean $m(k)$ and standard deviation $\sigma(k)$ of the maximal excitations. If the Lyapunov function in the peak of R' along the trajectory satisfies the following inequality

$$\lambda(k_p) > h_{max}(k_p) = m(k_p) + 3\sigma(k_p) \tag{5.8}$$

we treat the found point on the trajectory as a factor, in the opposite case — as a spurious state. The borders h_{max} separating true and spurious trajectories are shown in Fig. 2(a) by the dashed lines. The upper curve corresponds to the beginning of the recall procedure when the first factor with the highest Lyapunov function was found. The lower curve corresponds to the end of the recall procedure when the last factor with the lowest Lyapunov function was found. Note that the border $h_{max}(k)$ separating the true and spurious trajectories markedly decreases as the factor discovering proceeds. The spurious trajectories shown in Fig. 2(a) obtained in solving BP appeared only after unlearning all factors. The appearance of only spurious attractors in the recall procedure indicates that all factors are found. This is the second termination criterion LANNIA.

Let us summarize the main steps of LANNIA.

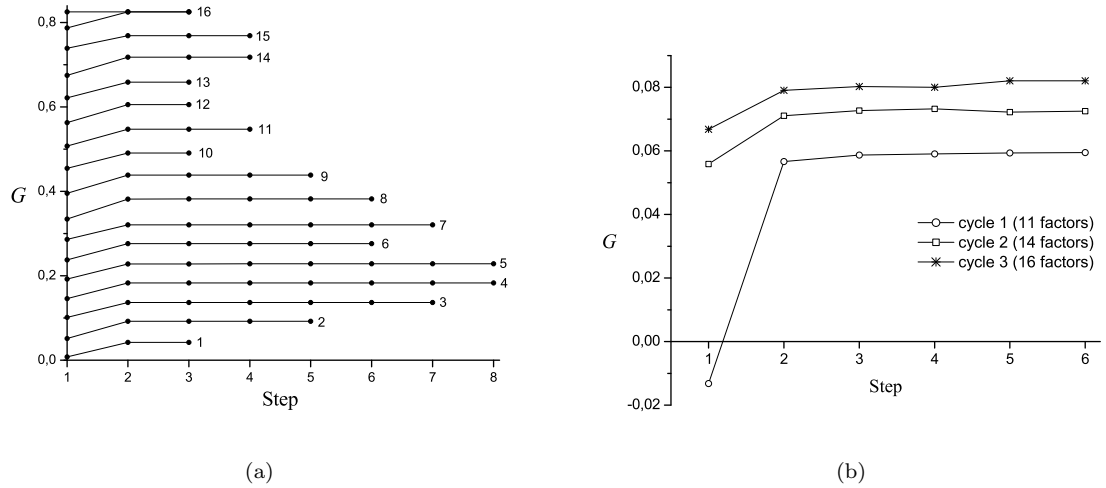
1. Signals of a data set are stored in the fully connected neural network according to Hebbian rule, forming the matrix of synaptic connections between neurons $\mathbf{J}'$ given by (5.1).

2. Two global attractors are excluded from the network dynamics by subtracting the matrix $\mathbf{J}''$ given by (5.2) from the matrix $\mathbf{J}'$.

3. 10 – 50 trajectories sequentially start from random states containing k_{in} active neurons and continue according to the two-run procedure until the number of active neurons reachs k_{fin}. The initial activity k_{in} is chosen to be lower and the final activity k_{fin} is chosen to be higher than the expected number of Ones in the factors. Usually we take $k_{in} = 5$ and $k_{fin} = 100$.

4. Only the trajectories satisfying the two following criteria are chosen from the obtained trajectories for a further analysis. First, they are smooth, according to the condition $Sim(k) \ge Sim_{thr} = 0.8$, where $Sim(k)$ is given by (5.5). Second, they are true, according to (5.8). The patterns of the network activity at peaks of R' at the chosen trajectories are treated as revealed factors. If neither trajectory satisfies both these criteria, LANNIA is terminated.

5. The probabilities p_{ij} for factors revealed at step 5 are estimated by (5.7). The probabilities π_i for these factors are set to be 0.5. The probabilities p_{ij} and π_i for factors revealed earlier and the probabilities q_j are taken from the output of the LM procedure at the previous LANNIA cycle. At



(a) (b)

Figure 5.3: Increase of information gain at each cycle of LANNIA in solving the Bars Problem when noise is absent ($p = 1$ and $q = 0$) (a) and present ($p = 0.7$ and $q = 0.2$) (b).

the first cycle of LANNIA only the probabilities p_{ij} estimated by (5.7) and $\pi_i = 0.5$ are used as the input to the LM procedure, and the seed probabilities q_j are taken to be zero.

6. The LM procedure alternates steps E and M until converges. The output of the procedure is a set of the generative model parameters and factor scores.

7. The information gain provided by found factors for this temporary BFA solution is calculated. If it is not higher than the gain obtained at the previous step, LANNIA is terminated. In the opposite case the parameters of the generative model obtained at step 6 are used for unlearning the new found factors from the neural network by the rule (5.6), and the procedure returns to step 3.

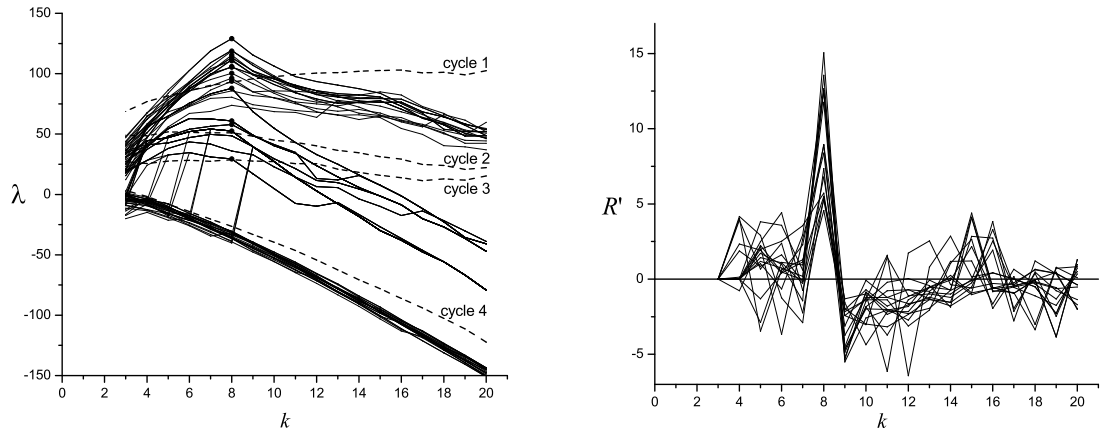
LANNIA algorithm is given in Appendix 2.

In the absence of noise all bars as the factors could be revealed during one cycle of LANNIA. However, for the reason of clarity we started only one trajectory at each its cycle and used it for further analysis. Thus, each trajectory shown in Fig. 5.2 was obtained after unlearning previously found factors. The order of their finding corresponds to the decrease of their Lyapunov function λ and threshold T.

Fig. 3(a) demonstrates the dependence of the information gain G on the number of found factors. For each set of found factors the gain is shown depending on the number of the iterative steps in LM. The maximal gain increase occurs at the first step. Since all bars are statistically equivalent, the finding of each new bar provides the almost equal increment of the gain. For the shown example LANNIA was terminated according to the step 5 because all true factors were deleted and all next trajectories happened to be spurious. The factors found by LANNIA are just the same bars as shown in Fig. 5.1(A).

The LANNIA performance for the case of noisy images ($p = 0.7$ and $q = 0.2$) is illustrated in Figs. 3(b) and 5.4. All factors were revealed for three full cycles. Twenty trajectories were run in ANNIA at the beginning of each LANNIA cycle. At the first cycle 11 trajectories were identified as true (Fig. 4(a)). These trajectories were continuous and had the values of the Lyapunov function at the peaks of R' (Fig. 4(b)) exceeding the separation border. The increment of the information gain provided by LM at the first cycle amounted to 0.06. LM converged for 4 steps. The largest increase of G occurred at the first step.

After the factors found were unlearned from ANNIA at the end of the first LANNIA cycle, ANNIA found three factors at the second cycle. The increment of the gain amounted to 0.015, that is proportional to the number of the found factors. At the third full LANNIA cycle, ANNIA found two factors and the gain increased for about 0.01. After unlearning the factors found at the third cycle, all trajectories at ANNIA happened to be spurious and the procedure was terminated.



(a) (b)

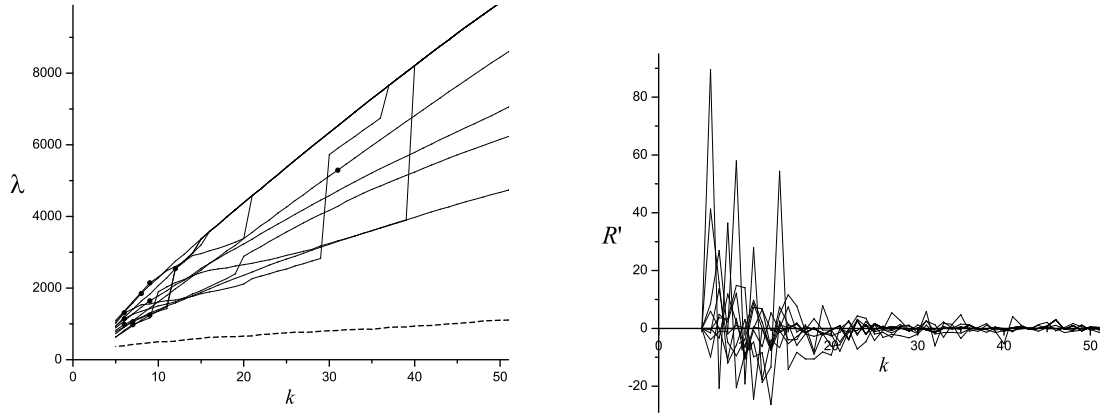
Figure 5.4: Lyapunov function λ (a) and function R' (b) in dependence on the number of active neurons k. Dashed lines in (a) are thresholds for separating true and spurious trajectories at four cycles of LANNIA. Results were obtained for data set consisting of $M = 800$ noisy patterns ($p = 0.7$, $q = 0.2$).

The factors found by LANNIA are shown in Fig. 5.1(D). They almost coincide with bars shown in Fig. 5.1(A) but contain also pixels with small probabilities p_{ij}. Note that LANNIA revealed correctly the hidden factor structure of the data set even when the structure is practically invisible (see Fig. 5.1(C)). For this case the "theoretical" gain calculated for the precise generative model parameters amounts to 0.061. The gain obtained by LANNIA was the same, that is, it provides almost precise solution of BP.

The role of ANNIA in the hybrid procedure is illustrated in Fig. 5.1(E). There are shown the factors found by LM alone when it starts from random initial scores, that is, without the approximate solution provided by ANNIA. The analyzed data set contained $M = 800$ not noisy images similar to that shown in Fig. 5.1(B). For each of 16 factors, the number of initial unitary scores was the same as in the data set but they were chosen randomly, i.e. independently of the presence of the relating factors in the images. LM found only 8 factors and consequently the gain amounts to only $G = 0.43$ that is almost twice smaller than "theoretical" gain $G = 0.82$ for the not noisy data. As shown in Fig. 3(a), the gain achieved in the hybrid LANNIA is equal to "theoretical" one, that is, it provides the perfect BP solution.

6 LANNIA application to the genome data set analysis

One of the important problems in modern biology is to identify functions of proteins in the organisms. The extensive experimental studies are required to identify the function of even a single protein. Therefore, even for well-studied model organisms, the functions of the most proteins are yet unknown [18]. A fast growing number of organisms with fully sequenced genomes makes it possible to reveal the protein function by comparing protein phylogenetic profiles of different organisms. The protein phylogenetic profile is defined as a binary pattern that encodes by Ones and Zeros the presence or the absence of proteins in a given organism with the fully sequenced genome, respectively [19]. When two proteins show the correlated events of the presence or absence over the organisms, it is assumed that these proteins are also functionally correlated. This idea is based on the observation that proteins seldom act as single isolated species to perform their functions. Usually a set of proteins is involved in each particular cellular process interacting in performing some function [20]. This leads to the concept of the modularity which assumes that the genome functionality can be partitioned into a collection



(a) (b)

Figure 6.1: Lyapunov function λ (a) and function R' (b) depending on the number of active neurons k at the first cycle of LANNIA during the analysis of the KEGG data set. Dashed line in (a) is a threshold for separating the true and spurious trajectories.

of modules. Each module is a discrete entity of elementary components and performs an identifiable task, separable from the functions of the other modules [21]. Thus, revealing sets of proteins which coherently appear in different organisms may facilitate the search for functional modules in the genome structure. Recently there were many attempts to reveal the modular structure in genome data sets by different blind statistical methods such as cluster analysis, independent component analysis and others (see [18] for review). Since the concept of the genome functional modularity is completely compatible with the BFA generative model described here it was a challenge for us to apply LANNIA to reveal the hidden factor structure in some large genome data set. We consider its BFA analysis only as an example of the LANNIA application to the large real data set and thus discuss only formal indexes of its performance such as the number of found factors, their relation to the protein phylogenetic profiles, and the information gain obtained which shows, particularly, the relevance of the BFA generative model to the genome data. The analysis of the factor contents and their relation to the known metabolic pathways is out of our competence and will not be discussed here.

For the BFA analysis we used the largest genome database KEGG [22], containing the fully sequenced genomes of $M = 1368$ organisms (January 2011). The protein phylogenetic profile of each organism is a binary pattern $\mathbf{x}_m$ of dimension $N = 11451$, where N is a whole number of proteins taken into account (specifically, gene/protein ortholog groups). This number was obtained after excluding duplicates from the whole set of 14139 gene/protein ortholog groups of KEGG.

LANNIA revealed 38 factors after four full cycles of LANNIA. Each cycle began by running twenty random trajectories in ANNIA. The Lyapunov functions along the eleven true trajectories at the first cycle of LANNIA are shown in Fig. 1(a). The peaks of R' used for the factor identification are shown in Fig. 1(b). During the first cycle the LM procedure converged for five steps and excluded two factors of eleven. The information gain provided after each LM step is shown in Fig. 6.2. At the fifth step it amounts to $G = 0.27$. At the second full LANNIA cycle ANNIA found fourteen factors. LM converged in four steps, excluding one factor and providing the gain increase up to $G = 0.31$ (Fig. 6.2). During the third and fourth full LANNIA cycles ANNIA found thirteen and twelve factors, LM excluded six and three, respectively. In each of the next cycle of LANNIA the growth of G was lower. During the fifth cycle the gain decreased and LANNIA was terminated. The maximal gain provided by LANNIA for the KEGG data set amounts to 0.32. The relatively high gain obtained shows that, first, the genome data indeed correspond to the generative BFA model and, second, LANNIA is the efficient method for finding its parameters. The high information gain is in favor of the hypothesis of the modular genome structure [21].

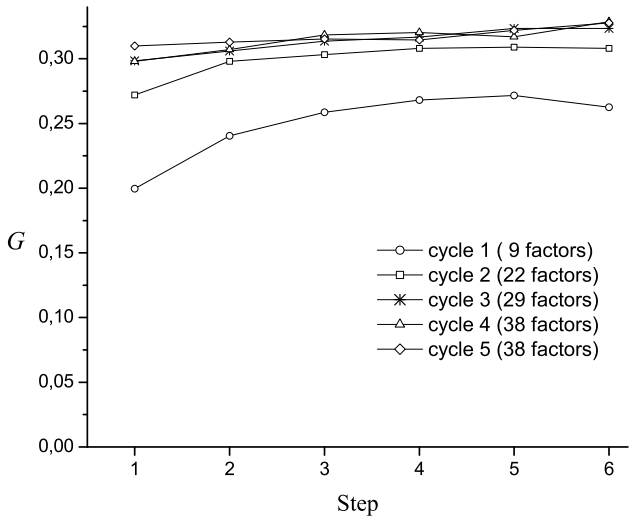


Figure 6.2: Increase of the information gain at each cycle of LANNIA during the analysis of the KEGG data set.

Fig. 6.3 demonstrates the distribution of proteins over found factors. We prescribed the protein j to the factor i if the probability p_{ij} exceeded a threshold p_{th}. The distributions are shown for three thresholds. The factors were ranged according to the decrease of the number of proteins constituting them for p_{th} = 0.5. On average one factor contains 235 proteins with $p_{ij} > 0.9$, 407 proteins with $p_{ij} > 0.7$ and 598 proteins with $p_{ij} > 0.5$. The number of proteins in the factors greatly exceeds the number of proteins in the metabolic pathways described in KEGG. Thus, it is unlikely that the factors correspond to metabolic pathways. Neither factor found by ANNIA contains more than 100 proteins. Thus, LM is able to enlarge the set of elements constituting a factor comparing with ANNIA significantly. As mentioned above, it is also able to eliminate some factors found by ANNIA. Thus, the results obtained actually represent the combined performance of ANNIA and LM, so the results truly represent the synergy of both methods.

Fig. 6.4 demonstrates the distribution of the factors over the organisms. All the organisms are grouped in types according to the taxonomy of KEGG from animals to bacteria. The type of the organisms is depicted by the number on the top of Fig. 6.4. The meaning of numbering is given in the legend. The factors are ranged in descending order according to the frequencies of their appearance in the data set. The factor number one appeared in the most organisms (in 22% of the organisms) and the factor number 38 appeared in the least of them (in 5% of the organisms). For the most factors the frequencies of their appearance in the organisms are distributed around 0.1. In Fig. 6.4 the appearance of a given factor in a given organism is marked by the point. Thus, the frequency of appearance of each factor in the data set corresponds to the number of points in each horizontal line. Fig. 6.4 demonstrates that each type of the organisms is characterized by the specific set of factors. For example, animals are characterized by factors 20 and 37, fungi by factor 20, plants by factors 2, 20 and 37 and so on. Factor 20 was identified only in eukaria and never in prokaria. Conversely, factor 1 was identified in all types of prokaria but never in eukaria. Thus, the distribution of factors over the types of organisms seems to reflect some peculiarities of their functioning. It is interesting that LANNIA revealed only little effect of specific factors: only 472 proteins over 11451 taken into account have q_j exceeding 0.01. Thus, almost all the organisms are completely described by common factors.

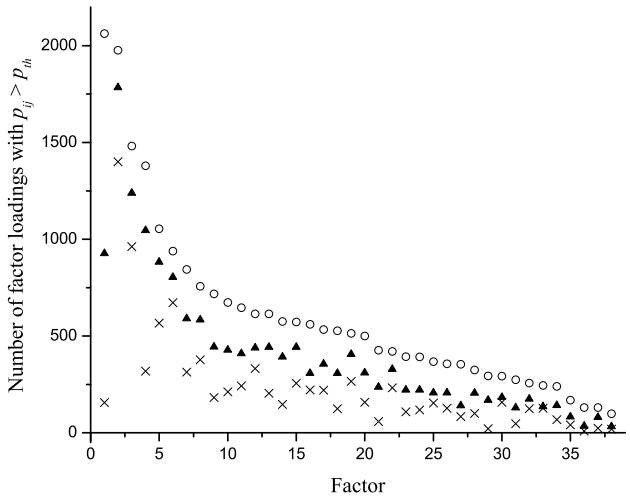


Figure 6.3: Distribution of number of proteins constituting factors found by LANNIA for three threshold values p_{th}, p_{th} = 0.5 - circles, p_{th} = 0.7 - triangles, p_{th} = 0.9 - crosses. Protein j was prescribed to factor i if $p_{ij} > p_{th}$. Factors are ranged according to decrease of the number of proteins for p_{th} = 0.5.

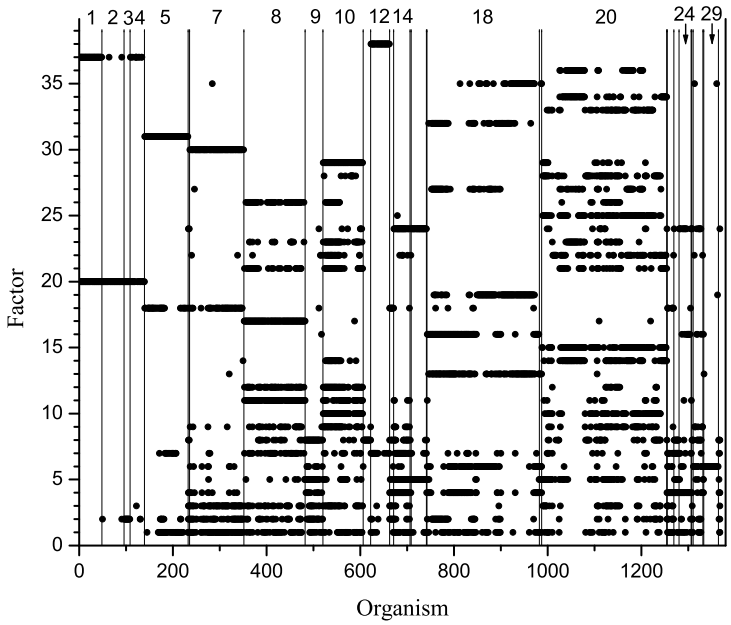


Figure 6.4: Distribution of factors over types of organisms. Eukaryotes: 1 – Animals, 2 – Fungi, 3 – Plants, 4 – Protists; Prokaryotes: 5 – Archaea; Bacteria: 6 – Acidobacteria, 7 – Actinobacteria, 8 – Alphaproteobacteria, 9 – Bacteroidetes, 10 – Betaproteobacteria, 11 – Chlamydiae, 12 – Cyanobacteria, 13 – Deinococcus-Thermus, 14 – Deltaproteobacteria, 15 – Elusimicrobia, 16 – Epsilonproteobacteria, 17 – Fibrobacteres, 18 – Firmicutes, 19 – Fusobacteria, 20 – Gammaproteobacteria, 21 – Gemmatimonadetes, 22 – Green nonsulfur bacteria, 23 – Green sulfur bacteria, 24 – Hyperthermophilic bacteria, 25 – Other proteobacteria, 26 – Planctomycetes, 27 – Spirochaetes, 28 – Synergistetes, 29 – Tenericutes, 30 – Verrucomicrobia.

7 Discussion

The proposed BFA generative model of binary signals follows the general idea that the external world is organized regularly and the typical form of such regularity is the existence of objects characterized by a set of highly coherent attributes. To create notions of objects, the brain has to calculate the statistics of incoming signals in order to characterize them by some representative variables, the number of which is much smaller than signal dimension. It is generally accepted that such a reduction of information redundancy of the incoming signals is one of the main functions of the brain [12, 23, 11, 14, 24]. We also believe that this kind of signal redundancy is typical for many fields such as social science, marketing, zoology, genetics, medicine and others that operate with nominal data.

We assume that expression (2.1) defining the BFA generative model provides a rather general form of binary signal representation. Most important here is the introduction of two kinds of noise: the distortion of common factors and a noise in the form of specific factors. The presence of specific factors is a typical assumption of linear factor analysis, whereas distortion of common factors is a peculiarity of BFA. For example, for textual data, a factor is some topic characterized by keywords related to factor loadings, and each factor score is defined by whether a given document is dedicated to the topic. Though each topic is represented by a set of keywords, there are no or few documents containing the whole set. Factor distortion means the absence of some keywords from a topic keyword list in a given document dedicated to the topic. Each specific factor relates to each individual word. It is characterized by the probability of the related word to be present in the document independently of topics. In principle, the description of the generative model could be made more homogeneous if we introduce instead of the vector of specific factors, additional special common factor $\mathbf{f}_0$ with $f_{0j} = 1$ for all $j = 1, \ldots, N$ and $p_{0j} = q_j$ which appears in all observations, i.e. with $S_{0m} = 1$ for all $m = 1, \ldots, M$. Then our generative model would be identical with the model suggested in [2]. However this description of the generative model is incompatible with ANNIA which uses the assumption that factors are coded sparsely [5]. Thus, we prefer to describe the model in terms of common and specific factors similar to the notations of linear factor analysis.

Signals containing certain factors can be grouped. Since a signal can contain several factors, it can be related to several groups. In this aspect, BFA is close to fuzzy clustering. However, BFA provides an explicit knowledge explaining why the signal is shared between clusters. BFA efficiency for fuzzy clustering is demonstrated by Frolov et al. [8]. When noise is absent, BFA is equivalent to Boolean matrix factorization [25, 26]. In the BFA generative model scores are assumed to be nonnegative (1 or 0) so it could be related to the methods of Nonnegative Matrix Factorization [27]. On the other hand factor scores are assumed to be sparse (BFA evidently fails for dense factor scores [5]) which means that BFA could be related to the methods of Sparse Component Analysis [28]. What more, BFA could be also related to the methods of Independent Component Analysis [29] because factors are assumed to be independently distributed in the data set.

We have shown in [9] that the information gain is sensitive both to the noise in the data and to the errors in the BFA. When the noise increases (in the form of factor distortion or specific factors), the information gain decreases and becomes zero or negative. Looking at the bar images with small gain (as shown, for example, in Fig. 5.1(C)), one might agree that zero gain corresponds to the threshold when the hidden factor structure in the data set becomes invisible. The gain also decreases when some true factors are missing or factor scores are found incorrectly. Thus, it can be used for comparing different BFA methods. Note that gain is a measure of BFA efficiency that does not require any a priori knowledge concerning signal structure.

We have also shown here that hybrid LANNIA method is able to find the hidden factor structure in the artificial data set consisting even of rather noisy images when BFA provides almost zero gain. The artificial images we used here are the Boolean superpositions of vertical and horizontal bars. Revealing individual bars as entire objects in the data set containing their mixtures (so called Bars Problem, BP) was introduced by Foldiak [14] as a benchmark for BFA. As shown in our previous papers [5, 8], ANNIA itself is a good method to reveal the loadings of the factors in complex artificial or natural data sets. However, it fails in finding the factor scores. Thus, we proposed in [8] some less elaborate method to complement ANNIA by possibility of finding factor scores. Here we have developed the idea of the likelihood maximization and presented it in the form similar to expectation-maximization

method [15]. In this form the LM procedure consists of two intermittent steps M and E. Step M serves to optimize generative model parameters estimation and step E serves to optimize the distribution of factor scores over data set signals. Although LM is shown to be a rather effective procedure it requires a reasonable initial approximation which is provided by ANNIA. In turn, LM provides ANNIA by the generative model parameters to exclude the found factors from the ANNIA dynamics.

Since the hybrid LANNIA procedure occurs to be perfect in BFA even with the very noisy artificial data, it was a challenge for us to apply the method to a large set of natural data. As we have already mentioned,identifying functions of proteins in the organisms is an important problem in modern biology. A fast growing number of fully sequenced genomes makes it possible to reveal the protein function by comparing the protein phylogenetic profiles of different organisms. The protein phylogenetic profile is defined as a binary pattern that encodes by Ones and Zeros the presence or, respectively, the absence of proteins in a given organism with fully sequenced genomes [19]. When two proteins show correlated patterns of their presence or absence over the organisms, it is assumed that these proteins are also functionally correlated. This leads to the concept of modularity which assumes that the genome functionality can be partitioned into a collection of modules. Each module is a discrete entity of elementary components and performs an identifiable task, separable from the functions of other modules [21]. Thus, revealing sets of the proteins coherently appeared in different organisms may facilitate the search for the functional modules in the genome structure. Since the concept of the genome functional modularity is completely compatible with the BFA generative model and taking into account the importance of the problem of the protein function identification, we chose for LANNIA application the largest genome database KEGG [22], containing fully sequenced genomes of 1368 organisms. LANNIA revealed 38 factors which happened to be reasonably distributed over organisms so that each type of organisms contains the specific set of factors. The resulting high information gain $G = 0.32$, first, confirms the hypothesis of the genome modular structure and, second, the high efficiency of LANNIA in processing natural data. The analysis of factor contents and their relation to known metabolic pathways is out of our competence and is not discussed here. However, we believe that the results obtained will attract the attention of genome researchers to BFA methods and, particularly, to LANNIA.

8 Appendix 1

To clarify Equation (5.6) used for unlearning found factors from the network, let us consider the case when all patterns of the data set have equal number of active neurons, i.e. $a_m = const$ in (5.1). Then according to (5.2) and (5.3)

$$J_{ij} = \sum_{m=1}^{M} (x_{mi} - a_m)(x_{mj} - a_m) - M(p_i - \bar{p})(p_j - \bar{p}) = M(< x_i x_j > - < x_i >< x_j >), \tag{8.1}$$

where $< x_i x_j >$ is a frequency of the simultaneous activation of the i^{th} and the j^{th} neurons in the data set and $< x_i >$ and $< x_j >$ are frequencies of their individual activation.

To estimate the contribution of the k^{th} factor to the weight of synaptic connection between the i^{th} and the j^{th} neurons due to these 'suspicious coincidences', let us suppose that the neurons are independently activated by other factors. Since $< x_i x_j > - < x_i >< x_j >=< (1-x_i)(1-x_j) > - < 1-x_i >< 1-x_j >$, then according to (3.5) and (8.1)

$$\begin{aligned} < x_i x_j > &- < x_i >< x_j >= \pi_k(1-p_{ki})(1-p_{kj})(1-p^0_{ki})(1-p^0_{kj}) + (1-\pi_k)(1-p^0_{ki})(1-p^0_{kj}) \\ &- [\pi_k(1-p_{ki})(1-p^0_{ki}) + (1-\pi_k)(1-p^0_{ki})][\pi_k(1-p_{kj})(1-p^0_{kj}) + (1-\pi_k)(1-p^0_{kj})] \\ &= \pi_k(1-\pi_k)p_{ki}p_{kj}(1-p^0_{ki})(1-p^0_{kj}) \end{aligned}$$

that directly results in the unlearning rule (5.6).

9 Appendix 2

Algorithm 1: Hybrid algorithm of ANNIA and Likelihood Maximization for BFA

input : $\mathcal{X}$ is a set of M binary signals $\mathbf{x_m}$ of dimension N with components x_{mj}, $m = 1 \dots M$, $j = 1 \dots N$

output: $\mathbf{\Theta} = (p_{ij}, q_j, \pi_i, i = 1, \dots, L, j = 1, \dots, N)$ are parameters of generative model, $\mathcal{S}$ is a set of binary factor scores $\mathbf{s}_m$ of dimension L with components s_{ml}, $m = 1 \dots M$, $l = 1 \dots L$

begin

  for $m \leftarrow 1$ **to** M **do** $a_m \leftarrow \sum_{j=1}^{N} x_{mj}/N$

  for $i \leftarrow 1$ **to** N **do**

    for $j \leftarrow 1$ **to** N **do**

      $\tilde{J}_{ij} \leftarrow \sum_{m=1}^{M} (x_{mi} - a_m)(x_{mj} - a_m) - \frac{1}{M}\Big[\sum_{m=1}^{M} (x_{mi} - a_m)\Big]\Big[\sum_{m=1}^{M} (x_{mj} - a_m)\Big]$

    $\tilde{J}_{ii} \leftarrow 0$

  $G \leftarrow 0, \quad L \leftarrow 0$

  for $j \leftarrow 1$ **to** N **do** $q_j \leftarrow 1/M$

  repeat

    $G^{old} \leftarrow G$

    if $L = 0$ **then**

      $\mathbf{J} \leftarrow \tilde{\mathbf{J}}$

    else

      for $l \leftarrow 1$ **to** L **do**

        for $i \leftarrow 1$ **to** N **do** $p_{li}^0 \leftarrow \sum_{m=1}^{M} x_{mi}(1 - s_{ml}) / \sum_{m=1}^{M} (1 - s_{ml})$

        for $i \leftarrow 1$ **to** N **do**

          for $j \leftarrow 1$ **to** N **do**

            $J_{ij} \leftarrow \tilde{J}_{ij} - M\pi_l(1 - \pi_l)p_{li}(1 - p_{li}^0)p_{lj}(1 - p_{lj}^0)$

          $J_{ii} \leftarrow 0$

    $\mathcal{F} \leftarrow$FactorsRevealing($\mathbf{J}$)

    if $\mathcal{F} = \varnothing$ **then** break algorithm

    foreach $\mathbf{f} \in \mathcal{F}$ **do**

      $L \leftarrow L + 1$

      $\mathbf{h} \leftarrow \mathbf{Jf}$

      for $j \leftarrow 1$ **to** N **do**

        if $h_j > 0$ **then** $p_{ij} \leftarrow 0.99 h_j / \max_i(h_i)$ **else** $p_{ij} \leftarrow 0$

    $[\mathbf{\Theta}, \mathcal{S}] \leftarrow$LikelihoodMaximization($\mathcal{X}, \mathbf{\Theta}$)

    $G \leftarrow$InformationGain($\mathcal{X}, \mathbf{\Theta}, \mathcal{S}$)

  until $G < G^{old}$

Procedure FactorsRevealing($\mathbf{J}$)

input : matrix of synaptic connections $\mathbf{J}$ of dimensionality $N \times N$
output : set of found factors $\mathcal{F}$
parameters: n_{in} and n_{fin} are initial and final number of active neurons along each trajectory, K_{trial} is number of trials of factor search, θ_{ov} is minimal overlap between two network states subsequently appeared in neurodynamics that are considered to be in the same attractor basin

1 **begin**
2 $\mathcal{F} \leftarrow \varnothing$
3 **repeat** K_{trial} **times**
4 generate `perm`, a random permutation from 1 to N
5 **for** $i \leftarrow 1$ **to** N **do** $x_i \leftarrow 0$
6 **for** $i \leftarrow 1$ **to** n_{in} **do** $x_{\mathtt{perm}(i)} \leftarrow 1$
7 $\mathbf{f} \leftarrow \mathbf{0}, \quad R'_{max} \leftarrow 0$
8 **for** $k \leftarrow n_{in}$ **to** n_{fin} **do**
9 $\mathbf{x}^1 \leftarrow \mathbf{x}^2 \leftarrow \mathbf{0}$
10 **while** $\mathbf{x} \neq \mathbf{x}^1$ **and** $\mathbf{x} \neq \mathbf{x}^2$ **do**
11 $\mathbf{x}^2 \leftarrow \mathbf{x}^1$
12 $\mathbf{x}^1 \leftarrow \mathbf{x}$
13 $\mathbf{h} \leftarrow \mathbf{J}\mathbf{x}$ `// vector of synaptic excitation`
14 $\mathbf{x} \leftarrow \mathbf{0}$
15 **for** $i \leftarrow 1$ **to** k **do**
16 $j \leftarrow \mathtt{argmax}_l(h_l)$
17 $x_j \leftarrow 1, \quad h_j \leftarrow \min_l(h_l)$
18 $\lambda \leftarrow \mathbf{x}^T\mathbf{J}\mathbf{x}^1/k, \quad T \leftarrow \max_l(h_l)$
19 $R \leftarrow \lambda/(k-1) - T/k$
20 **if** $k > k_{in}$ **and** $\max_{i,j=1,2}(Overlap(\mathbf{x}^i, \mathbf{x}^j_{prev})) > \theta_{ov}$ **then**
21 $R' \leftarrow R - R_{prev}$
22 **if** $R' > R'_{max}$ **then**
23 $R'_{max} \leftarrow R', \quad \mathbf{f} \leftarrow \mathbf{x}, \quad \lambda_f \leftarrow \lambda, \quad n_f \leftarrow k$
24 $R_{prev} \leftarrow R, \quad \mathbf{x}^1_{prev} \leftarrow \mathbf{x}^1, \quad \mathbf{x}^2_{prev} \leftarrow \mathbf{x}^2$
25 $x_{\mathtt{argmax(h)}} \leftarrow 1$ `// activating non-active neuron with maximal synaptic excitation`
26 **if** $\mathbf{f} = \mathbf{0}$ **or** $\mathbf{f} \in \mathcal{F}$ **then goto** line 3
27 $h_{thr} \leftarrow$SpurTrueThreshold$(n_f, \mathbf{J})$ `// threshold separating true and spurious attractors`
28 **if** $\lambda_f > h_{thr}$ **then** add $\mathbf{f}$ to $\mathcal{F}$

Function Overlap($\mathbf{x}, \mathbf{y}$)

input : binary vectors $\mathbf{x}$ and $\mathbf{y}$ of dimensionality N so that $|\mathbf{x}| \leq |\mathbf{y}|$ where $|\mathbf{x}|$ is the number of entries with value equal One in $\mathbf{x}$
output: overlap Ov between $\mathbf{x}$ and $\mathbf{y}$

1 **begin**
2 $Ov \leftarrow \dfrac{1}{|\mathbf{x}|(1-|\mathbf{y}|/N)} \sum_{i=1}^{N}(x_i - |\mathbf{x}|/N)(y_i - |\mathbf{y}|/N)$

Function SpurTrueThreshold(n, $\mathbf{J}$)

input : n number of entries with value equal One, synaptic connection matrix $\mathbf{J}$ of dimensionality $N \times N$

output : threshold h_{thr} separating true and spurious attractors

parameters: k_σ is standard deviation multiplier in calculation of maximal possible value of the Lyapunov function for spurious attractors

begin
  for $k \leftarrow 1$ **to** 100 **do**
    generate `perm`, a random permutation from 1 to N
    for $i \leftarrow 1$ **to** N **do** $x_i \leftarrow 0$
    for $i \leftarrow 1$ **to** n **do** $x_{\texttt{perm}(i)} \leftarrow 1$
    $\mathbf{h} \leftarrow \mathbf{Jx}$
    $h_k^{max} \leftarrow \max_i(h_i)$
  $h_{thr} \leftarrow m + k_\sigma\sigma$ where m and σ are mean and standard deviation of h_k^{max}

Function InformationGain($\mathcal{X}, \mathbf{\Theta}, \mathcal{S}$)

input : $\mathcal{X}$ is set of signals with components x_{mj}, $m = 1 \dots M$, $j = 1 \dots N$, $\mathbf{\Theta} = (p_{ij}, q_j, \pi_i, i = 1, \dots, L, j = 1, \dots, N)$ are generative model parameters, $\mathcal{S}$ is set of factor scores with components s_{ml}, $m = 1 \dots M$, $l = 1 \dots L$

output: relative information gain G for given $\mathcal{X}$, $\mathbf{\Theta}$ and $\mathcal{S}$

begin
  define the Shannon function $\texttt{h}(x) = -x\log_2 x - (1-x)\log_2(1-x)$
  for $j \leftarrow 1$ **to** N **do** $p_j \leftarrow \sum_{m=1}^{M} x_{mj}/M$
  $H_0 \leftarrow M \sum_{j=1}^{N} \texttt{h}(p_j)$
  $H_1 \leftarrow M \sum_{i=1}^{L} \texttt{h}(\pi_i)$
  for $m \leftarrow 1$ **to** M **do**
    for $j \leftarrow 1$ **to** N **do**
      $P(x_{mj}|\mathbf{s}_m, \mathbf{\Theta}) \leftarrow x_{mj} - (2x_{mj} - 1)(1 - q_j) \prod_{i=1}^{L} (1 - p_{ij})^{s_{mi}}$
  $H_2 \leftarrow \sum_{m=1}^{M} \sum_{j=1}^{N} \texttt{h}(P(x_{mj}|\mathbf{s}_m, \mathbf{\Theta}))$
  $G \leftarrow \dfrac{H_0 - H_1 - H_2}{H_0}$

Procedure LikelihoodMaximization($\mathcal{X}, \mathbf{\Theta}$)

input : $\mathcal{X}$ is set of signals with components x_{mj}, $m = 1 \dots M$, $j = 1 \dots N$,
$\mathbf{\Theta} = (p_{ij}, q_j, \pi_i, i = 1, \dots, L, j = 1, \dots, N)$ are generative model parameters
output: parameters of generative model $\mathbf{\Theta}$, set of factor scores $\mathcal{S}$ with components s_{ml}, $m = 1 \dots M$, $l = 1 \dots L$ (number of factors L is less or equal to those at the input of the algorithm)

1 **begin**
2 $\quad \mathcal{L} \leftarrow -\infty$
3 $\quad$ **repeat**
4 $\quad\quad \mathcal{L}^{prev} \leftarrow \mathcal{L}$
5 $\quad\quad \mathcal{S} \leftarrow$ProcEstep($\mathcal{X}, \mathbf{\Theta}$)
6 $\quad\quad$ **for** $i \leftarrow 1$ **to** L **do**
7 $\quad\quad\quad \pi_i \leftarrow \sum_{m=1}^{M} s_i^m / M$
8 $\quad\quad\quad$ **if** $\pi_i = 0$ **or** $\pi_i = 1$ **then**
9 $\quad\quad\quad\quad$ discard π_i, $\{p_{ij}\}$, $\{s_{mi}\}$ for all $j = 1, \dots, N$, $m = 1, \dots, M$
10 $\quad\quad\quad\quad L \leftarrow L - 1$
11 $\quad\quad\quad\quad$ **if** $L = 0$ **then** break procedure
12 $\quad\quad \mathbf{\Theta} \leftarrow$ProcMstep($\mathcal{X}, \mathbf{\Theta}, \mathcal{S}$)
13 $\quad\quad$ **for** $i \leftarrow 1$ **to** L **do**
14 $\quad\quad\quad$ **if** $\sum_{j=1}^{N} \mathtt{sgn}(p_{ij}) = 0$ **then**
15 $\quad\quad\quad\quad$ discard π_i, $\{p_{ij}\}$, $\{s_{mi}\}$ for all $j = 1, \dots, N$, $m = 1, \dots, M$
16 $\quad\quad\quad\quad L \leftarrow L - 1$
17 $\quad\quad\quad\quad$ **if** $L = 0$ **then** break procedure
18 $\quad\quad \mathcal{L} \leftarrow \sum_{m=1}^{M} \Big[\sum_{j=1}^{N} \log P(x_{mj} | \mathbf{s}_m, \mathbf{\Theta}) + \log P(\mathbf{s}_m | \mathbf{\Theta}) \Big]$
19 $\quad$ **until** $\mathcal{L} - \mathcal{L}^{prev} < 10^{-6} MN$

Procedure ProcEstep($\mathcal{X}, \mathbf{\Theta}$)

input : $\mathcal{X}$ is set of signals with components x_{mj}, $m = 1 \dots M$, $j = 1 \dots N$,
$\mathbf{\Theta} = (p_{ij}, q_j, \pi_i, i = 1, \dots, L, j = 1, \dots, N)$ are generative model parameters
output: set of factor scores $\mathcal{S}$ with components s_{ml}, $m = 1 \dots M$, $l = 1 \dots L$

1 **begin**
2 $\quad \mathcal{S} \leftarrow \varnothing$
3 $\quad$ **for** $m \leftarrow 1$ **to** M **do**
4 $\quad\quad$ **for** $i \leftarrow 1$ **to** L **do** $s_i \leftarrow 0$
5 $\quad\quad$ **if** $\mathbf{x}_m = \mathbf{0}$ **then goto** line 15
6 $\quad\quad$ **repeat**
7 $\quad\quad\quad \mathbf{s}^{old} \leftarrow \mathbf{s}$
8 $\quad\quad\quad$ generate `perm`, a random permutation from 1 to L
9 $\quad\quad\quad$ **for** $l \leftarrow 1$ **to** L **do**
10 $\quad\quad\quad\quad i \leftarrow \mathtt{perm}(l)$
11 $\quad\quad\quad\quad$ **for** $j \leftarrow 1$ **to** N **do** $P_j \leftarrow (1 - q_j) \prod_{k \neq i} (1 - p_{kj})^{s_k}$
12 $\quad\quad\quad\quad \Delta I \leftarrow \sum_{j=1}^{N} \Big[x_{mj} \log \frac{1 - (1 - p_{ij}) P_j}{1 - P_j} + (1 - x_{mj}) \log(1 - p_{ij}) \Big] + \log \frac{\pi_i}{1 - \pi_i}$
13 $\quad\quad\quad\quad$ **if** $\Delta I > 0$ **then** $s_i \leftarrow 1$ **else** $s_i \leftarrow 0$
14 $\quad\quad$ **until** $\mathbf{s} = \mathbf{s}^{old}$
15 $\quad\quad$ add $\mathbf{s}$ to $\mathcal{S}$

Procedure ProcMstep($\mathcal{X}, \mathbf{\Theta}, \mathcal{S}$)

input : $\mathcal{X}$ is a set of signals with components x_{mj}, $m = 1 \dots M$, $j = 1 \dots N$, $\mathbf{\Theta} = (p_{ij}, q_j, \pi_i, i = 1, \dots, L, j = 1, \dots, N)$ are generative model parameters, $\mathcal{S}$ is a set of factor scores with components s_{ml}, $m = 1 \dots M$, $l = 1 \dots L$

output: optimized parameters $\mathbf{\Theta}$ of the generative model

begin
- define ξ, a small positive number restricting p_{ij} and q_j for the sake of avoiding singularities
- **for** $j \leftarrow 1$ **to** N **do**
  - **repeat**
    - **for** $i \leftarrow 1$ **to** L **do** $p_{ij}^{prev} \leftarrow p_{ij}$
    - **for** $m \leftarrow 1$ **to** M **do** $P_m \leftarrow 1 - (1 - q_j) \prod_{i=1}^{L} (1 - p_{ij})^{s_{mi}}$
    - **for** $i \leftarrow 1$ **to** L **do**
      - $\gamma_p \leftarrow p_{ij}(1 - p_{ij})/(M\pi_i)$
      - $p_{ij} \leftarrow p_{ij} + \gamma_p \frac{1}{1 - p_{ij}} \Big(\sum_{m=1}^{M} \frac{s_{mi} x_{mj}}{P_m} - M\pi_i \Big)$
      - constraint p_{ij} to belong to $[0; 1 - \xi]$
      - **if** $p_{ij} < 1 - \prod_{l \neq i} (1 - \pi_l p_{lj})$ **then** $p_{ij} \leftarrow 0$
    - $\gamma_q \leftarrow q_j(1 - q_j)/M$
    - $q_j \leftarrow q_j + \gamma_q \frac{1}{1 - q_j} \Big(\sum_{m=1}^{M} \frac{x_{mj}}{P_m} - M \Big)$
    - constraint q_j to belong to $[\xi; 1]$
  - **until** $\sum_{i=1}^{L} |p_{ij} - p_{ij}^{prev}| < 10^{-3} L$

Algorithm 2: Observation Generation

input : M is a number of observations, N is a dimensionality of the observations, L is a number of factors, $\mathbf{\Theta} = (p_{ij}, q_j, \pi_i, i = 1, \dots, L, j = 1, \dots, N)$ are parameters of generative model

output: $\mathcal{X}$ is a set of M binary signals of dimension N, $\mathcal{S}$ is a set of M binary factor scores of dimension L

begin
- **for** $m \leftarrow 1$ **to** M **do**
  - Initialize $\mathbf{x}$ as N dimensional zero vector
  - Initialize $\mathbf{s}$ as L dimensional zero vector
  - **for** $i \leftarrow 1$ **to** L **do**
    - Generate r, a pseudo-random number from $[0, 1]$
    - **if** $r < \pi_i$ **then**
      - $s_i \leftarrow 1$
      - **for** $j \leftarrow 1$ **to** N **do**
        - Generate r, a pseudo-random number from $[0, 1]$
        - **if** $r < p_{ij}$ **then** $x_j \leftarrow x_j \vee 1$
  - **for** $j \leftarrow 1$ **to** N **do**
    - Generate r, a pseudo-random number from $[0, 1]$
    - **if** $r < q_j$ **then** $x_j \leftarrow x_j \vee 1$
  - Add $\mathbf{s}$ to $\mathcal{S}$
  - Add $\mathbf{x}$ to $\mathcal{X}$

Bibliography

[1] J. Lücke and M. Sahani, "Maximal causes for non-linear component extraction," *The Journal of Machine Learning Research*, vol. 9, pp. 1227–1267, 2008.

[2] E. Saund, "A multiple cause mixture model for unsupervised learning," *Neural Computation*, vol. 7, no. 1, pp. 0899–7667, 1995.

[3] A. C. Weber and C. Scharfetter, "The syndrome concept: history and statistical operationalizations." *Psychol Med*, vol. 14, no. 2, pp. 315–325, 1984.

[4] H. O. Veiel, "Psychopathology and Boolean Factor Analysis: a mismatch." *Psychol Med*, vol. 15, no. 3, pp. 623–628, 1985.

[5] A. A. Frolov, D. Husek, I. P. Muraviev, and P. Y. Polyakov, "Boolean factor analysis by attractor neural network," *IEEE Transactions on Neural Networks*, vol. 18, no. 3, pp. 698–707, 2007.

[6] A. A. Frolov, D. Húsek, H. Rezanková, V. Snásel, and P. Polyakov, "Clustering variables by classical approaches and neural network Boolean factor analysis," in *IEEE International Joint Conference on Neural Networks*, 2008, pp. 3742–3746.

[7] A. A. Frolov, D. Husek, P. Polyakov, and H. Rezankova, "New Neural Network Based Approach Helps to Discover Hidden Russian Parliament Voting Patterns," in *IEEE International Joint Conference on Neural Networks*, 2006, pp. 6518–6523.

[8] A. A. Frolov, D. Husek, and P. Y. Polyakov, "Recurrent neural network based Boolean factor analysis and its application to automatic terms and documents categorization," *IEEE Transactions on Neural Networks*, vol. 20, no. 7, pp. 1073–1086, 2009.

[9] A. Frolov, D. Husek, and P. Polyakov, "Estimation of Boolean factor analysis performance by informational gain," in *Proceedings of the 6th Atlantic Web Intelligence Conference (AWIC'2009)*, Praha, Czech Republic, Sep. 2009, p. in press.

[10] "KEGG: Kyoto Encyclopedia of Genes and Genomes." [Online]. Available: http://www.genome.jp/kegg

[11] H. B. Barlow, "Cerebral cortex as model builder," in *Models of the visual cortex*, D. Rose and V. G. Dodson, Eds. Chichester: Wiley, 1985, pp. 37–46.

[12] D. Marr, "A Theory for Cerebral Neocortex," *Proceedings of the Royal Society of London. Series B, Biological Sciences (1934-1990)*, vol. 176, no. 1043, pp. 161–234, 1970.

[13] E. M. Kussul, *Associative neuron-like structures.* Kiev: Naukova Dumka, 1992.

[14] P. Foldiak, "Forming sparse representations by local anti-hebbian learning," *Biological Cybernetics*, vol. 64, p. 165170, 1990.

[15] A. P. Dempster, N. M. Laird, and D. B. Rubin, "Maximum likelihood from incomplete data via the EM algorithm," *Journal of the Royal Statistical Society. Series B (Methodological)*, vol. 39, no. 1, pp. 1–38, 1977.

[16] A. A. Frolov, A. M. Sirota, D. Husek, I. P. Muraviev, and P. J. Polyakov, "Binary factorization in Hopfield-like neural networks: single-step approximation and computer simulations," *Neural Network Word*, vol. 14, pp. 139–152, 2004.

[17] A. A. Frolov, D. Husek, and P. Y. Polyakov, "Origin and Elimination of Two Global Spurious Attractors in Hopfield-like Neural Network Performing Boolean Factor Analysis," *Neurocomputing*, p. in press, 2010.

[18] P. Kensche, V. Van Noort, B. Dutilh, and M. Huynen, "Practical and theoretical advances in predicting the function of a protein by its phylogenetic distribution," *Journal of the Royal Society Interface*, vol. 5, no. 19, p. 151, 2008.

[19] M. Pellegrini, E. Marcotte, M. Thompson, D. Eisenberg, and T. Yeates, "Assigning protein functions by comparative genome analysis: protein phylogenetic profiles," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 96, no. 8, p. 4285, 1999.

[20] C. Von Mering, R. Krause, B. Snel, M. Cornell, S. Oliver, S. Fields, and P. Bork, "Comparative assessment of large-scale data sets of protein–protein interactions," *Nature*, vol. 417, no. 6887, pp. 399–403, 2002.

[21] E. Ravasz, A. Somera, D. Mongru, Z. Oltvai, and A. Barabási, "Hierarchical organization of modularity in metabolic networks," *Science*, vol. 297, no. 5586, p. 1551, 2002.

[22] M. Kanehisa, S. Goto, S. Kawashima, and A. Nakaya, "The KEGG databases at GenomeNet," *Nucleic acids research*, vol. 30, no. 1, p. 42, 2002.

[23] D. Marr, "Simple Memory: A Theory for Archicortex," *Philosophical Transactions of the Royal Society of London. Series B, Biological Sciences (1934-1990)*, vol. 262, no. 841, pp. 23–81, 1971.

[24] K. Doya, "What are the computations of the cerebellum, the basal ganglia and the cerebral cortex?" *Neural networks*, vol. 12, no. 7-8, pp. 961–974, 1999.

[25] R. Belohlavek and V. Vychodil, "Discovery of optimal factors in binary data via a novel method of matrix decomposition," *Journal of Computer and System Sciences*, vol. 76, no. 1, pp. 3–20, 2010.

[26] V. Snasel, J. Platos, P. Kromer, D. Husek, and A. Frolov, "On the road to genetic Boolean matrix factorization," *Neural Network World*, vol. 17, no. 6, p. 675, 2007.

[27] S. Zafeiriou, A. Tefas, I. Bucie, and I. Pitas, "Exploiting discriminant information in nonnegative matrix factorization with application to frontal face verification," *IEEE Transactions on Neural Networks*, vol. 17, no. 3, pp. 683–695, 2006.

[28] P. Georgiev, F. Theis, and A. Cichocki, "Sparse component analysis blind sourse separation of underdetermined mixtures," *IEEE Transactions on Neural Networks*, vol. 16, no. 4, pp. 992–996, 2005.

[29] Z. Koldovsky, P. Ticharsky, and E. Oja, "Efficient variant of algorithm fast ica for independent component analysis attaining the cramer-rao lower bound," *IEEE Transactions on Neural Networks*, vol. 17, no. 5, pp. 1265–1277, 2006.