Analysis and Evaluation of Different Methods for Barr Problem Solving

Húsek, Dušan
2010

Institute of Computer Science
Academy of Sciences of the Czech Republic

Analysis and Evaluation of Different Methods for Barr Problem Solving

Húsek Dušan, Frolov Alexander, Polyakov Pavel

Technical report No. 1082

Leden 2010

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

Analysis and Evaluation of Different Methods for Barr Problem Solving [1]

Húsek Dušan, Frolov Alexander, Polyakov Pavel

Technical report No. 1082

Leden 2010

Abstract:

Methods for hidden structure of high-dimensional binary data discovery are one of the most important challenges facing machine learning community researchers. There are many approaches in literature that try to solve this hitherto rather ill defined task. To elaborate our approach we propose a most general generative model of binary data for Boolean factor analysis. Our previous finding has shown that recurrent neural network is able to learn hidden factors as separate attractors. Proceeding this, we introduce here new Attractor Neural Network with Increasing Activity, acting as a Boolean Factor Analyzer. Then we introduce new Expectation-Maximization Boolean Factor Analysis algorithm based on a generative data model which maximizes likelihood of Boolean Factor Analysis solution. To show maturity of our solutions we propose here informational measure of Boolean Factor Analysis efficiency. Using the so-called bars problem benchmark , we compare efficiencies of our two BFA methods with Dendritic Inhibition neural network (DI), Maximal Causes Analysis and Boolean Matrix Factorization Then we discuss why the methods proposed by us outperform the rest of methods in the full range of Boolean factor analysis model parameters space.

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

[1] **Acknowledgement:** We wish to thank to Mr. Rachkovsky for a careful reading of the whole work and for stimulating comments, which helped us to improve both the clarity and quality of the work. We would also like to thank the institutions that provide financing of our projects codenamed AV0Z10300504, GACR P202/10/0262, GACR 205/09/1079, and GA MŠk(CZ) 1M0567.

1 Introduction

Factor analysis in general is one of the most efficient methods to reveal and to overcome informational redundancy of high-dimensional signals. Boolean Factor Analysis (BFA) as a special case of factor analysis implies that components of original signals, factor loadings and factor scores are binary values. Each binary signal (vector) can be interpreted as a representation of the appearance or the non-appearance of binary attributes (binary vector components) in the pattern. The number of considered attributes is the dimensionality of the signal space, the appearance of an attribute is encoded as One, and absence as Zero. The patterns are assumed to be composed of many "objects" in different combinations. 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, the 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). Correlation between the attributes constituting each factor can be revealed by statistics over 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. Factor may also be interpreted as some hidden cause resulting in the sets of observations [Lücke and Sahani (2008)] . For example in medical researches, a cause is a syndrome and an observation is a symptom [Weber and Scharfetter (1984), Veiel (1985)].

In spite of the fact that binary data representation is typical for many fields, including social science, marketing, zoology, genetics and medicine, BFA methods are rather moderately developed. We [Frolov et al. (2007)] proposed a BFA method that is based on the Hopfield-like attractor neural network. This new 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 network dynamics by tightly connected neurons. Since neurons representing a factor are activated simultaneously each time when 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 - factor neurons become more tightly connected than other neurons. So factors can be revealed as attractors of network dynamics. In our previous papers [Frolov et al. (2008), Frolov et al. (2006c), Frolov et al. (2009)] we demonstrated the efficiency of the method solving the task of mushroom data set clustering, in analysis of parliament voting, and in text analysis.

The well-known benchmark for learning of objects from complex patterns is the Bars Problem (BP) introduced by [Foldiak (1990)]. The BP in various modifications has been considered in many papers for references see [Lücke and Sahani (2008)]. In this problem, each pattern of the data set is n-by-n binary pixel image containing several of $L = 2n$ possible (one-pixel wide) horizontal and vertical bars (Fig. 1.1). Pixels belonging and not belonging to the bar take values 1 and 0, respectively. For each image each bar could be chosen with a probability C/L, where C is the mean number of bars mixed in an image. In the point of intersection of vertical and horizontal bars, pixel takes the value 1. The 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 $n \leq 8$.

In terms of BFA, bars are factors, each image is Boolean superposition of factors, and factor scores take values 1 or 0 dependently on bar presence or absence in the image. Thus, the bars problem is a special case of BFA. It was a challenge for us to test our ANNIA method using BP benchmark, because it is a demonstrative and well investigated example of complex image analysis [Spratling (2006), Lücke and Sahani (2008)].

To be able to evaluate performance of any BFA method we suggest, first, a general generative model of signals appropriate for BFA and, second, a general measure of BFA efficiency based on the comparison of data set entropies when its hidden structure is ignored or taken into account. This difference of the entropies, that is information gain, we suggest to consider as a general information theoretic measure of BFA efficiency. Next, we show that this measure is sensitive to both noise in signals and errors in BFA results. This analysis allows us to conclude that information gain is a

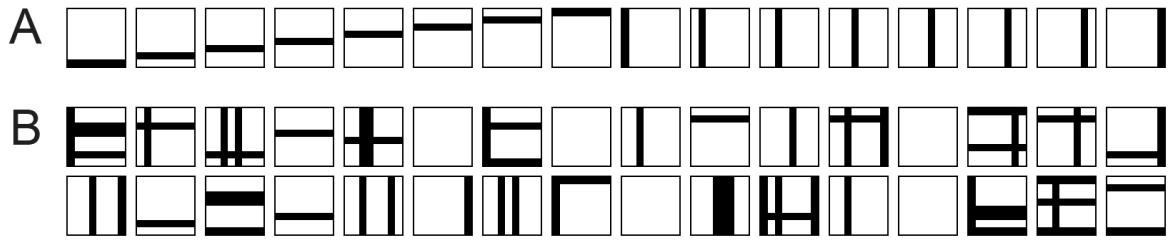


Figure 1.1: **A** Sixteen vertical and horizontal bars in 8-by-8 pixel images. **B** Examples of images in the standard bars problem. Each image contains two bars on average.

reliable basis for different BFA methods comparison and for detecting the presence of hidden factor structure in a given data set as well.

In the experimental part of the paper, we compare ANNIA with four other methods supposed to be most efficient in solving the bars problem. The first method is the Dendritic Inhibition neural network (DI). This method was suggested and studied by [Spratling and Johnson (2002)], [Spratling and Johnson (2003)], [Spratling (2006)] and was shown to be more efficient than other related methods.

The second method, suggested in [Lücke and Sahani (2008)], is Maximal Causes (MCA_3) method based on powerful and popular Expectation-Maximization (EM) algorithm [Dempster et al. (1977), Neal and Hinton (1998)]. It was shown [Lücke and Sahani (2008)] that it is even more efficient than DI . The third method, suggested in [Belohlavek and Vychodil (2010)], is fast Boolean Matrix Factorization (BMF). The last method, Expectation-Maximization Binary Factor Analysis (EMBFA), developed in this paper, is based on EM method, as well as MCA_3, however EMBFA is rigorously built on here suggested BFA generative model, whereas MCA_3 is based on different assumption about the data structure.

The paper is organized as follows. A general generative model is proposed in Section 2. In Section 3, we propose the procedure for information gain calculation. In Section 5, we provide important details of ANNIA. Related methods, including the development of EMBFA, are discussed in Section 6. The sensitivity of information gain to signal noise and to errors in the BFA results is investigated in Section 4. Abilities of different methods to solve the bars problem are compared in Section 7. The strengths and weaknesses of the considered BFA methods are discussed in Section 8.

2 A Generative Model of Signals Suitable for Boolean Factor Analysis

In formulating a generative model of signals suitable for BFA, we follow ideas of [Barlow (1985)], [Marr (1970)], [Kussul (1992)] and others who assumed that factor revealing is one of the main brain functions. Explaining Barlow ideas, [Foldiak (1990)] writes: "According to [Barlow (1985)] 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, i.e., 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 $\mathbf{X}$ with dimensionality N equal to 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 $\mathbf{f}_i$ is a binary row vector

of dimensionality N whose One valued entries correspond to highly correlated attributes of the i-th object. Although the probability of the object's attribute to appear in a pattern simultaneously with its other attributes is high, it is not obligatory equal to 1. For example, the attribute "has a tail" does not always appear with the appearance of the object "dog". We denote this probability as p_{ij}, where j is the index of an attribute and i is the index of a factor. For attributes constituting the factor, probability p_{ij} is high, and for other attributes it is zero.

As in linear factor analysis, we suppose that additionally to common factors $\mathbf{f}_i$, each signal also contains some specific factors. The contribution of specific factors is defined by a binary row vector η, which we call "specific noise". Each specific factor is characterized by a probability q_j that j-th component of vector η takes One.

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

$$\mathbf{X} = [\bigvee_{i=1}^{L} S_i \mathbf{f}_i'] \vee \eta, \tag{2.1}$$

where $\mathbf{S}$ is a binary row vector of factor scores of dimensionality L, L is the total number of factors, $\mathbf{f}_i'$ is a distorted version of factor $\mathbf{f}_i$ and η is a specific noise defining the influence of specific factors. Factor distortion implies that some Ones of the i-th factor become Zeros with probability $1 - p_{ij}$. We suppose that each component of the common factor is distorted independently of the presence of other factors in the pattern and independently of specific noise. Thus, the probability of the j-th component of $\mathbf{X}$ to take the value X_j is

$$\mathcal{P}(X_j) = X_j - (2X_j - 1)(1 - q_j) \prod_{i=1}^{L} (1 - p_{ij})^{S_i}, \tag{2.2}$$

where scores S_i are assumed to be given. We suppose that different components of $\mathbf{X}$ (attributes) are also statistically independent. Thus

$$\mathcal{P}(\mathbf{X}) = \prod_{j=1}^{N} \mathcal{P}(X_j). \tag{2.3}$$

BFA is performed on the set $\mathcal{X}$ of patterns $\mathbf{X}_m$ containing M representatives. We assume that factor distortion in each pattern of the data set does not depend on others. Thus

$$\mathcal{P}(\mathcal{X}) = \prod_{m=1}^{M} \mathcal{P}(\mathbf{X}_m). \tag{2.4}$$

In most of the examples considered in this paper, we assume that factors appear in patterns (i.e., related scores S_i take Ones) independently with probabilities π_i, $i = 1, \ldots, L$. In some examples we assume that the number of factors is fixed in each pattern. Then the appearances of different factors become slightly correlated.

The aim of Boolean Factor Analysis is to find the parameters of generative model $\mathbf{\Theta} = (p_{ij}, q_j, \pi_i, i = 1, \ldots, L, j = 1, \ldots, N)$ and the factor scores $S_{mi}, m = 1, \ldots, M$ for each of M patterns of the data set. However, it is supposed that the found factors could also be detected in any arbitrary pattern $\mathbf{X} \notin \mathcal{X}$ if generated by the same model.

3 Information Gain

If the factor structure of the signal space is not taken into account, 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 probability of the j-th component to take 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 and all factor loadings and scores are found, then representing the whole data set requires

$$H = H_1 + H_2 + H_3 \tag{3.2}$$

bits of information. Here first term

$$H_1 = N \sum_{i=1}^{L} h(r_i), \tag{3.3}$$

where r_i is the portion of Ones in i-th factor, is an information that is required for storing factor loadings, second term

$$H_2 = M \sum_{i=1}^{L} h(\pi_i) \tag{3.4}$$

is an information required for storing factor scores and third term

$$H_3 = \sum_{m=1}^{M} \sum_{j=1}^{N} h(\mathcal{P}(X_{mj})), \tag{3.5}$$

where $\mathcal{P}(X_{mj})$ is given by (2.2), is an information required for storing all patterns of the data set when factor loadings and scores are given.The information gain is defined as the difference between H_0 and H. Relative information gain is then

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

From a practical point of view, BFA is beneficial (in sense of dimensionality reduction) only if $G > 0$.

So let us optimize our formula in the light of this idea. The first term in (3.2) is proportional to LN, the second one to ML and the third one to MN. Since in all the cases considered in the sequel, $M \gg N \gg L$, then one could expect that the third term in (3.2) is always dominant. However, when noise is absent it vanishes and becomes dominant only when the level of noise is sufficiently high. At the same time, the first term in (3.2) is always small and hence could be neglected. However in principle it is essential to avoid the trivial solution of BFA, i.e.when each pattern of the data set is assumed to be individual factor. In this case, $H_3 = 0$, $H_2 = M^2 h(1/M) \simeq M \log_2(Me)$ and since H_0 is proportional to MN, this solution provides unreasonably high gain if H_1 is ignored. On other side, if it is taken into account, gain is negative since for this solution $H_1 = H_0$. Thus, the role of this term is to prevent the choice of BFA solution with the high gain but incredibly large number of factors. In all BFA methods considered below the growth of this number is restricted by their own means thus, we can exclude this term from the gain formula and in the sequel, when speaking about information gain, we will have the following formula

$$G = (H_0 - H_2 - H_3)/H_0, \tag{3.7}$$

in mind.

According to (3.4) and (3.5), to calculate the information gain one needs to know factor scores S_{mi} assigned to patterns of the data set and parameters of the generative model p_{ij} and q_j. However, this information is redundant because the knowledge of factor scores allows for calculation of generative model parameters and conversely the knowledge of these parameters allows for assigning proper factor scores to patterns of the data set. Since most of the methods considered in the sequel (except EMBFA developed in the present paper) do not use the notion of BFA generative model introduced here, hence they cannot provide the search of its parameters in principal but allow for searching for factor scores. That is why our comparison of their efficiencies has to be based on their abilities to assign proper factor scores to patterns of the data set. To find the parameters of BFA generative model required for information gain calculation we suggest the procedure based on maximization of the data set likelihood under given factor scores provided by any of considered BFA method.

For the BFA generative model the data set likelihood function takes the form

$$\mathcal{L} = \log \mathcal{P}(\mathcal{X}) = \sum_{m=1}^{M} \mathcal{L}_m, \tag{3.8}$$

where

$$\mathcal{L}_m = \log \mathcal{P}(\mathbf{X}_m) = \sum_{j=1}^{N} \log \mathcal{P}(X_{mj}), \tag{3.9}$$

and $\mathcal{P}(X_{mj})$ is given by (2.2). Maximum of $\mathcal{L}$ is defined by the following system of $L \times N + N$ equations for p_{ij} and q_j $(i = 1, \ldots, L,\ j = 1, \ldots, N)$:

$$\begin{aligned} \frac{\partial \mathcal{L}}{\partial p_{ij}} &= \sum_{m=1}^{M} \mathcal{P}(X_{mj})^{-1} \frac{\partial \mathcal{P}(X_{mj})}{\partial p_{ij}} = 0, \\ \frac{\partial \mathcal{L}}{\partial q_j} &= \sum_{m=1}^{M} \mathcal{P}(X_{mj})^{-1} \frac{\partial \mathcal{P}(X_{mj})}{\partial q_j} = 0, \end{aligned} \tag{3.10}$$

where

$$\begin{aligned} \frac{\partial P(X_{mj})}{\partial p_{ij}} &= \frac{S_{mi}(2X_{mj} - 1)(1 - q_j) \prod_{l=1,L} (1 - p_{lj})^{S_{ml}}}{1 - p_{ij}} \\ \frac{\partial P(X_{mj})}{\partial q_j} &= (2X_{mj} - 1) \prod_{l=1,L} (1 - p_{lj})^{S_{ml}}. \end{aligned} \tag{3.11}$$

Then (3.10) takes the form

$$\begin{aligned} \sum_{m=1}^{M} S_{mi} &= \sum_{m=1}^{M} \frac{S_{mi} X_{mj}}{1 - (1 - q_j) \prod_{l=1,L} (1 - p_{lj})^{S_{ml}}} \\ M &= \sum_{m=1}^{M} \frac{X_{mj}}{1 - (1 - q_j) \prod_{l=1,L} (1 - p_{lj})^{S_{ml}}} \end{aligned}$$

The obtained system can be solved by iterative procedure

$$\begin{aligned} p_{ij}(k+1) &= \frac{1}{\sum_{m=1}^{M} S_{mi}} \sum_{m=1}^{M} \frac{p_{ij}(k) S_{mi} X_{mj}}{1 - (1 - q_j(k)) \prod_{l=1,L} (1 - p_{lj}(k))^{S_{ml}}} \\ q_j(k+1) &= \frac{1}{M} \sum_{m=1}^{M} \frac{q_j(k) X_{mj}}{1 - (1 - q_j(k)) \prod_{l=1,L} (1 - p_{lj}(k))^{S_{ml}}}. \end{aligned} \tag{3.12}$$

Let us prove that this procedure converges: At its each step $\Delta p_{ij} = p_{ij}(k+1) - p_{ij}(k)$ is

$$\begin{aligned} \Delta p_{ij} &= \frac{p_{ij}(k)}{\sum_{m=1}^{M} S_{mi}} \sum_{m=1}^{M} \frac{S_{mi}(2X_{mj} - 1)(1 - q_j(k)) \prod_{l=1,L} (1 - p_{lj}(k))^{S_{ml}}}{X_{mj} - (2X_{mj} - 1)(1 - q_j(k)) \prod_{l=1,L} (1 - p_{lj}(k))^{S_{ml}}} \\ &= \frac{p_{ij}(k)(1 - p_{ij}(k))}{\sum_{m=1}^{M} S_{mi}} \sum_{m=1}^{M} \mathcal{P}(X_{mj})^{-1} \frac{\partial \mathcal{P}(X_{mj})}{\partial p_{ij}} = \frac{p_{ij}(k)(1 - p_{ij}(k))}{\sum_{m=1}^{M} S_{mi}} \frac{\partial \mathcal{L}}{\partial p_{ij}}. \end{aligned}$$

Similarly $\Delta q_j = q_j(k+1) - q_j(k)$ is

$$\begin{aligned} \Delta q_j &= \frac{q_j(k)}{M} \sum_{m=1}^{M} \frac{(2X_{mj} - 1)(1 - q_j(k)) \prod_{l=1,L} (1 - p_{lj}(k))^{S_{ml}}}{X_{mj} - (2X_{mj} - 1)(1 - q_j(k)) \prod_{l=1,L} (1 - p_{lj}(k))^{S_{ml}}} \\ &= \frac{q_j(k)(1 - q_j(k))}{M} \sum_{m=1}^{M} \mathcal{P}(X_{mj})^{-1} \frac{\partial \mathcal{P}(X_{mj})}{\partial q_j} = \frac{q_j(k)(1 - q_j(k))}{M} \frac{\partial \mathcal{L}}{\partial q_j}. \end{aligned}$$

Then

$$\Delta L \simeq \sum_{i,j} \frac{\partial \mathcal{L}}{\partial p_{ij}} \Delta p_{ij} + \sum_j \frac{\partial \mathcal{L}}{\partial q_j} \Delta q_j = \sum_{i,j} \frac{p_{ij}(k)(1-p_{ij}(k))}{\sum_{m=1}^{M} S_{mi}} \left(\frac{\partial \mathcal{L}}{\partial p_{ij}} \right)^2 + \sum_j \frac{q_j(k)(1-q_j(k))}{M} \left(\frac{\partial \mathcal{L}}{\partial q_j} \right)^2 .$$

Thus at each iteration step likelihood does not decrease and hence iteration procedure converges.

Since we assume that for attributes constituting a factor, probabilities p_{ij} are sufficiently high but equal to zero for other attributes, at each iteration step we set $p_{ij} = 0$, if p_{ij} is small. In particular, we set it to Zero, if

$$p_{ij} < 1 - \prod_{l \neq i} (1 - \pi_l p_{lj}), \tag{3.13}$$

where the right side of the inequality is the probability that the j-th attribute appears in the pattern due to other factors except $\mathbf{f}_i$. We consider such p_{ij}" cleaning" procedure as factor binarization, because we treat the component with a high probability p_{ij} as constituting the i-th factor ($f_{ij} = 1$), and the component with a small p_{ij} as not constituting it ($f_{ij} = 0$). According to our experience the iteration procedure converges in 3-5 steps.

As the input to the iterative procedure (3.12), we used p_{ij} and q_j obtained from probabilities p_{ij}^1 that the j-th attribute appears in the pattern when the i-th factor is present and p_{ij}^0 when it is absent. On one hand, probabilities p_{ij}^1 and p_{ij}^0 can be estimated as frequencies of the j-th attribute taking One in patterns of the data set containing and not containing the i-th factor. On the other hand, these probabilities can be estimated as

$$\begin{aligned} p_{ij}^0 &= 1 - (1 - q_j) \prod_{l \neq i} (1 - \pi_l p_{lj}) \\ p_{ij}^1 &= 1 - (1 - q_j)(1 - p_{ij}) \prod_{l \neq i} (1 - \pi_l p_{lj}) \end{aligned} \tag{3.14}$$

This results in

$$p_{ij} = (p_{ij}^1 - p_{ij}^0)/(1 - p_{ij}^0).$$

As in the previous procedure of likelihood maximization, we set the probability p_{ij} to zero, if it satisfies (3.13).

After finding p_{ij}, q_j can be obtained from (3.14). Probabilities π_i are estimated as the frequencies of the related scores provided by BFA. Probabilities p_j required for calculation of H_0 are estimated as frequencies of related components in the data set.

4 Relevance of Information Gain for the Bars Problem

In this section, we illustrate the general properties of the information gain G defined by (3.6), using the bars problem data. Particularly, we compare the values of G for

- "theoretical solution", i.e.for the case when all scores and generative model parameters are exactly the same as those used for data set generation,
- "ideal solution", i.e.for the case when all scores are exactly the same as those used in the generated data set, but parameters of the generative model are found by likelihood maximization (this case simulates the situation when BFA provides scores ideally matching those in the generated data set under analysis),
- "erroneous solution", i.e.for the case when some factors or scores are missed or false factors or scores are added.

Fig. 4.1 illustrates the dependence of information gain for "theoretical", "ideal" and "erroneous" solutions on the probabilities q_j and p_{ij}, and on the size of the data set M. Here, $p_{ij} = p$ for components

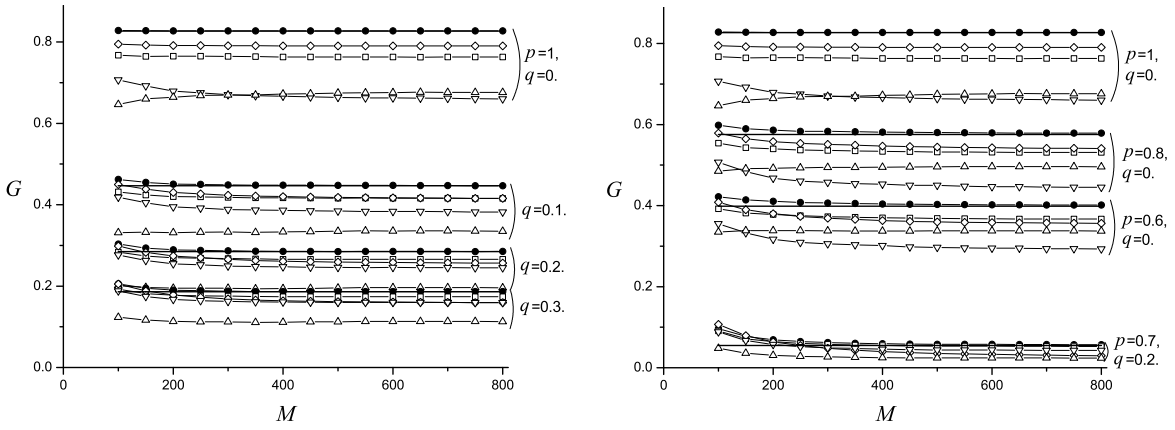


Figure 4.1: Information gain for "theoretical" (thick lines), "ideal" (thin lines marked by ●) and "erroneous" solutions in dependence on the size of the data set M, □ – one of the factors was excluded, ◇ – 16 false factors in the form of crossing bars were added, ▽ – 10 % of randomly chosen scores were excluded, △ – 10% of randomly chosen scores were added. (a) – dependence on q (specific noise) for $p = 1$, (b) – dependence on p (factors distortion) for $q = 0$ and for both kinds of noise ($q = 0.2$, $p = 0.7$).

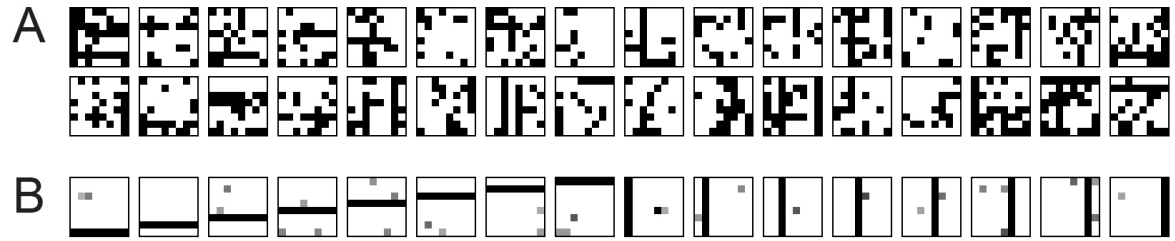


Figure 4.2: **A** Examples of noisy images for $p = 0.7$, $q = 0.2$. **B** Probabilities p_{ij} (shown by the shades of grey) of pixels activation obtained by the likelihood maximization for the ideal solution ($M = 100$, $p = 1$, $q = 0.3$) in one of the trials.

constituting factors and $q_j = q$ for any j. Recall that $p_{ij} = 0$ for components not constituting factors. In Fig. 4.1, and in subsequent figures, each shown value of G is obtained by averaging over 50 trials. Each trial is made, using randomly generated data set of a given size M. Patterns of the data set were 8-by-8 binary images (i.e., $N = 64$). $L = 16$ vertical and horizontal bars (one pixel width, Fig. 1.1(A)) were randomly mixed in images with probabilities $\pi_i = C/L = 1/8$, thus 2 bars were mixed in each image on average. Examples of the standard BP images ($p = 1$, $q = 0$) are shown in Fig. 1.1(B), and their noisy versions for $p = 0.7$, $q = 0.2$ are shown in Fig. 4.2(A).

For small M the information gain for "ideal" solution is paradoxically higher (Fig. 4.1) than for the "theoretical" one. But it is the usual case for the procedure of likelihood maximization: when data set is relatively small, the procedure provides the solution for p_{ij} and q_j that better fits randomly obtained peculiarities of a given data set realization than the "theoretical" solution. That is why both G and $\mathcal{L}$ are higher for the "ideal" solution adjusted to those specific peculiarities. Fig. 4.2(B) shows values of p_{ij} obtained for one of the trials with $M = 100$, $p = 1$ and $q = 0.3$. 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. For the pixels constituting bars all $p_{ij} = 1$. However, the factors found by likelihood maximization contain some additional pixels. For some of them, probability of their appearance with factors is rather high. It means that for this particular data set those pixels were activated by chance simultaneously with the activation of the related bar, and this peculiarity of the data set was detected by likelihood maximization. When M increases, this effect disappears and

"ideal" solution coincides with "theoretical" one.

As shown in Fig. 4.1, the maximal information gain is achieved when bars mixed in the scenes are not distorted ($p = 1$, $q = 0$), and G decreases when noise increases due to both increasing of q or decreasing of p. We suppose that when the information gain G is positive, BFA is appropriate for a given data set analysis, and on the other side when it is negative, BFA has no sense. The smaller is G, the less explicitly the factor structure of the data set is exposed. For example, when G is small (Fig. 1(b), $p = 0.7$, $q = 0.2$), bars in images are almost invisible (Fig. 4.2(A)).

Information gain also decreases when BFA is not perfect. Particularly, G decreases when one of the factors is missing (Fig. 4.1). The decrease of G occurs in this case due to increasing q_j. G also decreases when false factors are added to true factors. In the experiments, to the true 16 factors we added 16 false factors that were crosses of randomly selected vertical and horizontal bars. As shown below, such kind of false factors is typical for some BFA methods. In the experiment whose results are depicted in Figs. 4.1 and 4.2, the scores for the false factors were given precisely as those for the true factors. Additional false factors result in decreasing G due to the increase of the first term in (3.2) that gives the information required to describe scores. As shown in Fig. 4.1, G also decreases when true scores were excluded or false scores were added. Thus, all kinds of errors result in decrease of the information gain. Hence, we can conclude that information gain is a reliable measure for different BFA methods comparison and for detecting hidden BFA structure in a given data set as well.

5 Attractor Neural Network with Increasing Activity — ANNIA

We describe here the basic version of ANNIA method, which is sufficient to solve BP. Some extensions of the method for more complex problems were presented earlier [Frolov et al. (2007), Frolov et al. (2009)]. At the same time, we describe here some details of the basic version which were omitted in our previous papers.

In this section, ANNIA operation is illustrated when solving BP for 16 bars on a grid of 8 by 8 pixels. The average number of bars mixed in data set patterns is 2. Both factor distortion and specific noise are absent.

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

$$J_{ij} = \sum_{m=1}^{M} (X_{mi} - q_m)(X_{mj} - q_m), \qquad i, j = 1, \ldots, N, \; i \neq j, \qquad J_{ii} = 0, \tag{5.1}$$

where $q_m = \sum_{i=1}^{N} X_{mi}/N$ is the total activity of the m-th pattern. Factors are revealed as attractors of network dynamics in the 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 presentation of $\mathbf{X}^{in}$, 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. 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 $k_{in} + 1$ neurons is treated as the initial network state for the next iteration step, and 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}. 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 network activity stabilizes at the level of k active neurons, a Lyapunov function is calculated:

$$\lambda(k) = \mathbf{X}(t+1)\mathbf{J}\mathbf{X}^T(t)/k, \tag{5.2}$$

where $\mathbf{J}$ is a 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 point attractor, $\mathbf{X}(t+1) = \mathbf{X}(t)$). As suggested by [Frolov et al. (2007)], the identification of factors was based on the analysis of the change 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 is the number of active neurons in the factor), $\lambda(k)$ increases proportionally to k, and then sharply breaks at the point $k = n_f$ (Fig. 1(a)). As shown in Fig. 1(b), the activation threshold $T(k)$ also increases proportionally to k, and then jumps down at the point $k = n_f$. The increment of $R(k) = \lambda(k)/(k-1) - T(k)/k$ has a clearly expressed peak at this point (Fig. 1(d)). Thus, the peak of $R'(k) = R(k) - R(k-1)$ was used as an indicator of factor on each recall trajectory. The pattern of the network activity at the peak gives the factor loadings for the found factor.

Some trajectories 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 crossing bars, and $k = 16$ corresponds to parallel bars. Similar but weaker peaks appear also at points $k = 22$, $k = 23$ and $k = 24$ (not depicted in Fig. 5.1) 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. 1(a), sometimes Lyapunov function jumps up from one to another continuous trajectory. In this step, network activity transfers to an attractor far from the attractor at the previous step. As shown in Fig. 1(d), such a transfer could also produce a peak of R'. To avoid false treating of such transfers as factors, we calculated on each point of each trajectory the similarity $Sim(k)$ between patterns of network activity in the current attractor $\mathbf{X}^{attr}(k)$ and in the previous attractor $\mathbf{X}^{attr}(k-1)$ as

$$Sim(k) = \frac{a - (k-1)k/N}{(k-1)(1-k/N)}, \tag{5.3}$$

where a is the number of common Ones in $\mathbf{X}^{attr}(k)$ and $\mathbf{X}^{attr}(k-1)$. If $\mathbf{X}^{attr}(k)$ contains $\mathbf{X}^{attr}(k-1)$, then $Sim(k) = 1$. If $\mathbf{X}^{attr}(k)$ and $\mathbf{X}^{attr}(k-1)$ are independent, then $Sim(k)$ is equal to zero on average. We assumed that 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 transfer from $\mathbf{X}^{attr}(k-1)$ to $\mathbf{X}^{attr}(k)$ as a jump. Thus, the point on the trajectory with the largest peak of R' could be considered as related to 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 appearances in the data set. When the initial network states are chosen randomly, as in the procedure described above, network activity tends to converge to factor with the largest attraction basin. To suppress the dominance of such factor in a subsequent search, we deleted it from the network memory according to the Hebbian unlearning rule, i.e., by subtracting ΔJ_{ij} from synaptic connections J_{ij}:

$$\Delta J_{ij} = \bar{J}[(X_i(t) - r)(X_j(t+1) - r) + (X_i(t+1) - r)(X_j(t) - r)], \quad j \neq i, \quad \Delta J_{ii} = 0, \tag{5.4}$$

where $\mathbf{X}(t)$ and $\mathbf{X}(t+1)$ are the successive patterns of network activity in the attractors (this unlearning rule takes into account that attractor can be cyclic of the length two), $\bar{J} = \lambda/(n_f - 1)$ is the mean weight of synaptic connections between the factor neurons, and $r = n_f/N$ is the level of factor activity.

Network dynamics can converge not only to true attractors corresponding to factors, but also to spurious attractors far from all factors [Frolov et al. (2007)]. The Lyapunov function for the spurious attractors is smaller than that for factors (Fig. 1(a)). To separate true attractors from spurious ones, we used the following heuristic method. After finding a peak of R' on the recall trajectory, we activated random set of k_p neurons (k_p is the number of active neurons at the peak) and found the maximal synaptic excitation over all neurons of the network. We repeated this procedure 100 times and calculated mean $m(k_p)$ and standard deviation $\sigma(k_p)$ of maximal excitations. If the Lyapunov function in the peak exceeded the value $h_{max} = m(k_p) + 2\sigma(k_p)$, we treated the found point on the

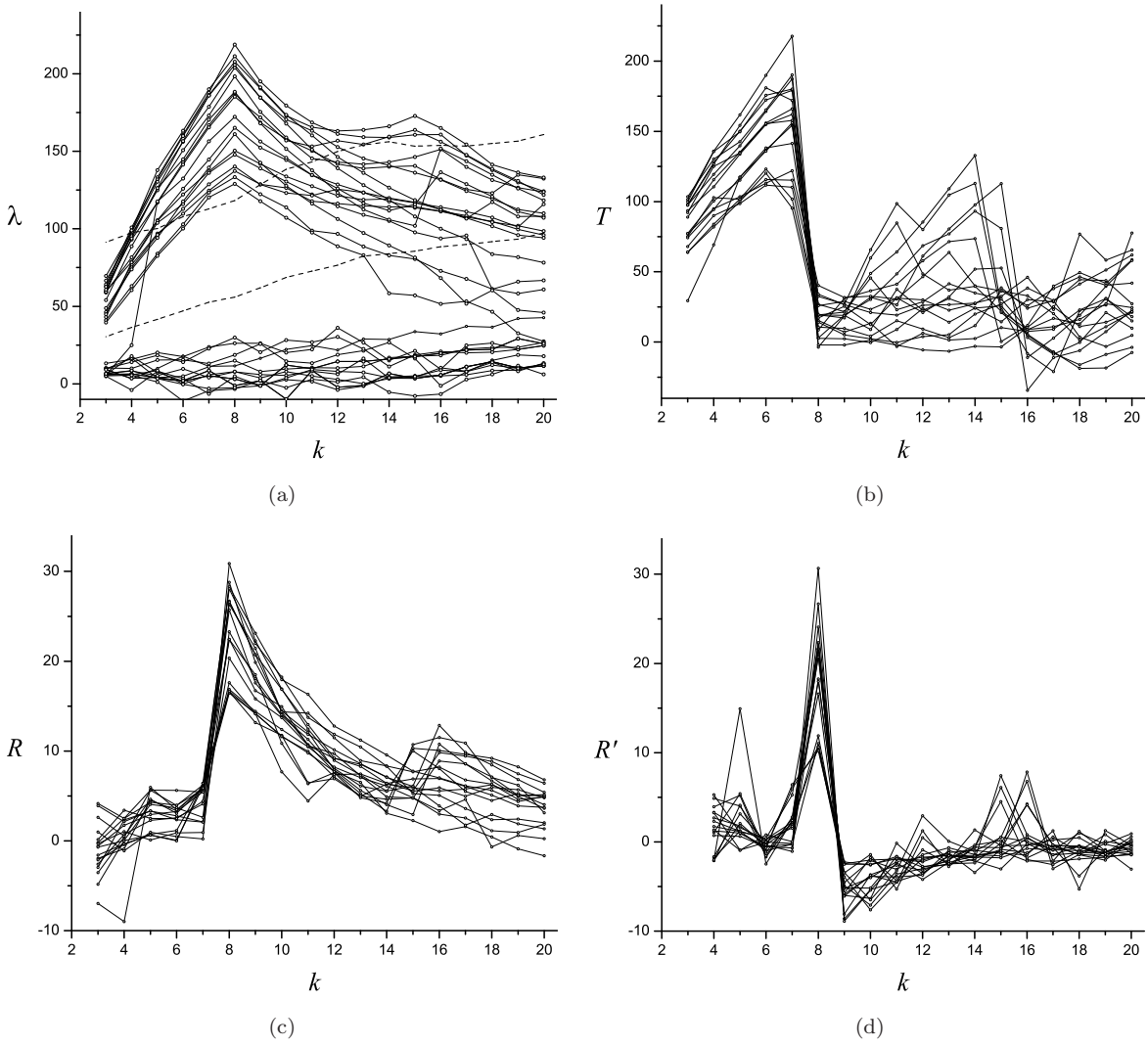


Figure 5.1: Lyapunov function λ (a), activation threshold T (b), function $R = \lambda/(k-1) - T/k$ (c) and its derivative R' (d) in dependence on the number of active neurons k. Dashed lines in (a) are thresholds for separating true and spurious trajectories at the beginning (upper line) and at the end (lower line) of the recall procedure. Results were obtained for data set consisting of $M = 400$ patterns.

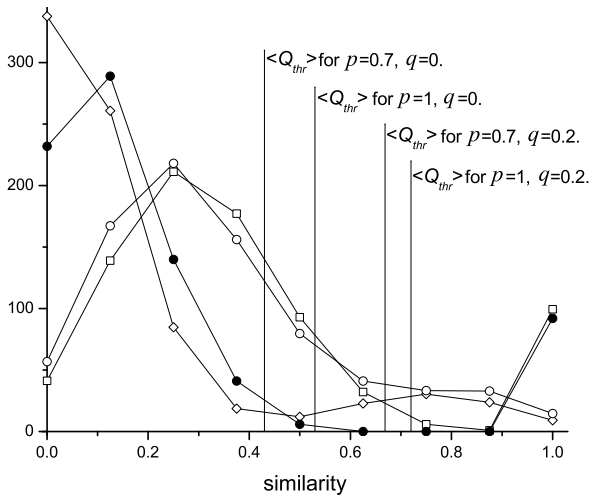


Figure 5.2: Distribution of similarity Q between the factors and patterns containing them (right mode) and not containing them (left mode). Vertical lines are thresholds for mode separation used to prescribe factor scores to patterns. ● – $p = 1$, $q = 0$, □ – $p = 1$, $q = 0.2$, ◇ – $p = 0.7$, $q = 0$, ○ – $p = 0.7$, $q = 0.2$.

trajectory as a factor, in the opposite case — as a spurious state. The borders h_{max} separating true and spurious trajectories at the beginning (upper curve) and at the end (lower curve) of the recall procedure are shown in Fig. 1(a) by the dashed lines. We did no unlearning when spurious states were identified. When all factors were found and deleted from the network memory, only spurious attractors can be activated. The appearance of only spurious attractors in the recall procedure indicates that all factors are found. We stopped the recall procedure when ten sequential recall attempts lead to spurious attractors only. Initially, almost all trajectories were true. So, only about 16 trials were required to find all factors. Note that both $\lambda(k)$ for true trajectories and the border $h_{max}(k)$ separating true and spurious trajectories markedly decrease as factor discovering proceeds, due to sequential deleting of factors with the highest values of the Lyapunov function. The spurious trajectories shown in Fig. 1(a) appeared only after deleting all factors.

To find factor scores, we calculated similarity between each pattern of the data set and each found factor as $Q = a/n_f$, where a is the number of common Ones in the factor and in the pattern. Usually, the distribution of Q has two clearly separate modes (Fig. 5.2). The mode with the larger Q corresponds to patterns containing the factor, the second one corresponds to patterns not containing it. To separate these modes, we used the threshold

$$Q_{thr} = m_Q + 2\sigma_Q\,, \tag{5.5}$$

where $m_Q = \sum_{j:p_{ij}\neq 0} p_j$, $\sigma_Q = \sqrt{\sum_{j:p_{ij}\neq 0} p_j(1-p_j)}$, p_j is the frequency of appearance of the j-th signal component in the data set, and summation is performed only over the attributes j constituting the i-th factor. The values m_Q and σ_Q are estimations of mean and standard deviation of Q. So we put the score to be One if $Q > Q_{thr}$ and Zero in the opposite case. As shown in Fig. 5.2, this threshold provides successful separation of two modes even in the case of large noise in images.

Thus, the basic version of the ANNIA Boolean Factor Analysis procedure by consists of three steps:

1. Patterns of the data set are stored in the network according to the Hebbian learning rule (5.1).

2. Factors are searched by the two-run recall procedure with the Hebbian unlearning (5.4). Factors are identified as the peaks in the R' curves, by comparison of their Lyapunov function values λ with h_{max}, and taking into account the continuity of the recall trajectory. Pattern of the network activity at the peak gives the factor loadings.

3. Factor scores for each revealed factor and each pattern of the data set are determined by similarity Q between the factor and the pattern.

6 Related Methods

In this section, we consider four other BFA related methods investigated in this paper. Some of them were supposed [Spratling (2006), Lücke and Sahani (2008)] to be the most efficient for BFA, at least for solving the bars problem.

6.1 Boolean Matrix Factorization — BMF.

Boolean Matrix Factorization [Belohlavek and Vychodil (2010)] implies presentation of binary matrix of observed data set $\mathbf{X}$ in the form

$$\mathbf{X} = \mathbf{S} \otimes \mathbf{F}, \tag{6.1}$$

where each row of binary $M \times N$ matrix $\mathbf{X}$ is an observed pattern, each row of binary $L \times N$ matrix $\mathbf{F}$ is a representation of factor in the signal space and each row of binary $M \times L$ matrix $\mathbf{S}$ is a set of factor scores defining which factors are mixed in the patterns. Boolean matrix product $\otimes$ means that each component of matrix $\mathbf{X}$ is obtained as $X_{mj} = \bigvee_{i=1}^{L} S_{mi} f_{ij}$. This representation of each scene corresponds to formula (2.1), when noise (both factor distortion and specific noise) is absent, that is $p_{ij} = 1$ and $q_j = 0$. The method implies identification of a minimal set of factors that provide representation of the observed data in the form (6.1). Since this combinatorial problem is NP complete [Miettinen et al. (2008)] existing methods give reasonable, but not obligatory optimal solutions. Recently [Belohlavek and Vychodil (2010)] revealed a tight relationship between BMF and formal concept analysis [Ganter et al. (1999)] and developed two simple greedy algorithms of BMF. In our computer simulation we used the second faster algorithm. Since Boolean matrix factorization implies exact (non-noisy) observations as an input, only the first term in (3.2) is nonzero.

6.2 Dendritic Inhibition Network — DI

This method was developed by [Spratling and Johnson (2002)] for finding parts-based decompositions of images [Hoyer (2004)]. The method is based on a feed-forward neural network with lateral inhibition. The main idea of the method is to use lateral inhibition of individual synapses instead of total inhibition of a neuron. As a result of network learning, neurons of the output layer acquire specific sensitivity to factors constituting patterns of the data set: the appearance of a factor in the pattern presented to the input layer leads to the strong activation of the related neuron at the output layer. Output neurons can be activated when factors are partially distorted and in the presence of noise. Thus, activity of each neuron at the output layer provides a gradual estimation of the confidence that this pattern contains the related factor. To assign binary factor scores to the pattern we chose the neurons with highest activity as winners. The threshold of this binarization procedure is chosen to maximize the information gain. Thus, for the used measure of BFA efficiency, this procedure provides the optimal outcome of the method.

[Spratling and Johnson (2003)] and [Spratling (2006)] compared the efficiency of DI with the method suggested by [Foldiak (1990)] and some methods of nonnegative matrix factorization, which were treated as related methods. Since those related methods showed lower efficiency than DI, we do not consider them.

6.3 Expectation-Maximization Method for Maximal Causes Analysis — MCA_3

Recently [Lücke and Sahani (2008)] have studied the bars problem with the Expectation-Maximization (EM) method [Dempster et al. (1977)]. The method allows for finding parameters of a given probabilistic generative signal model to maximize the likelihood of the observed data. In the generative model studied by [Lücke and Sahani (2008)], multiple active hidden causes (factors in terms of BFA)

combine to determine the values of an observed variable through a max function. Each cause results in a set of observations given by a vector of generative influences (factor loadings in terms of BFA).

If several causes result in the same observation, then the strongest influence alone determines the value of the observed variable. If all influences have the same value, then the max function is equivalent to Boolean summation of the influences and the generative model becomes almost equivalent to the generative model of BFA introduced in Section 2. The special case of noise studied by [Lücke and Sahani (2008)] is a random choice of the observed variable according to Poisson distribution with the mean equal to the strongest influence. [Lücke and Sahani (2008)] suggested three EM methods for the described generative model that provided similar results, but the method called MCA_3 was slightly better than others. The method is restricted to the case of sparse scores when each pattern of the data set contains not more than three factors.

The output of MCA_3 are probabilities that patterns of the data set contain found factors. To assign binary factor scores to the pattern we performed the same binarization procedure as described for DI.

6.4 Expectation-Maximization Method for Boolean Factor Analysis — EMBFA

We applied the EM approach directly to the generative model introduced in Section 2 and call the method as EMBFA. The EM method maximizes the likelihood of the observed data by maximizing the free energy [Dempster et al. (1977)]

$$\mathcal{F}(\mathbf{\Theta}, g) = \sum_{m=1}^{M} \sum_{\mathbf{S}} g_m(\mathbf{S})[\log P(\mathbf{X}_m|\mathbf{S}, \mathbf{\Theta}) + \log P(\mathbf{S}|\mathbf{\Theta})] + H(\mathbf{g}),$$

where $g_m(\mathbf{S})$ is the expected distribution of factor scores for the m-th pattern and $H(\mathbf{g}) = \sum_m H(g_m(\mathbf{S}))$ is the Shannon entropy of $\mathbf{g}$. Note that in Section 3 parameters of the generative model $\mathbf{\Theta} = \{p_{ij}, q_j, \pi_i\}$ were estimated by likelihood maximization under the given scores, while EM maximizes the likelihood optimizing model parameters as well as factor scores. The iterations of EM alternatively increase $\mathcal{F}$ with respect to the distributions g_m, while holding $\mathbf{\Theta}$ fixed (the E-step), and with respect to $\mathbf{\Theta}$ (as in Section 3), while holding g_m fixed (the M-step).

At the E-step, when $\mathbf{\Theta}$ is fixed, the distributions g_m which maximize $\mathcal{F}(\mathbf{\Theta}, \mathbf{g})$ are calculated according the following equation

$$g_m(\mathbf{S}|\mathbf{\Theta}) = \frac{P(\mathbf{S}|\mathbf{\Theta})P(\mathbf{X}_m|\mathbf{S}, \mathbf{\Theta})}{\sum_{\mathbf{S}} P(\mathbf{S}|\mathbf{\Theta})P(\mathbf{X}_m|\mathbf{S}, \mathbf{\Theta})},$$

where $P(\mathbf{S}|\mathbf{\Theta}) = \prod_{i=1,L} {\pi_i}^{S_i}(1-\pi_i)^{1-S_i}$, $P(\mathbf{X}_m|\mathbf{S}, \mathbf{\Theta}) = \prod_{j=1,N} P(X_{mj}|\mathbf{S}, \mathbf{\Theta})$, and $P(X_{mj}|\mathbf{S}, \mathbf{\Theta})$ is given by (2.2). The obtained distributions g_m provide the equality $\mathcal{F}(\mathbf{\Theta}, \mathbf{g}) = \sum_{m=1}^{M} \mathbf{E}|_{\mathbf{S}}[\mathcal{L}_m]$ [Neal and Hinton (1998)], where $\mathcal{L}_m$ is defined by (3.9). Thus, they provide the expected likelihood of the observed data over factor scores under the given parameters of the generative model.

At the M-step, when distributions g_m are fixed, π_i can be found by

$$\pi_i = (1/M) \sum_{m=1}^{M} S_{mi},$$

where

$$S_{mi} = \sum_{\mathbf{S}} g_m(\mathbf{S}|\mathbf{\Theta}) S_i. \tag{6.2}$$

Respectively, p_{ij} and q_j can be found by maximization of $\mathcal{F}(\mathbf{\Theta}, \mathbf{g})$ according to the system of $L*N+N$ equations

$$\begin{aligned} \frac{\partial \mathcal{F}}{\partial p_{ij}} &= \sum_{m=1}^{M} \sum_{\mathbf{S}} g_m(\mathbf{S}|\mathbf{\Theta}) P(X_{mj}|\mathbf{S}, \mathbf{\Theta})^{-1} \frac{\partial P(X_{mj}|\mathbf{S}, \mathbf{\Theta})}{\partial p_{ij}} = 0, \\ \frac{\partial \mathcal{F}}{\partial q_j} &= \sum_{m=1}^{M} \sum_{\mathbf{S}} g_m(\mathbf{S}|\mathbf{\Theta}) P(X_{mj}|\mathbf{S}, \mathbf{\Theta})^{-1} \frac{\partial P(X_{mj}|\mathbf{S}, \mathbf{\Theta})}{\partial q_j} = 0, \end{aligned} \tag{6.3}$$

where $\partial P(X_{mj}|\mathbf{S},\mathbf{\Theta})/\partial p_{ij}$ and $\partial P(X_{mj}|\mathbf{S},\mathbf{\Theta})/\partial q_j$ are given by (3.11). To solve the system (6.3), we use the same iterative procedure as defined by (3.12). Thus we obtain

$$\begin{aligned} p_{ij}(k+1) &= \frac{1}{\sum_{m=1}^{M}\sum_{\mathbf{S}} g_m(\mathbf{S}|\mathbf{\Theta})S_i} \sum_{m=1}^{M}\sum_{\mathbf{S}} g_m(\mathbf{S}|\mathbf{\Theta}) \frac{S_i p_{ij}(k) X_{mj}}{1-(1-q_j(k))\prod_{l=1,L}(1-p_{lj}(k))^{S_l}} \\ q_j(k+1) &= \frac{1}{M}\sum_{m=1}^{M}\sum_{\mathbf{S}} \frac{q_j(k)X_{mj}}{1-(1-q_j(k))\prod_{i=l,L}(1-p_{lj}(k))^{S_l}}. \end{aligned} \tag{6.4}$$

As shown in Section 3, this iterative procedure provides monotonic increase (not decrease) of likelihood function at each step. Since we assume that probabilities p_{ij} are sufficiently high for components constituting the i-th factor ($f_{ij} = 1$) and equal to zero for other components ($f_{ij} = 0$), at each iteration step we put $p_{ij} = 0$ if the j-th component satisfies inequality (3.13). It is interesting to note that without this threshold truncation, EM does not converge because of uncertainties arising from the competition between factors defined by p_{ij} and noise defined by q_j. For example, one of the factors (let it be the first factor $\mathbf{f_1}$) could contain all signal components ($f_{1j} = 1$ for $j = 1 \dots N$) then let us suppose that it appears in all patterns of the data set. In this case the presence of such factor is equivalent to the introduction of specific noise defined by probabilities $q_j = p_{1j}$. Thus, this factor will compete with the specific noise during the iteration process but threshold truncations of p_{ij} allows for avoiding such competition.

The obtained values of p_{ij}, q_j and π_i are used as the input for the next E-step. EM iterative procedure terminates once all $\sqrt{\sum_j (p_{ij}^{old} - p_{ij}^{new})^2}/\sum_j p_{ij}^{old}$ remained smaller than $2.5 \cdot 10^{-3}$ for 20 sequential iterations. The same criterion was used by [Lücke and Sahani (2008)] to terminate EM procedure in MCA_3.

The procedure convergence provides some maximum of the likelihood function. When procedure converges, the final values S_{mi} are estimates of the factor scores. To satisfy the generative model, we binarized those values. As for DI and MC_3, the threshold of binarization was chosen to provide maximal information gain for a given set of data.

As [Lücke and Sahani (2008)] did for MCA_3, we also restricted EMBFA algorithm to the case of sparse scores, when only a small number of factors (no more then 3) are supposed to be mixed in the observed patterns. In this case, summation over $\mathbf{S}$ in the above formulas is reduced to

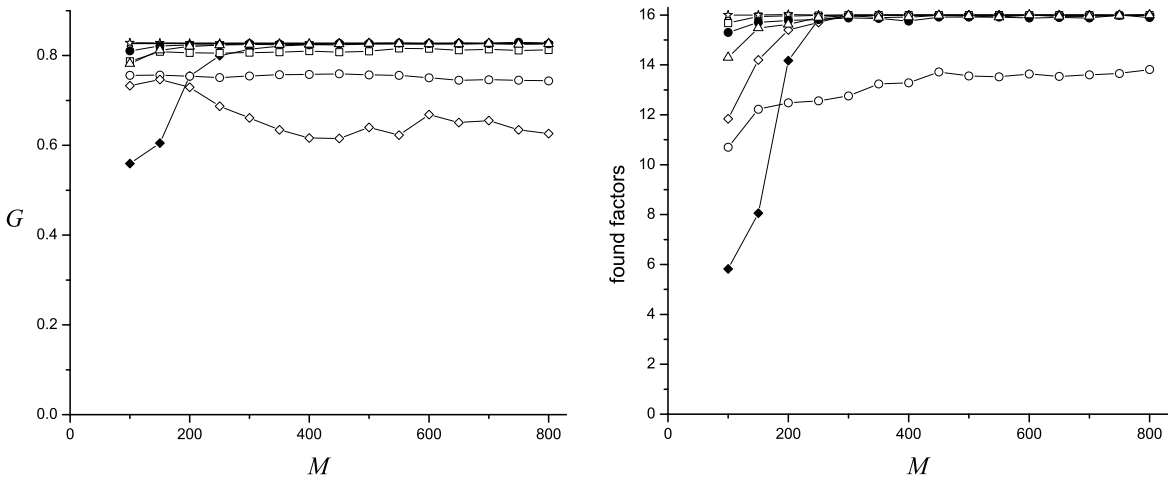
$$\sum_{\mathbf{S}}(\dots) = (\dots)_{\mathbf{S}=0} + \sum_{i}(\dots)_{\mathbf{S}=\mathbf{S}_i} + \sum_{i<j}(\dots)_{\mathbf{S}=\mathbf{S}_{ij}} + \sum_{i<j<k}(\dots)_{\mathbf{S}=\mathbf{S}_{ijk}}, \tag{6.5}$$

where $\mathbf{S}_i$ is the vector of factor scores with all zeros except S_i, $\mathbf{S}_{ij}$ is the vector of factor scores with all zeros except S_i and S_j, and $\mathbf{S}_{ijk}$ is the vector of factor scores with all zeros except S_i, S_j and S_k. An increase of the number of terms in (6.5) leads to the considerable rise in computational complexity.

7 Performance of BFA Methods in Solving Bars Problem

In this section, we compare efficiency of the five methods for Boolean Factor Analysis: ANNIA, BMF, DI and two EM methods: MCA_3 and EMBFA. We especially emphasize two BFA methods: ANNIA, since it was already shown to be efficient in various tasks by [Frolov et al.(2006a), Frolov et al. (2008), Frolov et al. (2009)], and thus is expected to be efficient for BFA in general, and EMBFA, since it represents the application of powerful EM technique to the proposed general generative BFA model, and thus is also expected to be efficient for BFA in general.

The methods are compared according to two criteria. The first one is information gain introduced in this paper. The second one is a commonly used measure, which is the estimation of the number of true factors L_f^{true} among the whole set of found factors L_f [Spratling (2006), Lücke and Sahani (2008)]. Each of the considered BFA methods provides weights w_{ij} evaluating the confidence that j-th attribute pertains to i-th found factor. To estimate L_f^{true}, the sum of the weights corresponding to each true factor was calculated for each found factor. Since true factors are horizontal and vertical bars, the sums of weights for each found factor are calculated over all rows and columns of the image grid. A found factor is considered to represent a particular bar if the total weight corresponding to that



(a) (b)

Figure 7.1: Information gain G (a) and number of true found factors L_f^{true} (b) for five BFA methods in dependence on data set size M. Noise is absent ($q = 0$, $p = 1$). △ – ANNIA, ○ – EMBFA, ☆ – BMF, □ – DI, ◇ – MCA_3, ● – EMBFA with fixed number C=2 of mixed bars, ◆ – MCA_3 with fixed number C=2 of mixed bars. Thick line – "ideal" solution.

bar was twice that of the sum of the weights for any other bar, and if the minimum weight in the row or column corresponding to that bar was greater than the mean of all the weights for that found factor. In ANNIA the weights w_{ij} are binary. They are determined by patterns of the network activity relating to the peaks on the curves for R'. In BMF the weights are binary components of matrix F in the equation (6.1). In methods DI, MCA_3 and EMBFA, the weights are gradual. In DI, these are weights of synaptic connections between neurons of input and output layers of the network. In MCA_3, weights are represented by influences of causes on observed variables. In EMBFA, weights are represented by probabilities p_{ij}. Note that in contrast to the first criterion the second one requires a priori knowledge concerning true factors, and thus it can be applied only to artificial data set when signal hidden structure is known in advance. Since this criterion is approved for the bars problem, it is reasonable to compare the performances of the BFA methods by both criteria. Note that information gain G is calculated with the use of the likelihood procedure presented in Section 3. In contrast, L_f^{true} is estimated before this procedure as in original papers on the bars problem [Spratling (2006), Lücke and Sahani (2008)].

7.1 Clean Factors

Initially, the methods are compared for the case of clean, undistorted bars ($p = 1$ and $q = 0$). As above, we consider patterns of the data set that are 8-by-8 binary images, the factors are 16 vertical and horizontal bars (one pixel width), 2 bars are mixed in each image on average.

ANNIA

The information gain obtained by ANNIA is shown in Fig. 1(a) depending on the size of the data set. The information gain obtained by ANNIA almost coincides with that for ideal one shown in Fig. 1(a) by thick line, and discrepancy is visible only when M is relatively small.

To clarify the origin of the discrepancy we show in Fig. 1(b) the average number of factors correctly found by ANNIA. The fraction of not revealed true factors increases when M decreases. When sample data set is small, some factors by chance could appear much less frequently than they would on average. Since the Lyapunov function of factor is proportional to the frequency of its appearance in the data set, for some factor that happens occurs less frequently, it could be smaller than h_{max}. As a result,

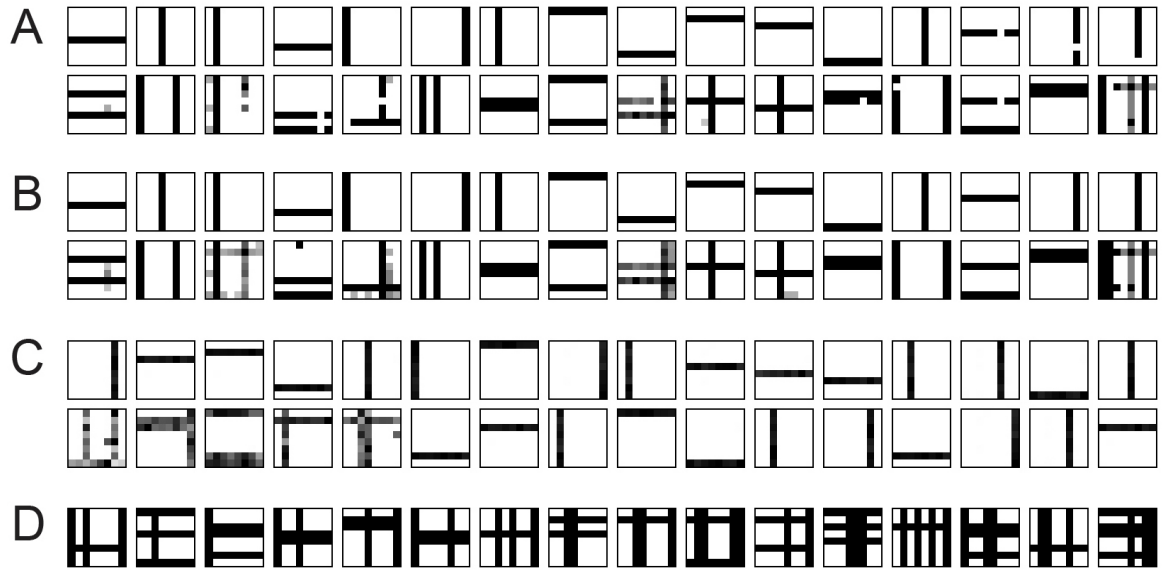


Figure 7.2: Factors found by EMBFA (A – before, B – after likelihood maximization) and by MCA_3 (C), D – examples of images where bars were identified by EMBFA but not by MCA_3. $M = 800$.

this factor could not be found. However, the omission of the factor by ANNIA leads to a smaller decrease of gain than shown in Fig. 4.1 for the case when one factor is missed (3% in Fig. 1(a) vs 5% in Fig. 4.1). It happened because ANNIA omits the factor with the lowest frequency of appearance, and thus the influence of its omission to the gain is minimal. For the case shown in Fig. 4.1, the omitted factor was chosen randomly and thus on average the influence of its omission is larger. When M increases the difference between the frequencies of appearance of different factors becomes smaller, and this effect disappears.

EMBFA

For EMBFA, as well as for DI and MCA_3, the number of desired hidden factors has to be set in advance. In the majority of computer experiments performed by [Spratling (2006)] and [Lücke and Sahani (2008)], this number was taken twice higher than the actual number of factors. In most cases, such setup ensured the successful search of all 16 true factors among predefined 32 factors. In our experiments with EMBFA, DI, and MCA_3 the predefined number of hidden factors was also taken twice higher than the actual number of factors.

To start the EM procedure, we set $\pi_i = 1/32$, $q_j = 0$ and initialized p_{ij} with random values uniformly distributed in the range from 0.3 to 0.8.

As shown in Fig. 7.2(A), in EMBFA false factors are mainly mixtures of two bars, and some true factors are not complete: they contain only 7 pixels (the last three bars of the first row). According to the used criterion these factors are not identified as true, and the average number of found true factors is less than 14 (Fig. 1(b)). However, as shown in Fig. 7.2(B), the procedure of likelihood maximization corrected some of those incomplete factors, and after it the average number of factors identified by this criterion as true increases to about 15.

Although after likelihood maximization on average 15 of 16 true factors were revealed correctly, the information gain G provided by EMBFA is less than that in Fig. 4.1 for the case when one factor is missing. This gain decrease occurs because of some missing scores. According to (6.5), the method is able to identify scores only in patterns containing not more than three mixed factors. For the used generative model, the number of mixed factors k has binomial distribution $B(k, C/L, L)$ where $C = 2$, $L = 16$. According to this distribution, 13% of patterns containing more than three factors are generated. Since only three factors can be identified in these patterns, 18% of scores are expected to be missed. However, the actual gain obtained by EMBFA is higher than that shown in Fig. 4.1 for the

case with only 10% of missing scores. The reason is that EMBFA was sometimes able to recognize all the bars in a pattern containing a mixture of 5 bars. This occurs if EMBFA considered the pattern to be the mixture of 3 found factors: one is true factor corresponding to single bar, and two others are false factors that are the mixtures of two bars. Some examples of patterns in which some true factors were recognized by EMBFA are shown in Fig. 7.2(D). Particularly, the first of the shown patterns is recognized as created by true factors 4 and 7 in the upper row, and by false factor 13 in the lower row in Fig. 7.2(A). Another example is the last image in Fig. 7.2(D), where EMBFA recognizes factor 13 in the upper row and factors 8 and 12 in the lower row of (A). Two remaining bars are treated by EMBFA as specific noise. So, totally EMBFA missed less than 10% of scores.

To expose the effect of the limitation of EMBFA algorithm to the case when each image of the data set contains not more than three bars, we tested it for the case when each pattern of the data set contains exactly two randomly chosen bars. Those results are shown in Fig. 7.1 (curves marked by ●). For this generative model, EMBFA provides perfect solution. So, we conclude that the observed decrease of information gain compared to the "ideal" one results from this restriction.

Other Methods

As shown in Fig. 7.1, the BMF method proposed by [Belohlavek and Vychodil (2010)] provides an exact solution of the bars problem. Both information gain and the number of correctly found factors coincide with the ideal solution.

In the computer experiments involving DI and MCA_3 we used parameters recommended in the original papers [Spratling (2006), Lücke and Sahani (2008)]. When the size of sample data set is sufficiently large, both methods precisely reveal of all true factors (Fig. 1(b)). This is achieved at $M = 200$ for DI and at $M = 300$ for MCA_3. In spite of the fact that all true factors were found, the information gains obtained by both methods are less than the ideal one. Just as for EMBFA, the reason for G decrease is omission of some factor scores. For DI, the fraction of missing scores was 2.3%, and for MCA_3 it was to 23%. The fraction of missing scores for MCA_3 is about twice higher than that for EMBFA, and even slightly higher than 18%. Recall that namely this fraction of missing scores is expected in both EM methods if not more than three factors can be found in each pattern, as claimed in the formulation of these methods according to formula (6.5). Note that unlike EMBFA, for MCA_3 this principal weakness of EM methods is not compensated by false factors. As shown in Fig. 7.2(C), false factors in MCA_3 are mainly not mixtures of bars but their duplicates. Thus, in distinction to EMBFA, MCA_3 could not reveal true factors in images containing mixture of four or more factors shown in Fig. 7.2(D).

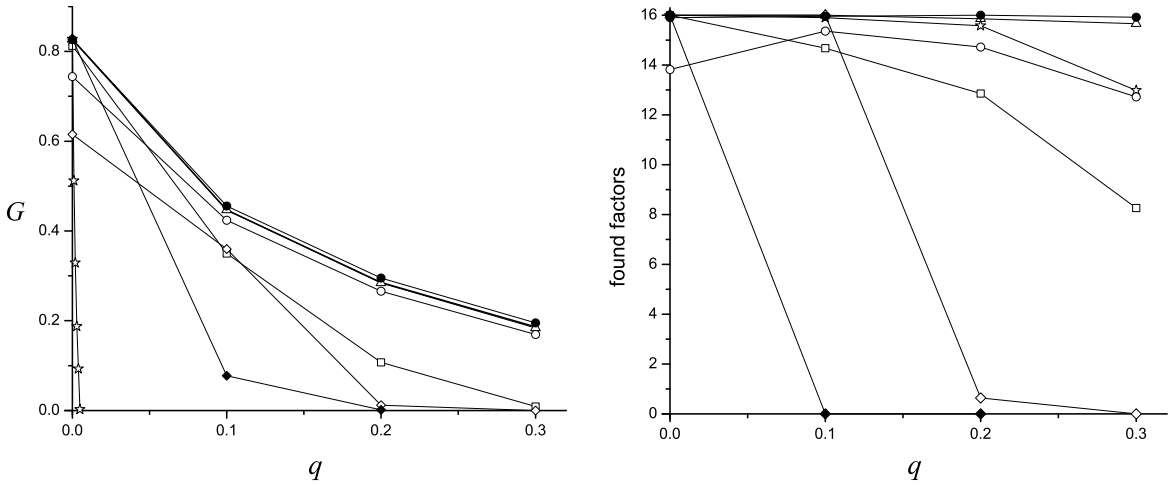
We also tested MCA_3 for the data set with patterns containing exactly two randomly chosen bars. MCA_3 (as well as EMBFA) provides perfect solution for this generative model.

7.2 Sensitivity to Noise

Figures 7.3 and 7.4 demonstrate sensitivity of BFA methods to noise. As in Section 4, the noise was assumed to be distributed uniformly over signal components and factors so that $q_j = q$ for any j, and $p_{ij} = p$ for any i and j. We tested BFA methods using data sets of the size $M = 800$. As shown in Fig. 7.1, in the absence of noise both the information gain G and the number of found true factors L_f^{true} reach saturation at that particular M. We checked that saturation is also reached for $M = 800$ in the presence of noise for all methods except BMF.

Specific Noise: $q > 0$, $p = 1$

As shown in Fig. 3(a), BMF is most sensitive to the presence of noise. Its information gain G drops to zero at the noise level that all other methods easily handle. Note that in spite of the fast drop of G, the number of true factors L_f^{true} found by BMF remains high (Fig. 3(b)) and scores for true factors are found precisely. As follows from the BMF description in Section 6.1, it treats specific noise as common factors, and thus as q increases, the total number of found factors L_f increases as well. For example, for $q = 0.001$ $L_f = 40$, and for $q = 0.01$ L_f increases to 85. Since BMF provides precise decomposition of the data set matrix in the form (6.1), the second term in (3.2) for BMF is always



(a) (b)

Figure 7.3: Information gain G (a) and number of found true factors L_f^{true} (b) in dependence on q for $p = 1$. △ – ANNIA, ○ – EMBFA, ☆ – BMF, □ – DI, ◇ – MCA_3, ● – EMBFA with fixed number C=2 of mixed bars, ◆ – MCA_3. Thick line – "ideal" solution.

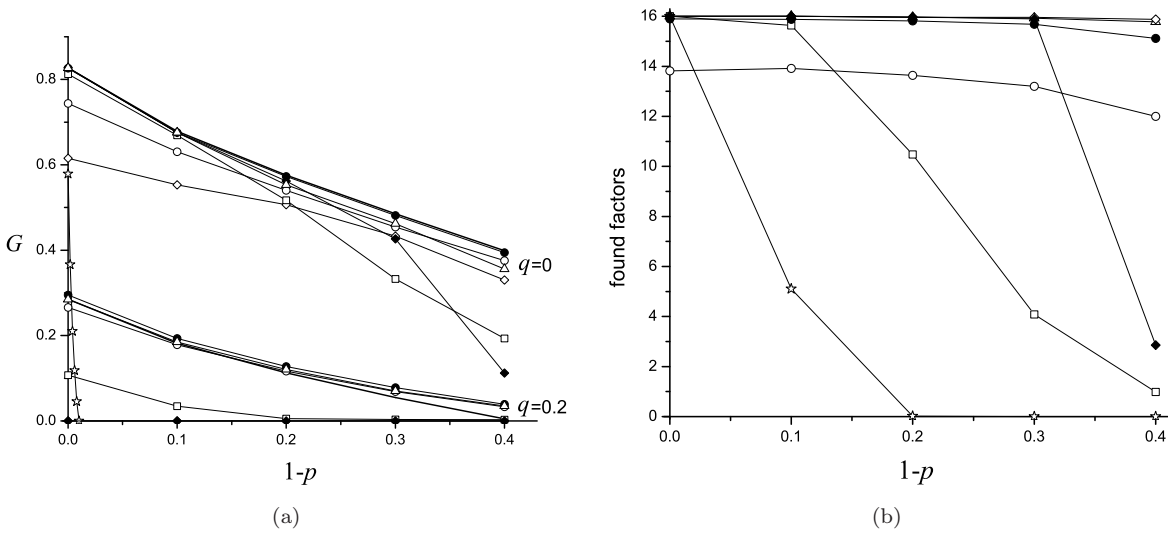
equal to zero. Thus, the drop of G occurs only due to the increase of the first term, which gives the entropy of scores. Although the frequency of each false factor in the data set is small (and hence its contribution to the entropy of scores is small), the total effect of false factors is high due to their large number, and so G quickly drops when q increases.

For BMF an increase in M results in an increase of L_f, and consequently in the decrease of G. For example, for $q = 0.005$ $G = 0.27$ for $M = 400$, $G = 0.0024$ for $M = 800$ and G becomes negative under a further M increase. In this case, all true factors could be found precisely while the number of false factors grows incredibly. Thus, according to the criterion of the number of found true factors, BMF performs perfectly. But this is counterintuitive because the portion of true factors among all found factors is negligibly small.

The information gain G obtained by MCA_3 and DI also demonstrate strong sensitivity to q (Fig. 3(a)). For DI an increase of q results in a decrease of G, because the number of found true factors decreases (Fig. 3(b)). For MCA_3, both G and L_f^{true} drop near to zero when q increases to 0.2. In this case, the solution of the bars problem by MCA_3 becomes unstable, i.e., it drastically depends on the peculiarities of the data set or on the choice of initial parameters for the EM procedure. With one random realization of the data set MCA_3 may provide perfect solution (after approximately 300 steps of the EM procedure), with another random realization chosen from the same distribution the procedure converges to some random images as factors (in just 3-5 steps). For $q = 0.2$, we observed a successful search of bars by MCA_3 only in 2 of 50 trials.

As mentioned above, poor performance of MCA_3 could be explained by omission of scores in images containing more than 3 mixed factors. Then, one could expect that, as for the case without noise (see Fig. 7.1), information gain should significantly increase if exactly $C = 2$ bars are mixed in every image instead of two on average. However, it does not happen, because the above-mentioned instability of MCA_3 appears even at the smaller q. Particularly, successful search of true factors was not observed in any of the 50 trials already at $q = 0.1$. Probably, the presence of specific noise contradicts the generative model behind MCA_3.

The results obtained by ANNIA were close to the precise ones. As shown in Fig. 5.2, for $p = 1$ and $q = 0.2$ the threshold Q_{thr} used in ANNIA to separate patterns containing and not containing the factor happened to be slightly shifted to the left relative to the point providing perfect separation of the two similarity modes. As a result, some images not containing factors were identified as containing them, i.e.some false scores were added to precisely found true scores. However, the portion of false scores is small, and so is reduction of G.



(a) (b)

Figure 7.4: (a) Information gain G vs p for $q = 0$ and $q = 0.2$. (b) Number of found factors vs p for $q = 0$. △ – ANNIA, ○ – EMBFA, ☆ – BMF, □ – DI, ◇ – MCA_3, ● – EMBFA with fixed number C=2 of mixed bars, ◆ – MCA_3. Thick line – "ideal" solution.

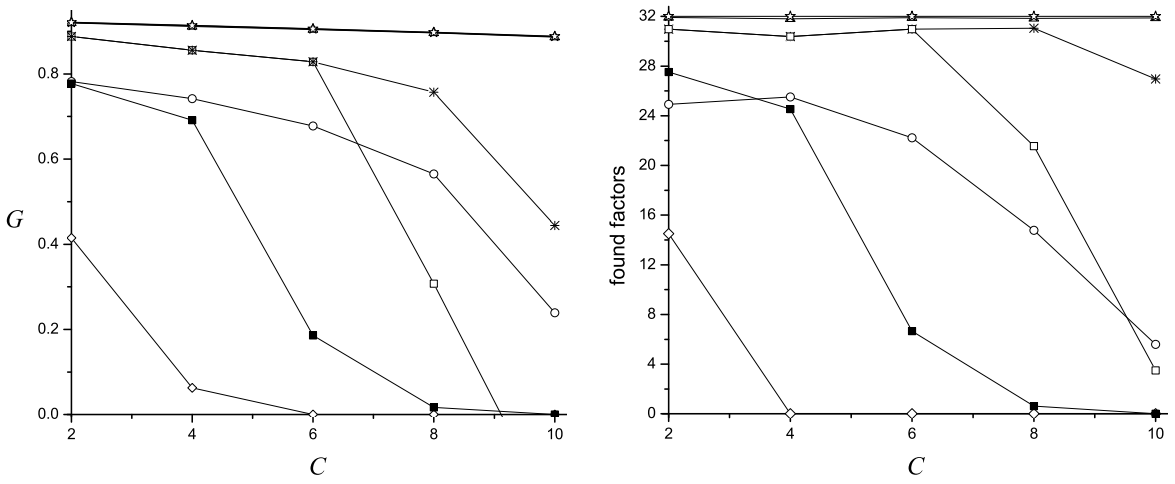
It is interesting that due to the competition between common and specific factors, L_f^{true} found by EMBFA does not depend monotonically on q. At $q < 0.1$, some pixels of true factors are assigned to specific factors which results in an increase of the estimated value of q compared to its actual value in the generative model. At $q > 0.1$, some pixels are added to true factors which results in a decrease of the estimated value of q. In both cases, these factors are not identified as true. For q of about 0.1, there is an equilibrium between these tendencies, and the number of true found factors is maximal. As discussed above, true factors are repaired by likelihood maximization and the errors in their factor loadings obtained by EMBFA do not influence information gain calculated after this procedure. Thus, EMBFA also provides G close to the theoretical value.

As could be expected, EMBFA performance can be improved by fixing the number of mixed factors in each image of the data set. However, in this case not only all true factors were found, but G paradoxically exceeded the ideal value. For fixed number of factors, the tendency of adding pixels to true factors dominates. The probability p_{ij} of added pixels is about $0.4 - 0.6$ (while for true pixels $p_{ij} = 1$). As a result, probability of specific noise q decreases (e.g., EMBFA estimates it as 0.19, while the exact value used in the generative model is 0.2). The increase of G due to decreasing q happened to exceed the decrease of G due to adding some extra pixels to bars.

Distortion of Factors: $q = 0$, $p < 1$

For this kind of noise the information gain obtained by BMF also drops faster than that for other methods (Fig. 4(a)). The reason is similar to the case of specific noise: the increasing number of false found factors which are distorted versions of bars. For $p = 0.998$, the number of found factors amounts to 40, and for $p = 0.99$ to 85. Most of found factors are distorted bars, so that only 5 not distorted bars were identified as factors for $p = 0.9$ (Fig. 4(b)). As for the case of specific noise, G obtained by BMF increases when M decreases. For example, at $p = 0.99$ G amounts to 0.045 for $M = 800$, and to 0.31 for $M = 400$.

DI exhibits similar sensitivity to factor distortion as to specific noise, while MCA_3 is not sensitive to this kind of noise at all. The reason is that factor distortion is a part of the MCA_3 generative model. All true factors are found by this method (Fig. 4(b)) and the information gain G (Fig. 4(a)) is smaller than ideal only due to missing scores in images containing more than 3 bars, as explained in Section 7.1. Thus, MCA_3 performance could be improved by fixing a number of factors in each image. This actually leads to an increase of G. However, when probability of factor distortion increases (i.e.,



(a) (b)

Figure 7.5: Information gain G (a) and number of found factors (b) vs C for 16-by-16 pixel images at $M = 800$. △ – ANNIA, ○ – EMBFA, ☆ – BMF, □ – DI, ◇ – MCA_3, ■ – DI for fixed number of mixed bars in patterns, ✳ – DI with repeated initializations of factor search.

p decreases), MCA_3 loses stability, as described above for the case of specific noise. For $p = 0.6$, MCA_3 provides reasonable solution for only 7 out of 50 trials.

Both ANNIA and EMBFA again provide G close to ideal (Fig. 4(a)). And again (compare with Fig. 3(a)), for EMBFA, G is high in spite of omission of some true factors (Fig. 4(b)). EMBFA performance is higher when the number of bars in each image is fixed, and its gain for this case reaches even higher values than those for ANNIA (which happened to be more sensitive to factor distortion than to specific noise). Fig. 5.2 shows that at $p = 0.7$ for both $q = 0$ and $q = 0.2$ the similarity threshold necessary for detection of factors in images is shifted to the left compared to the threshold for undistorted factors ($p = 1$). This leads to the omission of some factor scores and, in turn, to a drop in G compared to the ideal solution.

Fig. 4(a) also demonstrates the sensitivity of BFA methods to both kinds of noise applied simultaneously ($q = 0.2$, $p < 1$). For such noise parameters, BMF and MCA_3 practically failed (thus, their gain is not depicted in Fig. 4(a)). As shown in Fig. 3(a), at $q = 0.2$ the values of G for MCA_3 were close to zero even in the absence of factor distortion, and BMF failed for much smaller q. DI provides considerably smaller G than the ideal one. The information gain G obtained by ANNIA and EMBFA is again close to the ideal one and for small p it is even higher. This is especially amazing for ANNIA, because usually it finds not all true factors. For example, at $p = 0.7$ only about 14 true factors were found out of 16. The effect of increasing G is paradoxically explained by omission of some factor scores. ANNIA missed factors in images where they were highly distorted. The images containing missing factors were excluded from estimation of p_{ij} by likelihood maximization. Then, estimated values of p_{ij} increased, while estimated values of q_j remained almost unchanged. For example, at $p = 0.7$ the average estimated values of p_{ij} increased from 0.7 to 0.77, while q_j increased from 0.2 to 0.22 only. So, the increase in p_{ij} dominates over the increase of q_j, and thus G increased.

7.3 Sensitivity to Image Complexity C

To investigate sensitivity of BFA methods to complexity C, we increased the size of images to the grid of 16-by-16 pixels. As in the preceding, we considered two bar-choice cases: 1) each of 32 bars is chosen for each image independently with probability $C/32$ or 2) the number of mixed bars in each image is fixed to C. The increased image size allowed us to study the effects of increasing C up to $C = 10$. Here, we investigated only noiseless case, i.e., we put $p = 1$, $q = 0$.

Both ANNIA and BMF appeared insensitive to increasing C, and provide precise solution of the

bars problem even for $C = 10$ (Fig. 7.5). DI performance is getting worse with an increasing C, resulting in a decrease of both G and L_f^{true}. The reason is the solution stability loss similar to the case for MCA_3, described above in Section 7.2. When the number of active neurons at the input of the DI network becomes relatively large because of large C, DI fails. This disadvantage can be overcome by repeating factors search with other initial synaptic weights. We randomly changed synaptic weights until more than a half of bars were found, but made maximally 10 attempts. As shown in Fig. 7.5, this modification essentially improved DI performance.

It seems that the presence of the images with few mixed factors in the data set facilitated factors search by DI. To check this hypothesis, we studied DI performance for the case when the number of mixed bars in each image was exactly C (i.e there were no images containing less than exactly C bars). As shown in Fig. 7.5, this actually worsened DI performance for $C > 4$.

Since MCA_3 and EMBFA are both restricted to the case of sparse scores ($C \leq 3$), one would expect that those methods are most sensitive to the increase of C. This actually happened for MCA_3: it failed for $C = 4$. However, EMBFA amazingly gives reasonable results until $C = 8$, although with less efficiency. The reason is the above mentioned peculiarity of EMBFA to treat some redundant bars as noise when the number of bars mixed in the image exceeds three.

8 Discussion

We discuss below the results obtained in the study in three aspects: substantiation of the proposed generative model, validity of the information gain as an indicator of BFA performance, and comparison of 5 investigated BFA methods.

8.1 The Generative Model

The generative model of binary signals suitable for BFA 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 the set of highly coherent attributes. To create notions of objects, the brain has to calculate statistics on incoming signals in order to characterize them by some representative variables, the number of which is much smaller than signal dimensionality. It is generally accepted that such a reduction of information redundancy of the incoming signals is one of the main brain function [Marr (1970), Marr (1971), Barlow (1985), Foldiak (1990), Doya (1999)]. We believe that this function is performed in the brain by associative attractor neural networks, which specify the main way of brain functioning [Amit (1992), Kussul (1992)]. In our previous papers [Frolov et al. (2004b), Frolov et al. (2007), Frolov et al. (2009)], we showed that this kind of network (ANNIA) is actually able to extract hidden signal primitives in the form of objects from the incoming signals. This ability is based on the fact that, due to the Hebbian learning, the attributes inherent to the object activate the related neurons simultaneously, and thus create tightly connected groups of neurons. These groups become attractors of the network dynamics, and can be revealed by a simple procedure. In terms of statistics, objects are factors, the attributes characterizing the object define factor loadings, and the presence or absence of an object in scene define factor scores.

We also believe that this kind of signal redundancy is typical for many fields including social science, marketing, zoology, genetics, medicine and others that operate with nominal data. We already successfully applied ANNIA to the analysis of parliament voting [Frolov et al. (2006b)], mushrooms catalogue [Frolov et al. (2008)], and textual data [Frolov et al. (2004a), Frolov et al. (2009)]. In the present paper, we consider the application of BFA methods to the bars problem, which is a well-known benchmark test in image analysis [Foldiak (1990), Spratling (2006), Lücke and Sahani (2008)].

We assume that expression (2.1) provides a rather general form of signal representation, which is a good model defining BFA. 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. Moreover, some keywords are more common for the topic (one can say that they create the topic "kernel") and others are less common (topic "fringe"). 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.

Signals containing certain factor could 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. (2009)]. When noise is absent, BFA is equivalent (compare (2.1) and (6.1)) to Boolean matrix factorization [Belohlavek and Vychodil (2010)]. Since in BFA model, scores are assumed to be nonnegative (1 or 0), it could be related to the methods of Nonnegative Matrix Factorization [Zafeiriou et al. (2006)]. Since factor scores are assumed to be sparse (BFA evidently fails for dense factor scores [Frolov et al. (2007)]), BFA could be related to the methods of Sparse Component Analysis [Georgiev et al. (2005)]. Since factors are assumed to be independently distributed in data set, BFA could be also related to the methods of Independent Component Analysis [Koldovsky et al. (2006)].

The general generative model in the form (2.1) does not imply any specific form of factor distributions in the data set. However, most results were obtained in the present paper under the assumption that factors were distributed independently, i.e., factor scores $\mathbf{S}$ are drawn from a multivariate Bernoulli distribution. If nothing is known in advance concerning distribution of factor scores, it is reasonable to consider them as distributed independently. If something is known concerning their distribution, this information can be taken into account by modifying the respective model formulas.

The generative model (2.1) does not also imply any specific form of noise distribution. However, most formulas were obtained under the assumption that distortion of each attribute in each factor and in each signal occurs independently of other attributes, other factors, specific noise and other signals. We again suppose that such an assumption is reasonable without any a priori information about peculiarities of noise distribution. Note that three of five considered BFA methods (namely BMF, ANNIA and DI) are indifferent to the form of factor or noise distributions, MCA_3 was obtained for another generative model at all, and only EMBFA was developed in this paper using Bernoulli distributions.

8.2 Information Gain

Information gain is a difference between two entropies. First one is calculated for the given signals data set assuming that its factor structure unknown and second one is calculated for that same data set supposing factor structure is revealed by BFA and taken into account. Entropy of data set is obtained as a sum of Shannon entropies of all binary signals on the assumption that signal components are distributed independently. If data set factor structure is ignored, the probability of each component to take value One can be evaluated as the frequency of corresponding Ones in the data set. If data set factor structure is taken into account, this probability for the j-th component and the m-th signal is determined by the factors mixed in the signal, (i.e., by the vector of factor scores $\mathbf{S}_m$ for the signal), and by the probabilities p_{ij} and q_j characterizing factors distortion and specific noise. Since most BFA methods (exception is EMBFA) give only factor loadings and factor scores, but not probabilities p_{ij} and q_j, those probabilities have to be evaluated. For their evaluation, we suggest the procedure based on likelihood maximization. This is one of the most powerful and efficient statistical methods that can be easily implemented for the given generative model of signals, and uses only factor scores, but not factor loadings.

When probabilities p_{ij} and q_j are found and factor scores are given, the probabilities that the j-th component of the m-th signal in the data set takes One are given by (2.2). Note that neither likelihood maximization nor entropy calculation require knowledge of the factor scores distribution. Both procedures use the distribution of factor scores in the given data set provided by BFA.

We have shown that information gain is sensitive both to noise in data and to errors in BFA. When noise increases (in the form of factor distortion or specific factors), information gain decreases and becomes zero or negative even if scores are given precisely. Looking at the bar images with small gain

(as shown, for example, in Fig. 4.2(A)) one might agree that zero gain corresponds to the threshold of our feeling that attempts to detect some hidden but regular structure in the data set are useless. Gain also decreases when some true factors are missing or factor scores are found not correctly. Thus, this gain 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.

It seems amazing that for high noise levels some BFA methods provide information gain that exceeds the gain obtained by precise solution (see, for example, Fig. 4(a)). We have specially analyzed and thoroughly described those exceptions to show that in all cases the factor structure prescribed by BFA to the given data set was quite reasonable, and actually exposed the peculiarities of the data set provoked by noise better than the precise solution. Thus, those special cases only confirm the validity of information gain for estimating BFA performance. In [Frolov et al. (2009)], we compared information gain provided by ANNIA and by Program Committee of one of Neural Networks Conferences for the task of allocating conference papers to topics. It is interesting that for ANNIA the information gain happened to be three times bigger.

8.3 Performance of the BFA Methods

The bars problem is a common benchmark [Spratling (2006), Lücke and Sahani (2008)] to reveal strengths and weaknesses of the BFA methods. Binary Matrix Factorization (BMF) algorithm introduced in [Belohlavek and Vychodil (2010)] is perfect in the absence of noise and seems to be insensitive to the size of data set and to the number of mixed factors C in each signal. However, it fails even for very low noise levels that are easily handled by other methods. This is evidently not the weakness of the algorithm proposed by [Belohlavek and Vychodil (2010)], but the weakness of BMF approach in general, because it implies precise presentation of the data set matrix in the form (6.1). Thus, it treats specific noise and distorted factors as true factors and fails because of fast growth in the number of the found factors vs noise level. Since the number of factors is restricted by the size of data set M, this effect is less manifested for smaller M. As it is difficult to imagine any real data set without noise, the use of BMF is limited.

Dendritic Inhibition (DI) neural network [Spratling (2006)] is sensitive to noise of both kinds and to task complexity C growth. When C increases, the method becomes unstable in the sense that its operation strongly depends on the realization of initial synaptic weights of the basic network. For a set of initial weights DI may converge to true factors, for other initial weight realizations it converges to a random solution. It is especially evident for large C (see Fig. 7.5). Another peculiarity of DI is its sensitivity to hints in data set, i.e.to patterns with only a few mixed factors which seems to facilitate significantly the search for true factors. So, DI fails when the number of mixed factors in each pattern is large and fixed (Fig. 7.5).

Maximal Causes Analysis (MCA_3) [Lücke and Sahani (2008)] is most sensitive to the increase of q_j and C. First, MCA_3 is based on the generative model, which does not take into account specific noise defined by q_j. Second, it is restricted to the case of sparse scores and ignores signals where more than three factors are mixed. Taking into account signals with $C > 3$, however, would require prohibitive computation time. Similar to DI, MCA_3 depends on the choice of parameters and its set of adjustable parameters is rather large. This is also inherent to all methods of Maximal Causes Analysis suggested by [Lücke and Sahani (2008)].

Although Expectation-Maximization Binary Factor Analysis EMBFA proposed in this paper is also restricted to the case of sparse scores ($C \leq 3$), it is less sensitive to the rise of C compared to MCA_3. It gave reasonable results even when $C = 6$ (Fig. 7.5). When $C > 3$, EMBFA treated some bars as specific noise. The peculiarity of EMBFA is the competition between two kinds of noise. When specific noise is large, EMBFA prescribes additional pixels to bars. When factor distortion is large, EMBFA deletes some pixels from bars, adjusting itself to the specific noise realization in the data set. Paradoxically, due to this peculiarity information gain provided by EMBFA often exceeds the ideal gain given by precise solution (Fig. 7.5). Low sensitivity of EMBFA to both kinds of noise and its overall high efficiency is due to using powerful EM approach together with the proposed BFA generative model. In contrast, MCA_3 that is also based on EM is very sensitive to specific noise because it is not a part of its generative model. Note that EMBFA operation does not require any

tuning parameters.

Our methods Attractor Neural Network with Increasing Activity (ANNIA) as well as EMBFA, demonstrate low sensitivity to noise, however ANNIA has a lower sensitivity to the rise of C. It should be noted that in this paper we used a simplified version of ANNIA. Full ANNIA version is described in [Frolov et al. (2007), Frolov et al. (2009)]. In principle, ANNIA is restricted by two limits related to two critical amounts of factors present in a data set [Frolov et al. (2007)]. The first limit defines the condition when factors cease the ability to create attractors of network dynamics. This occurs when the Lyapunov function of spurious attractors becomes equal to that of true attractors. The second limit is given by the condition when number of spurious attractors becomes so large that random search fails to reveal factors. When dimensionality of the signal space N is relatively small, the first limit is crucial. When N increases, the second limit becomes crucial. For the bars problem both limits are inaccessible, so they do not influence an analysis. It can be seen that for majority of practical tasks those limits would not be broken either.

Finally, it is interesting to compare computational complexity of all considered BFA methods. Here and below we estimate it in the limit of large M, L and N. In this limit, the number of operations for BMF is proportional to $\Omega_{BFM} = MLN^2 < n_f >< p_j >$, where $< n_f >$ is mean number of Ones in factors and $< p_j >$ is mean probability of each signal component to be One in a data set. For PC Core2 6400, 2.13 GHz the execution time of one operation in seconds amounts to about 10^{-11}. For all methods, this time was estimated by dividing the total execution time by Ω when M, L and N were sufficiently large so that their doubling resulted in 5% change of estimated value.

The number of operations for DI in one iteration step is approximately proportional to $\Omega_{DI} = 2LMN < p_j >$ and the execution time for one step amounts to about $10^{-7}\Omega_{DI}$. Usually about 15-20 steps are required.

The number of operations required for one iteration step of EMBFA, according to formulas (6.4) and (6.5), is proportional to $\Omega_{EM} = MN(2L)^3 < p_j >$ and execution time for one step amounts to $5 \cdot 10^{-9}\Omega_{EM}$.

For EMBFA the mean number of iteration steps till convergence is about 100. Thus to evaluate the total execution time one must multiply the execution time for one step by a factor of 100.

The number of operations required for one iteration step of MCA_3 is the same as for EMBA. However, the mean time of one operation for PC Core2 2.13 GHz is approximately twice higher than for EMBFA, and the mean number of iteration steps until convergence is about 300. Thus, the total execution time for MCA_3 is six times higher than for EMBFA.

The execution time required for ANNIA is composed of two terms T_1 and T_2. The first time is required to create connection matrix: $T_1 \simeq 3 \cdot 10^{-9} MN^2$ sec. The second time is required to find factors: $T_2 \simeq 3 \cdot 10^{-8} LN < n_f^2 >$ sec.

Note that for BMF and ANNIA L is an actual number of factors unknown in advance. For other methods, the required number of factors is given in advance. It must be set certainly larger than actual number of factors. In DI, EMBFA and MCA_3 it was taken twice higher than the actual number of factors. That is why in formulas for Ω_{DI} and Ω_{EM} we use $2L$ instead of L. For the bars problem $n_f = \sqrt{N}$, $L = 2\sqrt{N}$, and in the absence of noise $< p_j >\simeq C/\sqrt{N}$.

As a whole, to perform BFA for data set of 3200 images of 64-by-64 pixels, containing two not distorted bars, about 460 sec is required for BFM, 110 sec for DI, 240 hours for MCA_3, 40 hours for EMBFA and 200 sec for ANNIA (T_1 = 160 sec, T_2 = 40 sec).

In conclusion, we analyzed strengthes and weaknesses of five most efficient BFA methods. Each of them has a specific application range. BMF is suited and performs well for not distorted data. MCA_3 shows good performance in the absence of specific factors and at small number of factors in a pattern. EMBFA and ANNIA are less sensitive to increasing noise level and number of factors in a pattern. When the amount of factors in a data set is large, computational complexity of EMBFA is higher than that of ANNIA, so in this case ANNIA is preferable. However, for signals of large dimensionality, computational complexity of ANNIA becomes higher than that of EMBFA, and so EMBFA could be preferable. If the properties of a data set are unknown in advance, the best strategy is to perform BFA by all methods, to compare their efficiencies by information gain, and to select the best method for a particular application.

Bibliography

[Amit (1992)] D. J. Amit. *Modeling brain function: The world of attractor neural networks.* Cambridge Univ Press, 1992.

[Barlow (1985)] H. B. Barlow. Cerebral cortex as model builder. In D. Rose and V. G. Dodson, editors, *Models of the visual cortex*, pages 37–46. Wiley, Chichester, 1985.

[Belohlavek and Vychodil (2010)] 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*, 76(1):3–20, 2010.

[Dempster et al. (1977)] 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)*, 39(1):1–38, 1977.

[Doya (1999)] K. Doya. What are the computations of the cerebellum, the basal ganglia and the cerebral cortex? *Neural networks*, 12(7-8):961–974, 1999.

[Foldiak (1990)] P. Foldiak. Forming sparse representations by local anti-hebbian learning. *Biological Cybernetics*, 64:165170, 1990.

[Frolov et al. (2004a)] A. A. Frolov, D. Husek, P. J. Polyakov, H. Rezankova, and V. Snasel. Binary Factorization of Textual Data by Hopfield-Like Neural Network. In *Proc. Computational Statistics (Compstat'04)*, pages 1035–1041, Prague, Czech Republic, August 2004.

[Frolov et al. (2004b)] 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*, 14:139–152, 2004.

[Frolov et al.(2006a)] A. A. Frolov, D. Husek, P. Polyakov, and H. Rezankova. New Neural Network Based Approach Helps to Discover Hidden Russian Parliament Voting Patterns. In *2006 IEEE World Congress on Computational Intelligence (WCCI'06)*, pages 6518–6523, Vancouver, Canada, July 2006.

[Frolov et al. (2006b)] A. A. Frolov, D. Husek, P. J. Polyakov, and H. Rezankova. Binary Factorization of Textual Data by Hopfield-Like Neural Network. In *Proc. Computational Statistics (Compstat'06)*, pages 1035–1041, Roma, Italy, August 2006.

[Frolov et al. (2007)] A. A. Frolov, D. Husek, I. P. Muraviev, and P. Y. Polyakov. Boolean factor analysis by attractor neural network. *IEEE Transactions on Neural Networks*, 18(3):698–707, 2007.

[Frolov et al. (2009)] 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*, 20(7):1073–1086, 2009.

[Frolov et al. (2009)] Alexander Frolov, Dusan Husek, and Pavel Polyakov. Estimation of Boolean factor analysis performance by informational gain. In *Proceedings of the 6th Atlantic Web Intelligence Conference (AWIC'2009)*, pages 83–94, Praha, Czech Republic, September 2009.

[Frolov et al. (2006c)] Alexander A. Frolov, Dusan Husek, Pavel Polyakov, and Hana Rezankova. New Neural Network Based Approach Helps to Discover Hidden Russian Parliament Voting Patterns. In *IEEE International Joint Conference on Neural Networks*, pages 6518–6523, 2006.

[Frolov et al. (2008)] Alexander A. Frolov, Dusan Húsek, Hana Rezanková, Václav Snásel, and Pavel Polyakov. Clustering variables by classical approaches and neural network Boolean factor analysis. In *IEEE International Joint Conference on Neural Networks*, pages 3742–3746, 2008.

[Ganter et al. (1999)] B. Ganter, R. Wille, and R. Wille. *Formal concept analysis*. Springer Berlin, 1999.

[Georgiev et al. (2005)] P. Georgiev, F. Theis, and A. Cichocki. Sparse component analysis blind sourse separation of underdetermined mixtures. *IEEE Transactions on Neural Networks*, 16(4): 992–996, 2005.

[Hoyer (2004)] P. O. Hoyer. Non-negative matrix factorization with sparseness constrains. *Journal of Machine Learning Research*, 5:1457–1469, 2004.

[Koldovsky et al. (2006)] 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*, 17(5):1265–1277, 2006.

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

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

[Marr (1970)] D. Marr. A Theory for Cerebral Neocortex. *Proceedings of the Royal Society of London. Series B, Biological Sciences (1934-1990)*, 176(1043):161–234, 1970.

[Marr (1971)] D. Marr. Simple Memory: A Theory for Archicortex. *Philosophical Transactions of the Royal Society of London. Series B, Biological Sciences (1934-1990)*, 262(841):23–81, 1971.

[Miettinen et al. (2008)] P. Miettinen, T. Mielikainen, A. Gionis, G. Das, and H. Mannila. The discrete basis problem. *IEEE Transactions on Knowledge and Data Engineering*, 20(10):1348–1362, 2008.

[Neal and Hinton (1998)] R. M. Neal and G. E. Hinton. A view of the EM algorithm that justifies incremental, sparse, and other variants. *Learning in graphical models*, 89:355–368, 1998.

[Spratling (2006)] M. W. Spratling. Learning image components for object recognition. *Journal of Machine Learning Reasearch*, 7:793–815, 2006.

[Spratling and Johnson (2002)] M. W. Spratling and M. H. Johnson. Preintegration lateral inhibition enhances unsupervised learning. *Neural Computation*, 14(9):2157–2179, 2002.

[Spratling and Johnson (2003)] M. W. Spratling and M. H. Johnson. Exploring the functional significance of dendritic inhibition in cortical pyramidal cells. *Neurocomputing*, 52(54):389–395, 2003.

[Veiel (1985)] H. O. Veiel. Psychopathology and Boolean Factor Analysis: a mismatch. *Psychol Med*, 15(3):623–628, 1985.

[Weber and Scharfetter (1984)] A. C. Weber and C. Scharfetter. The syndrome concept: history and statistical operationalizations. *Psychol Med*, 14(2):315–325, 1984.

[Zafeiriou et al. (2006)] 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*, 17(3):683–695, 2006.

Acknowledgement: We wish to thank to Mr. Rachkovsky for a careful reading of the whole work and for stimulating comments, which helped us to improve both the clarity and quality of the work. We would also like to thank the institutions that provide financing of our projects codenamed AV0Z10300504, GACR P202/10/0262, GACR 205/09/1079, and GA MŠk(CZ) 1M0567.