Boolean Factor Analysis by Attractor Neural Network

Summary of Ph.D. Thesis

Author:  
Mgr. Pavel Y. Polyakov
Thesis supervisor: Ing. Dušan Húsek, CSc.
Study program: P1807 Computer Science, Communication Technology and Applied Mathematics
Field of study:1801V001 - Informatics

Ostrava, May 2017
Boolean Factor Analysis by Attractor Neural Network

Summary of Ph.D. Thesis

Author: Mgr. Pavel Y. Polyakov
Thesis supervisor: Ing. Dušan Húsek, CSc.
Study program: P1807 Computer Science, Communication Technology and Applied Mathematics
Field of study: 1801V001 - Informatics

Ostrava, May 2017
Boolean Factor Analysis by Attractor Neural Network

Summary of Ph.D. Thesis

Author: Mgr. Pavel Y. Polyakov
Thesis supervisor: Ing. Dušan Húsek, CSc.
Study program: P1807 Computer Science, Communication Technology and Applied Mathematics
Field of study: 1801V001 - Informatics

Ostrava, May 2017
Konání obhajoby disertační práce:
VŠB TU Ostrava, Fakulta elektrotechniky a informatiky
Katedra informatiky
17. listopadu 15, 708 33 Ostrava–Poruba
v pátek 28. dubna 2017 v 12:30 hodin
v zasedací místnosti EA453

Oponenti:
Prof. Ing. Mirko Novák, Dr.Sc.
doc. Ing. Jan Platoš, Ph.D.
RNDr. Roman Neruda Ph.D.
Abstract

Methods for the discovery of hidden structures of high-dimensional binary data are one of the most important challenges facing the community of machine learning researchers at present. There are many approaches in the literature that try to solve this hitherto rather ill-defined task. The Boolean factor analysis (BFA) studied in this work represents hidden structure of binary data as Boolean superposition of binary factors complied with BFA generative model of signals, and the criterion of optimality of BFA solution is given. In these terms, the BFA is well-defined task completely analogous to linear factor analysis. The main contributions of the dissertation thesis are as follows: Firstly developed was the efficient BFA method based on original attractor neural network with increasing activity (ANNIA), which is subsequently improved by combination with expectation-maximization method. Secondly the analysis of characteristics of the ANNIA that are important for method functioning were performed. Then functioning of both methods was validated on artificially generated datasets. Next the application of method to real-world data from different areas of science was carried out to demonstrate their contribution to this type of analysis. Lastly the comparison of the BFA methods with related methods was done including applicability analysis.

Keywords

Boolean factor analysis, data mining, statistics, dimension reduction, attractor neural network, Hopfield neural network, Hebbian learning rule, information gain, dimension reduction, likelihood-maximization, expectation-maximization
## Contents

1 Introduction 3

2 **Formal definition of Boolean factor analysis** 5  
   2.1 Decomposition of signals in Boolean factor analysis 5  
   2.2 Statistical model of binary signals 6  
   2.3 Criterion for solution of Boolean factor analysis 8  
   2.4 Bars problem — illustrative example of Boolean factor analysis 9

3 **Neural network based method for Boolean factor analysis** 10  
   3.1 Attractor neural network with increasing activity – ANNIA . 11  
   3.2 Likelihood maximization 14  
   3.3 Hybrid ANNIA and likelihood maximization 15

4 **Properties of attractor neural network with increasing activity** 16  
   4.1 Lyapunov function of true attractors 17  
   4.2 Global spurious attractors 18  
   4.3 The size of factors attraction basins 20  
   4.4 Probability of true trials during random searches 24

5 **Application of hybrid ANNIA and likelihood maximization method** 26  
   5.1 Comparison with other methods in solving the bars problem 26  
   5.2 Application to text datasets 29  
      5.2.1 Analysis of the proceedings of the IJCNN and Neuroinformatics conferences 30  
      5.2.2 Analysis of the Reuters R52 dataset of news messages 33  
   5.3 Application to the Genome dataset analysis 33

6 Conclusion 36

References 38

**Author’s selected publications related to the topic of the thesis** 39
1 Introduction

Factor analysis is one of the most effective methods of elimination of information redundancy of multidimensional signals. The essence of the idea of information redundancy is intuitively clear: during everyday life human, like all living beings, deals with regular in space and time information structure. Due to this regularity, some characteristics of the environment, which are referred to as features, are encountered concurrently, i.e., the probability of their concurrent observation is considerably higher than the probability of random observation. The group of concurrent features is called as a factor. Encoding of signals using factors requires smaller volume of information and this means that the signals that arrive to the brain carry redundant information.

The idea of Boolean factor analysis (BFA) follows from the studies by Barlow and Marr, the outstanding neurophysiologists of the last century. According to Barlow [1], as Foldiak [2] said, “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 metre’, ‘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”. Obviously, objects are factors and the scene observed is incoming signal that is decomposed into factors. Both objects and scene are presented in binary code, therefore, operations with them may be performed in the frameworks of Boolean logics.

Binary presentation of data is quite typical of many fields including sociology, marketing, zoology, genetics, and medicine [3]. However, in contrast to linear one, BFA is only weakly developed. Those who need to use factor analysis for binary data have several possibilities. If data have small dimension, it is possible to find binary factors using brute force search. In this case the problem is NP-complete [4, 5], and for analysis of binary data of large dimension approximate methods are required. The more widespread approach is linear factor analysis followed by binarization of solution [6]. Of
course, like all methods based on linear algebra, it uses completely different model of signals, hence, it may be used cautiously and in narrow range of parameters when non-linearity of the initial data is small, for example, if one signal has only one factor or factors do not intersect.

A good alternative to linear approach is method of Boolean matrix factorization [7, 8]. Its advantage over linear methods is that initial data are decomposed into factors according to Boolean arithmetics. However, factor analysis could not be reduced to formal decomposition of data matrix. The most substantial limitation of this approach is that it cannot be used for analysis of noisy data.

Recently, several methods have been proposed for BFA including neural network based method of Spratling [9]. Unfortunately, efficacy of these methods for solution of BFA was not examined on data of large dimension where their use would be really rewarding. The above suggests that, despite importance of task, there are no good theoretically based BFA methods that can work with the data of large dimension.

Thus, on one hand, there are no good methods for BFA; on the other hand, there are strong arguments that the brain have to somehow separate factors. In this situation it is reasonable to address neurophysiology and look for the brain structures that would be able to have this function. According to widespread two-stage model of learning, new information that comes to the brain is initially kept in temporal storage and, then, interpreted, processed, and filtered information is directed to the neocortex for long-term storage. Numerous experimental results led to conclusion that the hippocampus, a phylogenetically ancient brain structure, plays a key role in consolidation, processing and storage of information. Due to an extensive system of recurrent collaterals, which provide self-excitation of group of neurons of one layer, the CA3 field can keep associatively linked patterns of information which allows the hippocampus to function as temporal storage of incoming information. According to two-stage model of learning, the desired procedure of separation of factors should be located somewhere between recording of information into temporal memory and its transfer to the long-term memory. The first candidate for the structure that is responsible for separation of factors is CA3 field.

Studies of Hopfield network, whose natural prototype is CA3 field, led to conclusion that its capacities are beyond the frameworks of autoassociative memory. For example, if a set of patterns that are close to each other is recorded into the Hopfield network, a pattern-prototype will appear which is somewhat average of these patterns [10]. It appears that associative neural network, a Hopfield network with sparse coding, can reveal factors [10].
This work is focused on study of this question. The main goal of Ph.D. thesis is to develop an efficient method for BFA. Since BFA is not generally known koncept, to avoid any ambiguity, in Chapter 2 of the thesis the formal definition of BFA based on the information criteria of optimality of solution is given. Then after an overview of related to BFA state-of-the-art methods presented in Chapter 3, a new method capable of solving BFA is suggested in Chapter 4 (the whole algorithm is given in Appendix D). The method is based on an original Hopfield-like attractor neural network with increasing activity. Since this network is new and little studied, the secondary objective of this work is to study the properties of the network. The corresponding results are presented in Chapter 5 and Appendices A-C. Another secondary objective of the thesis is to evaluate the practical value of the proposed method. For this purpose, the method was applied to various natural datasets. The description of the obtained results, as well as a comparison with the results obtained by other methods is given in Chapter 6. Finally, in Chapter 7, we provide a short conclusion and summary of the thesis.

2 Formal definition of Boolean factor analysis

2.1 Decomposition of signals in Boolean factor analysis

Boolean factor analysis implies that each observed signal \( x = [x_1, x_2, \ldots, x_N] \) of \( N \) binary components can be presented as Boolean superposition of \( L \) factors:

\[
x = \bigvee_{i=1}^{L} s_i \land f_i \lor u.
\]

where factor score \( s_i \) gives a contribution of \( i \)th factor to the signal, the vector of factor loadings \( f_i = [f_{i1}, f_{i2}, \ldots, f_{iN}] \) represents the \( i \)th factor in the original \( N \)-dimensional signal space and \( u = [u_1, u_2, \ldots, u_N] \) designates the row vector of residual part of the signal, namely the part of signal that is not described by factors. The factor scores, factor loadings and residuals are also binary. Symbol \( \lor \) denotes Boolean sum \( (\lor x_i = x_1 \lor x_2 \lor \cdots \lor x_L), \lor - \) operator OR, \( \land - \) operator AND.

In matrix form, binary matrix \( X \) of dimension \( M \times N \) decomposes into binary matrix of factor scores \( S \) of dimension \( M \times L \) and binary matrix of
factor loadings $\mathbf{F}$ of dimension $L \times N$:

$$\mathbf{X} = \mathbf{S} \odot \mathbf{F} \oplus \mathbf{U},$$

where $\mathbf{U}$ is binary matrix of residuals, $\odot$ denotes Boolean matrix multiplication and $\oplus$ denotes Boolean matrix summation. The operations of Boolean matrix multiplication and Boolean matrix summation are similar to matrix multiplication and matrix summation for linear algebra, except that bitwise AND and bitwise OR are used instead of common componentwise product and componentwise sum.

### 2.2 Statistical model of binary signals

On the basis of generative model for linear factor analysis [11], it is possible to define analogous model for binary signals. According to the model, every component of signal $\mathbf{x}$ takes 1 or 0, depending on the presence or absence of the related attribute (observed feature). Each factor can be represented by a binary raw vector of factor loadings $\mathbf{f}_i = [f_{i1}, f_{i2}, \ldots, f_{iN}]$ where one valued entries correspond to highly correlated attributes of the $i$th object and zero valued entries correspond to attributes not constituting the object. Although the probability of the object’s attributes appearing in a pattern simultaneously with its other attributes is high, it is not necessarily equal to 1. For example, the attribute “has a tail” does not always appear with the appearance of the object “dog”. Let us denote this probability by $p_{ij}$, where $j$ is the index of the attribute and $i$ is the index of the factor. For attributes constituting the factor, i.e., for attributes with $f_{ij} = 1$, the probability $p_{ij}$ is high, and for the other attributes (with $f_{ij} = 0$), it is zero.

As in linear factor analysis, let us suppose that in addition to common factors $\mathbf{f}_i$ that influence more than one attribute, each signal also contains $N$ specific or unique factors that influence only particular attributes. Specific factors are also called “specific noise.” The contribution of specific factors is defined by a binary row vector $\mathbf{\eta} = [\eta_1, \eta_2, \ldots, \eta_N]$ of dimension $N$. Each specific factor $\eta_j$ is characterized by the probability $q_j$ with which $\eta_j$ takes on the value 1. In the same way, each common factor $\mathbf{f}_i$ is characterized by the probability $\pi_i$ that it appears in a signal, i.e., factor score $s_i$ takes 1 with probability $\pi_i$. The probabilities $p_{ij}$, $q_j$ and $\pi_i$ are parameters of BFA generative model of signals: $\Theta = (p_{ij}, q_j, \pi_i, i = 1, \ldots, L, j = 1, \ldots, N)$.

As a result, instead of using equation 1, any observed signal $\mathbf{x}$ can be presented in the form

$$\mathbf{x} = \bigvee_{i=1}^{L} s_i \land \mathbf{f}_i^{ij} \lor \mathbf{\eta}$$

(3)
where \( s = [s_1, s_2, \ldots, s_L] \) is a binary row vector of factor scores of dimension \( L \), \( L \) being the total number of factors, \( f'_i = [f'_{i1}, f'_{i2}, \ldots, f'_{iN}] \) is a distorted version of factor \( f_i \) and \( \eta \) is a binary row vector of specific factors. Factor distortion implies that one valued entries of \( f_i \) can transform to 0 with probability \( 1 - p_{ij} \) before mixing in the observed pattern but none of the zero valued entries \( f_i \) can take 1 in the distorted version of the factor because the probability for them to transform to 1 is zero (\( p_{ij} = 0 \)).

It is supposed 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 \( x \) to take the value \( x_j \) is

\[
P(x_j | s, \Theta) = x_j - (2x_j - 1)(1 - q_j) \prod_{i=1}^{L} (1 - p_{ij})^{s_i},
\]

where scores \( s_i \) are assumed to be given. Different components of \( x \) (attributes) are also supposed to be statistically independent, and thus

\[
P(x | s, \Theta) = \prod_{j=1}^{N} P(x_j | s, \Theta).
\]

BFA is performed on the set \( \mathcal{X} \) of patterns \( x_m \) containing \( M \) representatives. Instantiation of vectors \( f'_i, i = 1, \ldots, L \) and \( \eta \) for some pattern \( x_m \) is assumed to be independent of other patterns, and thus

\[
P(\mathcal{X}) = \prod_{m=1}^{M} P(x_m).
\]

In most cases, it is fair to assume that factors appear in patterns independently of each others, then

\[
P(s | \Theta) = \prod_{i=1,L}^{i} \pi_i^{s_i} (1-\pi_i)^{1-s_i}.
\]

The aim of Boolean factor analysis is to find the parameters of the generative model \( \Theta \) and factor scores \( s_m \) \((m = 1, \ldots, M)\) for all \( M \) patterns \( x_m \) of the observed dataset. However, it is supposed that the factors found could also be detected in any arbitrary pattern if generated by the same model. Note that the finding of \( p_{ij} \) implies the finding of factor loadings \( f_{ij} \) since \( f_{ij} = \text{sgn}(p_{ij}) \).
2.3 Criterion for solution of Boolean factor analysis

Encoding of signals with the use of factors decreases information redundancy of signals and helps to use smaller volume of information for their storage. It is easy to show that elimination of information redundancy of signals also occurs during minimization of the number of factors at fixed $|U|$ and during minimization of the number of non-zero components in the residue matrix $|U|$ at fixed $L$. Information gain is the general information theoretic measure that could be used for estimation of successfulness of information redundancy elimination. It can be calculated as the difference of two entropies. The first is the entropy of a dataset when its hidden factor structure is unknown, and the second is the entropy when it is revealed and taken into account. If the factor structure of the signal space is unknown, then representing the $j$th component of vector $x$ requires $h(p_j)$ bits of information, where $h(x) = -x \log_2 x - (1 - x) \log_2(1 - x)$ is the Shannon function and $p_j$ is the probability of the $j$th component’s taking 1. Representing the whole dataset requires

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

bits of information. If the hidden factor structure of the signal space is detected, i.e., the parameters of the generative model $\Theta$, factor scores $s_m(m = 1, \ldots, M)$ and factor loadings $f_i(i = 1, \ldots, L)$ are found, then representing the whole dataset requires

$$H = H_1 + H_2 + H_3$$

bits of information. Here $H_1 = M \sum_{i=1}^{L} h(\pi_i)$ is the information required to represent the factor scores, $H_2 = L \sum_{j=1}^{N} h(r_j)$ is the information required to represent the factor loadings, $H_3 = \sum_{m=1}^{M} \sum_{j=1}^{N} h(P(x_{mj}|s_m, \Theta))$ is the information required to represent all patterns of the dataset when factor scores and loadings are given, $r_j$ is the probability of the $j$-th component’s taking 1 in factor loadings, $P(x_{mj}|s_m, \Theta)$ is given by (4). The information gain is determined by the difference between $H_0$ and $H$. Relative information gain defined as

$$G = (H_0 - H)/H_0$$

is used as criterion for solution of BFA.

The magnitude $G$ may vary in the range from $-\infty$ to 1. The positive $G$ means that the encoding of signals with the use of factors is more effective
as compared to the initial encoding. If $G$ is close to zero or negative then either the selection of factors was wrong or factor analysis in the frameworks of the postulated generative model is senseless because this model is inadequate to the internal structure of the data analyzed. As shown in [12], the relative information gain reaches maximum when BFA solution is right, i.e., all scores and generative model parameters used for artificially generated dataset are found correctly. It decreases when both the difference between the resulting BFA solution and the right solution increases and noise in data increases. Thus, information gain has proved to be a reliable measure for detecting the presence of hidden factor structures in a given dataset and for comparing efficiency of different BFA methods as well.

2.4 Bars problem — illustrative example of Boolean factor analysis

The well-known benchmark for learning of objects from complex patterns is the Bars Problem (BP) introduced by Foldiak [2]. In this problem, each pattern of the dataset is $n$-by-$n$ binary pixel image containing several of $L = 2n$ possible (one-pixel wide) horizontal and vertical bars (Fig. 1).

Pixels constituting a bar take the value 1 and pixels not constituting it take the value 0. For each image, each bar could be chosen with the probability $C/L$, where $C$ is the mean number of bars mixed in an image. At the point of intersection of a vertical and a horizontal bar, the pixel takes the value 1. This Boolean summation of pixels belonging to different bars simulates the occlusion of objects. The task is to recognize all bars as individual objects, exploring a dataset 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. Factor loadings $f_{ij}$ ($j = 1, \ldots, N$) take
value 1 for pixels constituting the $i^{th}$ bar and value 0 for pixels not constituting it. Each image is a Boolean superposition of factors, and the factor score takes the value 1 or 0 depending on the presence or absence of a bar in the image. Thus, the bars problem is a special case of BFA.

3 Neural network based method for Boolean factor analysis

The idea of using the Hopfield-like neural network for solving the BFA problem is based on the well known property of Hopfield network to create attractors of the network dynamics by assemblies of tightly connected neurons. If $N$-dimensional binary patterns of the signal space are interpreted as activities of $N$ binary neurons (1 – active, 0 – nonactive) then neurons representing a factor are activated simultaneously each time when the factor appears in the patterns of the dataset, and neurons representing different factors are rather seldom activated simultaneously. Therefore, due to the Hebbian learning rule, the factor neurons become more tightly connected than the other neurons. So factors can be revealed as attractors of the network dynamics.

Attractor neural network with increasing activity (ANNIA) presented in Section 3.1 reveals factors as groups of tightly connected neurons. It is shown in the thesis that ANNIA is able to reveal all factors in the wide range of parameters of input data so long as restrictions derived in Sections 4.3 and 4.4 are satisfied. Although it provides an accurate estimation of the factor loadings (as set of active neurons in true attractor), but it provides only approximate estimation of factor scores, and no estimation of the parameters of the generative model $p_{ij}$ and $q_{j}$. A way to overcome this drawback is to combine ANNIA with likelihood maximization (LM) described in Section 3.2. It is shown in the thesis that the LM procedure itself is able to provide complete solution of the BFA problem but requires an appropriate initial approximation. If it starts from the random initial parameters it commonly fails. In the combination of ANNIA and LM the role of ANNIA is to provide LM with the initial approximation. Another aspect of the ANNIA and LM interaction is a suppression of the dominant attractors in ANNIA using the data provided by LM. The resulting hybrid ANNIA and LM procedure called as LANNIA is presented in Section 3.3.
### 3.1 Attractor neural network with increasing activity – AN-NIA

The proposed neural network consists of N neurons of the McCulloch-Pitts type with gradually ranged synaptic connections between them. Only a fully connected case is considered here. Each pattern of the learning set $x_m$ is stored in the matrix of synaptic connections $J'$ according to the modified Hebbian rule:

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

(11)

where $M$ is the number of patterns in the learning set and bias $a_m = \frac{\sum_{i=1}^{N} x_{mi}}{N}$ is the total relative activity of the $m$-th pattern.

Under conditions discussed in Section 4.2, two global spurious attractors dominate in the network dynamics. To suppress their dominance one special inhibitory neuron is added to the N principal neurons of the Hopfield network. As shown in the thesis, adding the inhibitory neuron is equivalent to subtraction of $J''$ from $J'$ where

$$J''_{ij} = M(b_i - \bar{b})(b_j - \bar{b}), \ i, j = 1, \ldots, N, i \neq j, \ J''_{ii} = 0,$$

(12)

$b_i = \frac{\sum_{m=1}^{M} x_{mi}}{M}$ is a mean activity of the $i$-th neuron in the learning set and $\bar{b}$ is a mean activity of all neurons in the learning set (note $<a_m> = \bar{a} = \bar{b}$). The resulting connection matrix is defined as

$$J = J' - J''.$$  

(13)

The following two-run recall procedure is used to reveal factors. Its initialization starts by the presentation of a random initial pattern $x^{in}$ with $k_{in} = r_{in}N$ active neurons. Activity $k_{in}$ is supposed to be smaller than the activity of any factor. On presentation of $x^{in}$, network activity $x$ evolves to some attractor. This evolution is determined by the synchronous discrete time dynamics. At each time step:

$$x_i(t + 1) = \Theta(h_i(t) - T(t)), \ i = 1, \ldots, N, \ x_i(0) = x^{in}$$

(14)

where $h_i$ are components of the column vector of synaptic excitations

$$h(t) = Jx(t),$$

(15)

$\Theta$ is a step function, and $T(t)$ is an activation threshold. At each time step of the recall process the threshold $T(t)$ is chosen in such a way that the
level of the network activity is kept constant and equal to \( k_{in} \). Thus, on each time step \( k_{in} \) “winners” (neurons with the greatest synaptic excitation) are chosen and only they are active on the next time step. This choice of activation threshold enables the network activity to stabilize in point or cyclic attractors of length two \([10]\).

When activity stabilizes at the initial level of activity \( k_{in} \), \( k_{in} + 1 \) neurons with maximal synaptic excitation are chosen 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 \( k_{fin} = r_{fin}N \) where \( r = k/N \) is a relative network activity. Thus, one trial of the recall procedure contains \((r_{fin} - r_{in})N\) external steps and several internal steps (usually 2-3) inside each external step to reach an attractor for a given level of activity.

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

\[
\lambda = x^T(t + 1)Jx(t)/k,
\]

where \( x^T(t + 1) \) and \( x(t) \) are two network states in a cyclic attractor (for point attractor \( x^T(t + 1) = x(t) \)). In this definition, the value of the Lyapunov function gives a mean synaptic excitation of neurons belonging to an attractor at the end of each external step. The identification of factors is based on the analysis of the dynamics of the Lyapunov function \( \lambda(k) \) and the activation threshold \( T(k) \) in the recall procedure. Figure 2(a) demonstrates changes of the Lyapunov function along the trials of the recall procedure. Trajectories of the network dynamics in Fig. 2(a) form two separated groups. The trajectories with higher Lyapunov function values are called as true, they pass through network states coincided with factors. To identify factor on a trajectory we subtract the activation threshold from the Lyapunov function calculating \( R(k) = \lambda(k)/(k - 1) - T(k)/k \) and its derivative \( R'(k) = R(k) - R(k - 1) \). As shown in Fig. 2(b), the curves for \( R' \) have distinctly exposed peak at the point corresponding to one of the factors. The maximum of \( R'(k) \) is used as an indicator of the factor on each recall trajectory. The state of the neural network activity at the maximum gives the factor loadings \( f \) for the found factor.

As shown in Fig. 2(a), sometimes Lyapunov function jumps up from one to another continuous trajectory. In this step, the network activity transits to an attractor far from the attractor at the previous step. Such a transition could also produce a peak of \( R' \) (Fig. 2(b)). To avoid falsely treating such transition as factors, the similarity \( Sim(k) \) between the patterns of the network activity in the current attractor \( x_{attr}(k) \) and in the previous attractor
Figure 2: Lyapunov function $\lambda$ (a) and derivative of function $R = \lambda/(k-1) - T/k$ (b) in dependence on the number of active neurons $k$. Dashed lines in (a) are thresholds for separating true and spurious trajectories at the beginning (upper line) and at the end (lower line) of the recall procedure. The results were obtained for the 8x8 bars problem dataset ($\sum_j f_{ij} = 8$ for all $i$).

$x_{\text{attr}}(k-1)$ is calculated as

$$Sim(k) = \frac{c - (k-1)k/N}{(k-1)(1-k/N)},$$

(17)

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

The network dynamics can converge not only to the true attractors corresponding to the factors, but also to spurious attractors far from all factors. The Lyapunov function for the spurious attractors is smaller than that for factors (Fig. 2(a)). The trajectories with lower Lyapunov function values are called as spurious. To separate the true attractors from the spurious ones, the following heuristic method is used. For each $k$ a random set of $k$ neurons is activated and then the maximal synaptic excitation over all neurons of the network is calculated. After repeating this procedure 100 times, mean $m(k)$ and standard deviation $\sigma(k)$ of the maximal excitations are calculated.
If the Lyapunov function in the peak of $R'$ along the trajectory satisfies the following inequality

$$\lambda(k) > h_{\text{max}}(k) = m(k) + 3\sigma(k),$$  

the found point on the trajectory is treated as a factor, in the opposite case — as a spurious state. The borders $h_{\text{max}}$ separating true and spurious trajectories are shown in Fig. 2(a) by the dashed lines. The upper curve corresponds to the beginning of the recall procedure when the first factor with the highest Lyapunov function was found. The lower curve corresponds to the end of the recall procedure when the last factor with the lowest Lyapunov function was found and unlearned.

### 3.2 Likelihood maximization

Since the generative model is defined in terms of probabilities one can set up likelihood function and find generative model parameters by likelihood maximization. For BFA generative model, the dataset likelihood takes the form

$$\Lambda = \sum_{m=1}^{M} \Lambda_m, \text{ where } \Lambda_m = \log[P(s_m|\Theta)P(x_m|s_m, \Theta)],$$  

$P(x_m|s_m, \Theta)$ and $P(s_m|\Theta)$ are given by (5) and (7), correspondingly.

To maximize $\Lambda$ the iterative procedure could be used that is very similar to the classic expectation maximization (EM) procedure. The iterations alternately increase $\Lambda$ with respect to a set of factor scores $s_{mi}$ ($m = 1, \ldots, M$, $i = 1, \ldots, L$), while holding $\Theta$ fixed (the E-step), and with respect to parameters of the model $\Theta$, while holding $s_{mi}$ fixed (the M-step).

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

$$\Delta p_{ij} = \gamma_{ij} \frac{\partial \Lambda}{\partial p_{ij}}, \quad \Delta q_j = \gamma_j \frac{\partial \Lambda}{\partial q_j},$$  

where $\gamma_{ij}$ and $\gamma_j$ are positive learning rates and

$$\frac{\partial \Lambda}{\partial p_{ij}} = \sum_{m=1}^{M} \frac{(x_{mj} - P(x_{mj}|s_m, \Theta))}{P(x_{mj}|s_m, \Theta)} \frac{s_{mi}}{1-p_{ij}},$$

$$\frac{\partial \Lambda}{\partial q_j} = \sum_{m=1}^{M} \frac{(x_{mj} - P(x_{mj}|s_m, \Theta))}{P(x_{mj}|s_m, \Theta)} \frac{1}{1-q_j}.$$
Since the probabilities $p_{ij}$ are assumed to be sufficiently high for the components constituting the $i$-th factor ($f_{ij} = 1$) and equal to zero for the other components ($f_{ij} = 0$), at each iteration cycle of step $M$, $p_{ij} = 0$ if $p_{ij} < 1 - \prod_{l \neq i} (1 - \pi_l p_{lj})$, where the right side of the inequality is the probability that the $j$-th attribute appears in the pattern due to other factors except $f_i$. The learning rates in (20) are set to be

$$
\gamma_{ij} = \frac{p_{ij}(1 - p_{ij})}{(M \pi_i)}, \quad \gamma_j = \frac{q_j(1 - q_j)}{M}.
$$

The iterative procedure (20) at each $M$-step continues until $\sum_{i,j} |\Delta p_{ij}|/\sum_{i,j} p_{ij} < 10^{-5}$.

The generative model parameters $p_{ij}$ and $q_j$ obtained at the $M$-step are used as the input for the next E-step to find factor scores. For each individual signal $x_m$ of the dataset, factor scores $s_m$ can be found as those maximizing $\Lambda_m$ using the following procedure. At each iterative step the values $\Lambda_m|_{s_{mi}=1}$ and $\Lambda_m|_{s_{mi}=0}$ obtained by substituting $s_{mi} = 1$ and $s_{mi} = 0$ into (19) given $p_{ij}$, $q_j$ and $\pi_i$ are compared. The value of $s_{mi}$ that provides the greater $\Lambda_m$ is chosen and the procedure goes to another $i$ until it converges. To compute $s_m$ the following two-run iterative procedure is used. At each external cycle of the procedure all components of $s_m$ are processed to maximize $\Lambda_m$. Then the sequence of their processing is randomly permuted at each cycle. The procedure is terminated when $s_m$ remained the same at the next cycle. The procedure converges because at each iterative step the likelihood function does not decrease. The procedure starts with all $s_{mi} = 0$. After computing $s_m$ the procedure is applied to the next signal $x_{m+1}$ until the dataset is exhausted. The probabilities $\pi_i$ are estimated at the end of E-step as frequencies of $i$-th factor appearance in the dataset.

The scores found at the E-step are used as input to the next M-step. If for some factor, all found loadings or scores are zeros, this factor is excluded from the list of found factors. The LM-iterative procedure terminates when the increment of the $\Lambda$ at the next step does not exceed $10^{-6}MN$ or the number of steps in the LM procedure reaches 10.

### 3.3 Hybrid ANNIA and likelihood maximization

In hybrid method ANNIA and LM procedures are performing successively. It starts from ANNIA step. Then factors revealed by ANNIA in several trials (usually, 10–50) are used to initiate LM. The LM procedure is initiated from E-step. The probabilities $p_{ij}$ required to start LM procedure are estimated as $p_{ij} = \max_{j'=1...N} \{h_{ij'}\}$ if $h_{ij} > 0$, $p_{ij} = 0$ if $h_{ij} \leq 0$, where $h_i = f_j f_{ij}$ is a row vector...
of synaptic excitations of neurons when the $i$-th found factor is activated in the network.

After the LM convergence, each $i$-th found factor is unlearned from AN-NIA by subtracting the matrix $\Delta J^i$ from the matrix of synaptic connections $J$ where

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

(22)

where $p^0_{ij}$ is a probability that the $j$-th component takes One in signals not containing $i$-th found factors. Probabilities $\pi_i$, $p_{ij}$ and $p^0_{ij}$ are obtained from the LM procedure. The modified matrix of synaptic connections is used in next ANNIA step for searching the other factors with lower Lyapunov function.

The parameters of the generative model and the factor scores obtained by LM are used to calculate the information gain provided by all found factors. The LANNIA continues until $G$ stops to increase due to adding new found factors. This is the first criterion to terminate LANNIA. The border $h_{\text{max}}$ defined in (18) separates true and spurious trajectories. The appearance of only spurious attractors in the recall procedure indicates that all factors are found. This is the second termination criterion of LANNIA. The algorithm in details is provided in the thesis.

The execution time required for ANNIA is composed of two times. The first time $T_1$ is required to create connection matrix. In the limit of large $M$, $L$ and $N$, $T_1 \approx 10^{-9}MN^2$. The second time is required to find factors: $T_2 \approx 10^{-8}LN\langle n_f \rangle^2$, where $n_f$ is the mean number of ones in factors. For LANNIA additional time $T_3 \approx 10^{-7}ML\langle n_f \rangle$ is required to perform the procedure of likelihood maximization. Coefficients in these formulas are calculated for PC Core2 6400, 2.13 GHz.

4 Properties of attractor neural network with increasing activity

The proposed attractor neural network with increasing activity differs from common Hopfield network by sparse coding and original two-run network dynamics. The properties of this network not investigated yet. It is well known for common Hopfield network that network dynamics can converge to true or spurious attractors. The ability to distinguish between them and the probability of the convergence to true attractors are key features of the
network. In this section, we investigate the influence of the parameters of the neural network on the ability of revealing all true attractors.

In this section, each pattern of the training set is supposed to be generated in the form \( x_m = \bigvee_{i=1}^{L} s_{mi} \land f_i \), where \( x_m \), \( m = 1, \ldots, M \), is a binary row vector of dimension \( N \), \( f_i \in B_n^{N1}, i = 1, \ldots, L, \) is a binary row vector of factor loadings of dimension \( N \) (hereafter, it is also referred to as factor), and \( s_{mi} \in B_C^{L} \) is a binary factor score of \( i \)-th factor in \( m \)-th pattern. It means that in the generative model (3) noise is absent \( (p_{ij} = f_{ij} \text{ and } q_j = 0) \), the number of factors mixed in one pattern is fixed and is equal to parameter \( C \) called as a signal complexity, the number of attributes constituting the factor (i.e., for attributes with \( f_{ij} = 1 \)) is fixed and is equal to \( n = pN \) where parameter \( p \) specifies the level of sparseness of the factors encoding.

The properties of the network are investigated in dependence on the five parameters: \( N, M, L, C, p \). The size of the training set \( M \) should be large enough so that each factor could be presented several times in combinations with different other factors, i.e., \( MC/L \gg 1 \), however we put \( C/L \ll 1 \). Additionally we put \( L \gg 1 \) and \( N \gg 1 \), \( p \ll 1 \).

4.1 Lyapunov function of true attractors

By definition, the Lyapunov function of each attractor can be estimated as a mean synaptic excitation produced in the neurons of attractor by their proper activity. When \( r < p \) true attractors are created by fragments of factors. Thus to evaluate the Lyapunov function for true attractors we have to evaluate a synaptic excitation produced in the neurons of factors by activation of their fragments.

Without any loss of generality, we may assume that a fragment of factor \( f_1 \) is activated. As shown in Chapter 5 of the thesis, its mean Lyapunov function is equal to

\[
\lambda_{\text{true}} = \left[ (M^1)(1 - \bar{a})^2 - (M^0)\bar{a}(1 - \bar{a})/N \right]rN,
\]

where \( rN \) is the number of active neurons, \( \bar{a} = \langle a_m \rangle = 1 - (1 - p)^C \) is a mean bias, \( M^1 \) and \( M^0 \) are the number of patterns of the learning set which include and do not include the recalled factor \( f_1 \). The probability of factor to be mixed in the learning pattern is \( C/L \). Thus \( \langle M^1 \rangle = MC/L \) and \( \langle M^0 \rangle = M(1 - C/L) \approx M \) and the first term in the expression for \( \lambda_{\text{true}} \) is of

\[1B_n^N = \{x|x_i \in \{0, 1\}, \sum_{i=1}^{N} x_i = n\}\]
the order $MC/L$ and the second - $M/N$. Usually $CN/L \gg 1$ and the second term can be neglected. Then in the range $r \leq p$ the mean Lyapunov function for true attractor can be estimated as

$$\lambda_{\text{true}} = MC r N (1 - \bar{a})^2 / L.$$  \hspace{1cm} (23)

### 4.2 Global spurious attractors

In order to clarify the performance of the additional inhibitory neuron, computer simulations were made. Figure 3(a) demonstrates trajectories of the network dynamics obtained without the use of additional inhibitory neuron. Trajectories started from random states with $k_{\text{in}} = 5$ active neurons and continued to $k_{\text{fin}} = 33$. Fifty trajectories are shown in each figure, and only two trajectories with a high Lyapunov function dominate and attract all other trajectories.

As demonstrated in Fig. 3(b) the trajectories with a high Lyapunov function are spurious. The trajectories in Fig. 3(b) are obtained for the case when the number of active neurons is fixed and equal to the number ones in factors. The trajectories are shown in the plane constituted by axes $[m(t), \lambda(t)]$ where $t$ is the time step, $\lambda(t)$ is the Lyapunov function, $m(t)$ is the overlap of the current network state $x(t)$ with a recalled factor $f$ (nearest factor) defined as $m(t) = m(x(t), f) = 1 / N \sum_{i=1}^{N} (x_i(t) - p)(f_i - p)$. According to this
formula, the overlap is equal to 1 if \( x(t) \) is identical to \( f \) and is equal to 0 in average if \( x(t) \) is independent of \( f \). The trajectories are started from random states with \( m(0) = m_{in} = 0.3 \). It is shown that only small portion of trajectories converge to factors, i.e., to the final states with \( m = 1 \). The most trajectories converge to spurious states with high values of the Lyapunov function and \( m \) is close to 0. Attractors corresponding to these spurious states are called as global spurious attractors.

The global spurious attractors are two attractors created by neurons contained in the highest and in the lowest numbers of factors, respectively. To demonstrate this fact we redrew in Fig. 4 the trajectories in the axes \([m(t), \text{rank}(t)]\) where \( \text{rank}(t) \) indicates whether the neurons contained in the most or least numbers of factors contribute to the current network activity. To calculate \( \text{rank}(t) \), all neurons were ordered by the number of factors that contained them, so the neurons contained in the smallest number of factors had the lowest rank, and those contained in the largest number of factors had the highest rank. The rank of the current activity was calculated as the sum of the ranks of active neurons. The obtained rank was normalized so that the patterns created by the neurons contained in the smallest number of factors have the rank close to zero, whereas those created by the neurons contained in the largest number of factors have the rank close to one, and those created by random neurons have the rank close to 0.5. Figure 4(a) demonstrates that the patterns created by the global spurious attractors have ranks close to 0 and 1, while true attractors have ranks around 0.5. When two global spurious attractors completely suppressed (Fig. 4(b)), trajectories that converge to states far from the factors are attracted by local spurious attractors with ranks close to 0.5.

Two global spurious attractors dominate because their Lyapunov function exceeds that of true attractors. As shown in Chapter 5 of the thesis, the Lyapunov function for the global spurious attractors can be estimated as

\[
\lambda_{\text{spur}}^{gl} \simeq 2Mr\ln(1/r)N(1 - \bar{a})^2C^2p/L. \tag{24}
\]

According to (23) and (24) the Lyapunov function of true attractors increases proportionally to \( C \) while of spurious attractors proportionally to \( C^2 \). Their ratio amounts to

\[
\lambda_{\text{spur}}^{gl}/\lambda_{\text{true}} \simeq 2Cp\ln(1/r). \tag{25}
\]

It is shown by computer simulation that (25) gives a rather accurate estimation of this ratio. Thus the global spurious attractors become to dominate if \( 2Cp\ln(1/p) > 1 \). For example, for \( p = 0.02 \) the critical complexity of their
Figure 4: Trajectories of neurodynamics when special inhibitory neuron is excluded from (a) and included in (b) the network in the plane $(m(t), \text{rank}(t))$.

dominance amounts to $C \approx 10$. According to (25), the critical complexity increases when sparseness increases (i.e. $p$ decreases). Thus the global spurious attractors were not observed previously for ordinary Hopfield network used because for this network $C = 1$, and $\lambda_{spur}/\lambda_{true} \ll 1$.

### 4.3 The size of factors attraction basins

The network dynamics converge to one of the factors only when the initial state falls inside its attraction basin. In other cases it converges to one of the spurious attractors. Since generally there is no a priori information on factors the initial network state can only fall into a factor attraction basin by chance. Therefore the ability of a Hopfield-like network to perform factor searches is determined by the probability that network activity converges to one of the factors starting from a random state. And the estimation of parameters significant for the size of attraction basins is required.

In this section, the size of attraction basins around factors is estimated by Single-step (SS) approximation and using computer simulation in multi-step recall. In both cases, level of activity of the neural network during network dynamics is fixed, i.e., $k = n = \sum_{j=1}^{N} f_{ij}$.

Single-step (SS) approximation has been proposed by [13] for the densely encoded Hopfield network. It has been shown by Monte-Carlo simulations [14, 15] that SS approximation is very inaccurate for dense coding. However,
it becomes quite accurate when sparseness increases [10]. The principal peculiarity of this approach is that at each time step one ignores the statistical dependence between the network activity and the connection matrix and takes into account only two macroparameters of neurodynamics: the overlap $m(t)$ between the current and recalled patterns and the total network activity. Since the analysis is restricted to the case of fixed level of the relative network activity $r = p$, then the recall process is described by the evolution of only one parameter $m(t)$. Omission of the statistical dependence between the network activity and the connection matrix is possible only for the first step, when the initial activity is actually stated independently of the connection matrix. This is why this approximation is called the “single-step” or “first-step” approximation.

As shown in Chapter 5 of the thesis, when special inhibitory neuron is included in the network, in the limit case $N \to \infty$ and $p \to 0$, the dynamics of the proposed network is determined by the same equations as for the ordinary Hopfield network [10]. The curves which characterize the behavior of the network activity are presented in Fig. 5. The curves are drawn in the plane $(m_{\text{in}}, \alpha)$ where $\alpha = L h(p)/N$ is the relative informational loading and $h(x)$ is the Shannon function. Each curve corresponds to some fixed $p$. Let the initial state of the network activity for a given $\alpha$ be characterized by the point $(m_{\text{in}}, \alpha)$. If this point is under the curve, the overlap between the current pattern and the recalled pattern moves during the recall process to the right, that is to the final overlap $m_{\text{fin}}$ given by the right branch of the curve. The overlap moves to the left for each point above the curve. Maximum at the curve defines the critical information loading $\alpha_{\text{cr}}$, when factors cease to be attractors of the network dynamics. If $\alpha > \alpha_{\text{cr}}$, then network dynamics moves to some spurious state $(m_{\text{fin}} = 0)$, even starting from factor $(m_{\text{in}} = 1)$. If $\alpha > \alpha_{\text{cr}}$ and $m_{\text{in}} = 1$, then network dynamics converges to some attractor in the vicinity of the factor with $m_{\text{fin}}$ defined by right branch of the curve. Thus, the left branch of the curve corresponds to the border of an attraction basin and the right branch defines the distance between factor and attractor in its vicinity. In the SS approximation the border of the attraction basin corresponds to the condition $m(1) = m_{\text{in}}$.

Figure 4 demonstrates that as for the ordinary Hopfield network, the borders of the attraction basins around factors are fuzzy: starting from the states with the same $m_{\text{in}}$, the trajectories may converge to the recalled factor or to some spurious state far from all factors. Consequently, the distribution of final overlaps has two distinct modes: $m_{\text{fin}} \approx 1$ (“true”) and $m_{\text{fin}} \ll 1$ (“spurious”). It is well known for the ordinary Hopfield network that for small informational loading “true” mode prevails, and as informa-
Figure 5: Sizes of attraction basins $m_{in}$ in dependence on relative informational loading $\alpha$. Lines - Single-step approximation: $p = 0.02$ - solid line, $p = 0.004$ - dashed line, $p \to 0$ - dashed-dotted line. Points - results of multi-step recall obtained by computer simulation in the case when special inhibitory neuron is included: $\circ - p = 0.02$, $\bullet - p = 0.004$.

Informational loading increases, the distribution maximum shifts to "false" mode, demonstrating a sharp transition from a retrieval to a not-retrieval network dynamics at a certain $\alpha = \alpha_{cr}$. The transition becomes more sharp when network size increases. Let us consider probability $P$ that starting from some initial state with a given initial overlap $m_{in}$ with a factor, the network activity converges to it. For the ordinary Hopfield network (i.e., when $C = 1$) $P$ depends on $m_{in}$, $\alpha$ and $N$. For the network performing Boolean factor analysis (i.e., when $C > 1$) $P$ additionally depends on $C$ and on the size of the learning set $M$.

For each set of parameters the probability $P$ was estimated as a portion of true trajectories at the histogram of $m_{fin}$ distribution. The computed values of $P$ were transformed by the following logistic mapping to variable $F$:

$$ P = \frac{1}{1 + e^{-F}}. $$

(26)

The results of computer simulations performed for different parameters $N$, $M$, $C$ and $L$ were combined in one family of data for approximation by the regression model

$$ F = a_0 + a_1 \alpha + a_2 N + a_3 \ln N + a_4 \alpha N. $$

(27)
The fitted curves that approximate the data for fixed $\alpha$ are shown in Fig. 6 by thin lines. According to this approximation, the lines constitute two groups. The upper lines are concave and tend to $+\infty$ when $N \to \infty$. The lower line is convex and tends to $-\infty$ when $N \to \infty$. Transition from one to another group occurs due to the change of $\alpha$. The value of $\alpha$ which corresponds to the thick dashed line separating these groups is chosen as critical $\alpha_{cr}$. For each $\alpha < \alpha_{cr}$ the probability of trajectories to converge to factors tends to 1, when $N$ increases. And conversely for $\alpha > \alpha_{cr}$ it tends to zero. Thus $\alpha_{cr}$ corresponds to sharp transition from retrieval to nonretrieval conditions for $N \to \infty$. As mentioned above for Single-step approximation $\alpha_{cr}$ is found from the condition $m(1) = m_{in}$.

From the regression model, $\alpha_{cr}$ can be found as $\alpha_{cr} = -a_2/a_4$. For data combined in a joint family $\alpha_{cr} = 0.307 \pm 0.006$. This value is shown in Fig. 6 by the thick dashed line. For $p = 0.02$ it corresponds to $L = 2.17N$. The regression model, applied to each of datasets separately, gives $\alpha_{cr} = 0.303 \pm 0.005$ for $C = 20$, $\alpha_{cr} = 0.307 \pm 0.008$ for $C = 1$. These values differ insignificantly. However, they significantly exceed the value $\alpha_{cr} = 0.22$, predicted by SS (see Fig. 5). Thus for sparse encoding ($p = 0.02$) the SS approximation underestimates the size of attraction basins. This is confirmed by the computer simulation performed for $p = 0.02$ and $m_{in} = 0.1$ and for $p = 0.004$, $m_{in} = 0.1$.
and 0.3 for the ordinary Hopfield network. The obtained estimations are also shown in Fig. 5. When $\alpha$ is smaller than $\alpha_{cr}$ predicted by SS, then $m(t)$ increases monotonically according to SS assumption. But when $\alpha$ is larger than $\alpha_{cr}$ predicted by SS but smaller than $\alpha_{cr}$ obtained experimentally, then $m(t)$ changes non-monotonically: trajectories move away from the recalled factors at the first step but then return and terminate in their vicinities.

4.4 Probability of true trials during random searches

In this section, the size of attraction basins in the case of variable level of the neural network activity is estimated. The dominance of spurious attractors that prevents activation of true attractors could disturb the ability of the method to reveal factors. There are two reasons why spurious attractors come to dominate when $L$ increases: firstly, due to the increase of their Lyapunov function, and secondly, due to the increase of their number. Thus one can expect the existence of two limits that restrict the network’s ability to search for factors.

To find the first limit the values of the Lyapunov function for spurious attractors $\lambda_{sp}$ depending on $L/N$ when $N \to \infty$ were estimated using computer simulation. The estimates obtained for $C = 1$, $C = 10$ and $C = 20$ are presented in Fig. 7. The Lyapunov function of spurious attractors reaches the Lyapunov function of true attractors at $L \simeq 2.8N$. Thus, the critical loading, $L_1$, when spurious attractors become dominant due to their large Lyapunov function, is approximately $L_1 = 2.8N$. This value is slightly higher than the critical number of factors $\alpha_{cr}N/h(p) = 2.17N$ obtained in Section 4.3.

To estimate the second limit the probability of transitions from spurious to true trajectories along the recall process were analyzed. Initially, when $r = r_{in}$ most trajectories start as spurious but many of them transform into true ones. Transitions from spurious to true trajectories may occur at any point during the recall process and the probability $P_{spur}$ that the trajectory is spurious monotonically decreases when $r$ increases. As an example, probability $P_{spur}$ dependent on $r$ and $N$ is shown in Fig. 8 for $L = 0.7N$. It quickly drops when network size $N$ is relatively small and remains high for large $N$. Thus, when network size is relatively small most trajectories become true during the recall process and can be used for factor recognition. However, when network size increases considerably almost all trajectories are spurious and the recall procedure becomes incapable of factor searches.

As shown in Chapter 5 of the thesis, $P_{spur}$ can be estimated as

$$P_{spur}(r) = \exp\left[-\frac{N}{b} \exp(-aN)(\exp(-br_{in}) - \exp(-br))\right]$$

(28)
Figure 7: Lyapunov functions of spurious attractors for \( r = p \) normalized by mean values of this function over true attractors at the point \( r = p \). ▽ - an ordinary Hopfield-like network (\( C = 1 \)), ★ - \( C = 10 \), ○ - \( C = 20 \). Thin solid line - B-Spline approximation of experimental points. The horizontal line gives the mean Lyapunov function value for true attractors.

Figure 8: Probability \( P_{\text{spur}} \) dependent on \( r \) and \( N \) for \( C = 1, L = 0.7N \). \( P_{\text{spur}}(r) \) is the probability that a trajectory remains spurious until given \( r \). Each experimental point was obtained from over 10,000 trials of computer simulation. Solid lines are approximations of experimental data by the formula (28).
where coefficient $a$ happened to be proportional to $L/N$ and was presented as $a = c_1 L/N$, while coefficient $b$ could be presented as $b = c_2 + c_3 L/N$. Coefficients $c_i$ were found as the best fit over the whole set of tested network parameters: for $C = 1$ they were estimated as $c_1 = (8.8 \pm 0.1) \cdot 10^{-4}$, $c_2 = 25.5 \pm 3$ and $c_3 = 80 \pm 2$ while for $C = 20$ they are $c_1 = (9.6 \pm 0.6) \cdot 10^{-4}$, $c_2 = 690 \pm 30$ and $c_3 = 254 \pm 18$ and for $C = 10$ they are $c_1 = (9.0 \pm 0.4) \cdot 10^{-4}$, $c_2 = 680 \pm 20$ and $c_3 = 367 \pm 14$. According to (28) the probability that a trial finally happened to be true, i.e. $1 - P_{spur}(r_{fin})$, mainly depends on the term $a N - \ln(N/b) = c_1 L - \ln(N/(c_2 + c_3 L/N))$. The probability is relatively high when this term is small. Thus the probability of true trials is relatively high when $L < \ln(N/(c_2 + c_3 L/N))/c_1$ and it drops to zero when this condition is no longer fulfilled. For large $N$ this condition is satisfied for small loading $L/N$. In this case one can ignore the term $c_3 L/N$ comparing with $c_2$ and rewrite this condition as $L < L_2 \approx \ln(N/c_2)/c_1$. Thus for large $N$ the probability of true trials is relatively high when $L < L_2 \approx \ln(N/c_2)/c_1$. For $C = 1$ $L_2 \approx 10^3 \ln(0.04 N)$, for $C = 10$ and $C = 20$ $L_2 \approx 10^3 \ln(0.0014 N)$. The random search of factors is possible when $L$ satisfies both conditions $L < L_1$ and $L < L_2$.

5 Application of hybrid ANNIA and likelihood maximization method

In this section we present several examples of application of LANNIA to real world datasets that are supposed to comply with BFA generative model. For all presented examples this assumption seems to be justified because revealed factors can be interpreted and information gain for LANNIA results is rather high. In order to compare LANNIA with state-of-art methods we also applied several BFA related methods to the same datasets.

5.1 Comparison of LANNIA with other methods in solving the bars problem

The Bars Problem (BP) introduced by Foldiak [2] and described in Section 2.4 is a common benchmark to reveal strengths and weaknesses of BFA methods. In this section, the efficiency of LANNIA is compared with four other BFA related methods in solving BP: Dendritic Inhibition (DI) neural network [16], Maximal Causes Analysis restricted to the case when each pattern of the dataset contains not more than three factors (MCA$_3$) [17], Expectation-Maximization Binary Factor Analysis (EMBFA) [12], fast
Figure 9: Information gain $G$ for the six BFA methods in dependence on number of observations $M$ in dataset. Noise is absent ($q_j = 0$, $p_{ij} = f_{ij}$), (a) – standard BP problem, (b) – factors are double bars. ○ – LANNIA, • – BANNIA, ☆ – BMFCA, △ – EMBFA, □ – DI, ◇ – MCA$_3$. Thick line – right solution.

Boolean Matrix Factorization based on Formal Concept Analysis (BMFCA) [4]. The last compared method is the special version of LANNIA in which LM procedure is replaced by simple bayesian classifier, described in Section 4.4 of the thesis. The results of BFA methods are also compared with the right bars problem solution. The comparison is performed in terms of information gain.

In addition to testing the methods on the bars problem in its original formulation in Section 2.4, the following versions of the problem were used for comparison of the methods: strongly overlapping bars, in the presence of noise that was assumed to be distributed uniformly over signal components and factors, and with the increased mean number of bars mixed in images. The sensitivity of the methods to strong factor overlapping is tested on the version of bars problem when each of 16 factors is a double bar overlapping with two other factors by half of its pixels. In the noisy bars problem, the noise was assumed to be distributed uniformly over observation components and factors so that $q_j = q$ for any $j$, and $p_{ij} = p f_{ij}$ for any $i$ and $j$. This means that pixels constituting a bar can take 0 with the equal probabilities $1 - p$ and any pixel can take 1 with the probability $q$ due to related specific factor. To investigate the sensitivity of the BFA methods to the mean number of bars mixed in observations (encoded by parameter $C$), the size of the images should be increased in order to provide sparse coding of signals.
Figure 10: Information gain $G$ for the six BFA methods in dependence on $q$ for $p = 1$ (a) and on $p$ for $q = 0$ and $q = 0.2$. ◦ – LANNIA, • – BANNIA, ☆ – BMFCA, △ – EMBFA, □ – DI, ♦ – MCA$_3$. Thick line – right solution.

The increased image size to a grid of $16 \times 16$ pixels allowed us to study the effects of increasing $C$ up to $C = 10$. Only noiseless case is considered here (i.e., $p_{ij} = f_{ij}$, $q_{j} = 0$).

The dependence of the information gain $G$ on the number of observations $M$ in a dataset for the standard and double bars problem is shown in Figures 9(a) and 9(b). Figures 10 and 11 demonstrate the sensitivity of BFA methods to noise in solving the standard BP and to the mean number of bars mixed in observations, correspondingly.

As it follows from the experiments, only LANNIA provides almost the right bars problem solution for all analyzed bars problem tasks. It is least sensitive to noise in data, to the number of factors mixed in each observation and to the insufficient number of observations (decrease of $M$) in the dataset. BMFCA is perfect only in the absence of noise in factors, then it is insensitive to decrease of dataset size and to specific noise. However, it loses to other methods in the presence of noise in the form of factor distortion. On the contrary, MCA$_3$ is insensitive to factor distortion but very sensitive to specific noise. MCA$_3$ is also unable to reveal factors when the number of factors mixed in the pattern of the dataset $C > 3$. DI is moderately sensitive to noise of both kinds but very sensitive to signal complexity $C$. When $C$ increases, the method becomes unstable in the sense that its operation strongly depends on the realization of the initial synaptic weights of the basic network. For one set of initial weights, DI may converge to true factors, while for another, it converges to a random solution. The EMBFA...
Figure 11: Information gain $G$ vs $C$ for 16-by-16 pixel images at $M = 800$. ◯ – LANNIA, ● – BANNIA, ☆ – BMFCA, △ – EMBFA, □ – DI, ◇ – MCA$_3$. Thick line – right solution.

method exhibits a low sensitivity to noise of both kinds but is moderately sensitive to signal complexity $C$. BANNIA is only sensitive to a ratio of number of observations $M$ in the dataset to the number of attributes $N$. If $M < \alpha N$, observations of the dataset create individual attractors of ANNIA dynamics which can dominate over attractors created by factors preventing the search of factors. Coefficient $\alpha$ is estimated in Section 4.4. LANNIA is able to overcome this problem by means of highly-accurate unlearning of factors revealed first, but BANNIA does not.

5.2 Application to text datasets

We supposed that each textual document of the dataset is presented as a binary vector of the dimension of the used term dictionary. Each component of the vector is 1 or 0 depending on the presence or absence of the corresponding term in the document. We hypothesize that each topic is characterized by a specific set of terms (keywords) which appear in an article on the topic. We call this set as a concept. The coherent appearance of the terms of the concept in the article constitutes evidence that the article is dedicated to the corresponding topic.

In the frame of Boolean factor analysis, each concept represents a factor, factor loading is 1 or 0 depending on whether the term belongs to the concept, and factor score is also 1 or 0 depending on whether the article belongs to the topic. The identification of factors is equivalent to the automatic extraction of topics keywords. It is supposed that concepts occur in
documents independently of each other. Though each topic is represented by a set of keywords, there are no or few documents containing the whole set. Factor distortion means the absence of some keywords in the document dedicated to the topic. Each specific factor relates to each individual word to be present in the document independent of topics. One can expect that, first, factors are distorted inhomogeneously \( (p_{ij} \neq p_{f_{ij}}) \) and, second, that all words are distributed in the documents with different probabilities \( (q_j \neq q) \).

### 5.2.1 Analysis of the proceedings of the IJCNN and Neuroinformatics conferences

As a source for textual databases we used the papers published in the proceedings of the IJCNN conferences held in 2003 and 2004, and in the proceedings of the Russian conference on Neuroinformatics held in 2004 and 2005. The sizes of the considered databases amounted to \( M = 1042 \) and \( M = 189 \) articles, respectively. After stop-words and rare words filtering the sizes of the dictionaries were \( N = 3044 \) and \( N = 1716 \) words. Databases of international and Russian conferences were analyzed separately. We are interested in comparing the sets of topics of different conferences and the contents of similar concepts. In the present case, this task seems to be rather difficult a priori because all the topics of the conferences belong to one narrow domain “Neural Networks” and it is well known that keyword extraction for topics of a narrow domain is one of most difficult tasks in text analysis.

Ten top significant terms of factors revealed in the IJCNN database are presented in Table 1. By the specificity of words factor with we related these factors to the topics: 1) Neurobiology, 2) Classification, 3) Optimization, 4) Probability, 5) Hardware, 6) Genetic Algorithms, 7) Image processing, 8) Multilayer networks, 9) Dynamic stability, 10) Self-organizing mapping, 11) Source separation. The twelfth factor looks strange. It contains abbreviations “Fig.N” printed without space and “IEEE Trans”. We revealed that this factor was created due to the fact that articles from 2003 contained the misprint “Fig.N” without space two times more frequently than articles from 2004. The term “IEEE Trans” was bound with terms “Fig.N” due to the fact that the PDF-format for articles from 2003, in contrast to 2004, included the printing of “IEEE Trans” at the end of each page. The appearance of this factor stressed the fact that the method is based on pure statistics, however, the statistics correspond to the nature of the textual database: articles on the same topic tend to contain the same set of words. That is why all other topics are quite reasonable.
Table 1: Ten top significant terms for factors found in the IJCNN database. |f| – number of terms in factor, |s| – number of related articles.

On average one article contains 2.2 factors. All possible combinations of factors were not uniformly distributed over the set of articles. Most of their combinations did not appear at all. There were only 150 combinations of factors from the total set of 2048 possible ones.

The properties of factors revealed in the Neuroinformatics database are shown in Table 2. The words corresponding to the factors are translated in English. In the Neuroinformatics database, we also revealed twelve factors. We related these factors to the topics: 1) Neurobiology, 2) Multilayer networks, 3) Image processing, 4) Classification, 5) Optimization, 6) Intellectual systems, 7) Genetic Algorithms, 8) Recurrent networks, 9) Mathematics, 10) Intellectual agents, 11) Time series, 12) Clustering. On average one article contained 1.9 factors.

Seven topics in the Neuroinformatics coincide with those in the IJCNN database (namely Multilayer networks, Image processing, Classification, Optimization, Genetic Algorithms, Intellectual agents, Clustering).
<table>
<thead>
<tr>
<th>#</th>
<th>f</th>
<th>s</th>
<th>Top terms</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>16</td>
<td>41</td>
<td>physiology, nervous, excitatory, synaptic, inhibitory, activation, membrane, stimulus, brain, cortex</td>
</tr>
<tr>
<td>2</td>
<td>22</td>
<td>39</td>
<td>optimization, hidden, iteration, backpropagation, layer, minimization, neural network, perceptron, sampling, weight</td>
</tr>
<tr>
<td>3</td>
<td>18</td>
<td>29</td>
<td>brightness, orientation, undertone, two-dimensional, image, radial, vision, uniform, pixel, area</td>
</tr>
<tr>
<td>4</td>
<td>14</td>
<td>32</td>
<td>recognition, multilayer, classification, class, sampling, perceptron, practical, recommendation, stage, member</td>
</tr>
<tr>
<td>5</td>
<td>13</td>
<td>24</td>
<td>iteration, convergence, gradient, perceptron, multilayer, stop, optimum, optimization, testing, minimization</td>
</tr>
<tr>
<td>6</td>
<td>13</td>
<td>26</td>
<td>organisation, apparatus, objective laws, hierarchy, mechanism, intellectual, development, language, conception, understanding</td>
</tr>
<tr>
<td>7</td>
<td>12</td>
<td>19</td>
<td>selection, independent, stop, genetic, population, sampling, mutation, optimization, criterium, efficiency</td>
</tr>
<tr>
<td>8</td>
<td>14</td>
<td>46</td>
<td>vector, zero, number, equal, associative, iteration, cycle, perceptron, rule, change</td>
</tr>
<tr>
<td>9</td>
<td>15</td>
<td>30</td>
<td>geometry, discret, measurement, plane, differentiation, form, physical, mathematical, boundary, integral</td>
</tr>
<tr>
<td>10</td>
<td>13</td>
<td>24</td>
<td>intelligence, need, search, selection, presence, operation, mapping, quality, probability, adaptation</td>
</tr>
<tr>
<td>11</td>
<td>11</td>
<td>38</td>
<td>rule, statistics, finance, vector, knowledge, prognosis, random, prediction, probability, expectation</td>
</tr>
<tr>
<td>12</td>
<td>11</td>
<td>15</td>
<td>clustering, Kohonen, separation, distribution, center, noise, statistics, partition, distance, selection</td>
</tr>
</tbody>
</table>

Table 2: The same as in Table 1 but for case of Neuroinformatics database.

Topic “Mathematics” is more general than the topic “Probability” in the IJCNN. The topics “Hardware” and “Dynamic stability” are absent from Neuroinformatics, while topics “Intellectual systems”, “Recurrent networks” and “Time series” are absent from IJCNN. We cannot be sure that the articles on these topics were completely absent from IJCNN. We only know that their presence was too weak to create the factors.

Information gain obtained for Neuroinformatics by the method amounted to 0.15, that means that presentation of these textual data in the BFA model is quite reasonable. It is interesting that the grouping of the articles across scientific sections produced by Program Committees provided informational gain of only 0.04.
5.2.2 Analysis of the Reuters R52 dataset of news messages

Dataset R52 (from Reuters 21578) contains 9100 documents, which are labeled as belonging to 52 topics (classes). Before factor extraction stop words, rare words which appear less than in 10% of documents were removed. As a result, the size of the dictionary amounted to \( N = 3340 \) words.

LANNIA revealed 39 factors within its 4 full cycles alternating ANNIA and LM, providing information gain \( G = 0.12 \). Three factors were found in the first cycle. Two of them had high intersections with two large classes of R52 labeled "earn" and "acq" containing 43% and 25% of the whole number of documents. The first factor was completely embedded in the first of them, with precision and recall amounted to \( p = 0.94 \) and \( r = 0.43 \). The second factor intersected with the second class with \( p = 0.76 \) and \( r = 0.62 \). During the next cycles of LANNIA these factors were replaced by smaller factors which provide the hidden structure of the dataset in more details. For example, the combination of particular six of 39 factors obtained at the last cycle provides intersection with the first class with \( p = 0.91 \) and \( r = 0.92 \) and the combination of another particular five factors provides intersection with the second class with \( p = 0.78 \) and \( r = 0.83 \).

The set of factors obtained at the first and the last LANNIA cycles intersects with all classes of R52 with micro-averaged \( F_1 \) score \( F_{1\text{micro}} = 0.55 \). If to replace the 6 and 5 factors mentioned above by their combinations, then \( F_{1\text{micro}} \) score increases to \( F_{1\text{micro}} = 0.72 \). The mean number of factors mixed in one text amounts to 2. On average, one factor contains 5.8 words with \( p_{ij} > 0.3 \), 1.7 words with \( p_{ij} > 0.5 \), and only 0.5 words with \( p_{ij} > 0.7 \). On average \( q_j \) amounted to \( 4.5 \cdot 10^{-3} \). Thus the data are actually rather noisy.

It is interesting that division of R52 into classes made by experts provides smallest information gain \( G = 0.09 \). Note that LANNIA provides multi-assignment clustering of the dataset (for example, according to LANNIA each document is related to two topics on average) while experts prescribed each document to a single class. One may expect that multi-assignment clustering better corresponds to the data hidden structure than division of the dataset by classes without intersection.

5.3 Application to the Genome dataset analysis

One of the important problems in modern biology is to identify functions of proteins in organisms. A fast growing number of organisms with fully sequenced genomes makes it possible to reveal the protein function by comparing protein phylogenetic profiles of different organisms. The protein
phylogenetic profile is defined as a binary pattern that encodes by 1 and 0 the presence or the absence of proteins in a given organism with the fully sequenced genome. Revealing sets of proteins which coherently appear in different organisms may facilitate the search for functional modules in the genome structure. Since the concept of the genome functional modularity is completely compatible with the BFA generative model it was a challenge to apply LANNIA to reveal the hidden factor structure in the largest genome database KEGG [18] that contained the fully sequenced genomes of $M = 1368$ organisms. The protein phylogenetic profile of each organism is a binary pattern $x_m$ of dimension $N = 11451$, where $N$ is a whole number of proteins taken into account.

LANNIA revealed 38 factors after four full cycles of the combination of ANNIA and LM. On average one factor contains 235 proteins with $p_{ij} > 0.9$, 407 proteins with $p_{ij} > 0.7$ and 598 proteins with $p_{ij} > 0.5$. The number of proteins in the factors greatly exceeds the number of proteins in the metabolic pathways described in KEGG. The information gain provided by LANNIA after the last cycle amounts to 0.32. The relatively high gain obtained shows that, first, the genome data indeed correspond to the generative BFA model and, second, LANNIA is the efficient method for finding its parameters. The high information gain is in favor of the hypothesis of the modular genome structure [19].

Figure 12 demonstrates the distribution of the factors over the organisms. All the organisms are grouped in types according to the taxonomy of KEGG from animals to bacteria. The type of the organisms is depicted by the number on the top of Fig. 12. The factors are ranged in descending order according to the frequencies of their appearance in the dataset. The factor number one appeared in the most organisms (in 22% of the organisms) and the factor number 38 appeared in the least of them (in 5% of the organisms). For the most factors the frequencies of their appearance in the organisms are distributed around 0.1. In Fig. 12 the appearance of a given factor in a given organism is marked by the point. Thus, the frequency of appearance of each factor in the dataset corresponds to the number of points in each horizontal line. Figure 12 demonstrates that each type of the organisms is characterized by the specific set of factors. For example, animals are characterized by factors 20 and 37, fungi by factor 20, plants by factors 2, 20 and 37 and so on. Factor 20 was identified only in eukaryotes and never in prokaryotes. Conversely, factor 1 was identified in all types of prokaryotes but never in eukaria. Thus, the distribution of factors over the types of organisms seems to reflect some peculiarities of their functioning. It is interesting that LANNIA revealed only little effect of specific factors: only 472 proteins
over 11451 taken into account have $q_j$ exceeding 0.01. Thus, almost all the organisms are completely described by common factors as predicted by the modular hypothesis [19].

In order to compare LANNIA with other methods, we also applied BANNIA and BMFCA to KEGG. BANNIA provided the maximal information gain $G = 0.35$ that was reached for 27 factors. The obtained gain is much lower than the gain provided by LANNIA. The relatively small informational gain provided by BANNIA is explained by the fact that it revealed only 7 independent factors and all other found factors happened to be similar to them. The maximal information gain $G = 0.49$ obtained by BMFCA was reached for 84 found factors. The BMFCA efficiency in KEGG analysis is comparable with the results obtained by LANNIA. Since, as shown before, BMFCA is very sensitive to factor distortion, one can expect that factors in KEGG dataset are mixed in observations with relatively small distortion. Indeed, according to LANNIA, each factor contains on average 140 proteins with $p_{ij} = 1$ and 235 proteins with $p_{ij} = 0.9$. Factors found by all methods happened to be similar. This is in favor of the conclusion that the Genome dataset actually has the latent factor structure which can be revealed by the considered methods, however LANNIA has an advantage because it provides the highest gain with the smallest number of factors.
Factor analysis is one of the most powerful statistical methods to reveal and reduce information redundancy in high dimensional data. Revealing of hidden relationships among data allows to find an appropriate data representation in a space of fewer dimensions. Boolean factor analysis (BFA) is used to find hidden relationships among binary data. Binary presentation of data is quite typical of many fields including sociology, marketing, zoology, genetics, and medicine. However, BFA is only weakly developed.

The primary achieved goal of this contribution was the development of an efficient method for BFA that is suitable for high dimensional, large datasets. The method is suggested in Chapter 4 of the thesis. It is based on an original Hopfield-like attractor neural network with increasing activity (ANNIA) coupled with the expectation-maximization algorithm. ANNIA is very similar to the traditional sparsely encoded Hopfield network \[ \text{10} \] but has an original two-run recall procedure and several special techniques for factors identification. The expectation-maximization algorithm is used to estimate of the BFA generative model parameters by maximizing the data likelihood. The resulting method is called LANNIA.

The secondary achieved goal was to study the properties of ANNIA. The influence of the parameters of the neural network on the ability of revealing factors is studied in Chapter 5 of the thesis both theoretically and by computer simulations. The existence of two global spurious attractors was revealed. It was shown that these attractors become dominant only when signal complexity is high, and their dominance can be completely suppressed when one special inhibitory neuron is added to the network. The size of attraction basins around factors was estimated by computer simulations and by the single-step approximation. It turns out that the critical relative informational loading \( \alpha_{cr} \approx 0.3 \). The probability to find a factor when the network dynamics starts from a random initial state was estimated. This probability happened to be high only when the number of factors \( L \) is less than two limits \( L_1 \) and \( L_2 \). The first limit \( L_1 \) corresponds to the number of factors \( L \) which provides equal Lyapunov function values for true and spurious attractors. The second limit \( L_2 \) relates to the exponentially increasing number of spurious attractors when the network size increases. For the level of sparseness of the factors encoding \( p = 0.02 \), these limits are estimated as \( L_1 = 2.8N \) and \( L_2 \approx 10^3 \ln(0.0014N) \).

The tertiary achieved goal was to evaluate the practical value of the proposed method. For this purpose, the method was applied to various natural datasets. The description of the obtained results, as well as a comparison
with the results obtained by other methods is given in Chapter 6 of the thesis. First, the performance of LANNIA was compared with other BFA related method in solving the bars problem. It was shown that only the proposed method provides almost the right bars problem solution for all analyzed bars problem tasks. It is least sensitive to noise in data, to the number of factors mixed in each observation and to the insufficient number of observations in the dataset. Other methods exhibit good performance only in part of the tasks. The execution time of LANNIA in solving the bars problem was comparable with the fastest BFA related method.

Second, the performance of the method was tested on real-world binary datasets. The method was applied to two different text datasets on the same topic (Neural Networks) in two different languages (English and Russian), and was tested on Reuters R52 dataset of news messages. We observed high values of information gain, stability of the results when changing the dataset, strong intersect with manual classification. Then the method was applied to the problem of revealing the modular structure in genome datasets. It is assumed that solving this problem would allow to identify the functions of proteins in organisms. The information gain of solution found by LANNIA turns out to be very high ($G = 0.32$). The high gain shows that, first, the genome data indeed correspond to the generative BFA model and, second, LANNIA is the efficient method for finding its parameters. It is demonstrated that each type of the organisms is characterized by the specific set of factors. The analysis of information loadings of the revealed factors led to the conclusion that the influence of specific factors (noise) is minimal, and almost all the organisms are completely described by common factors as predicted by the modular hypothesis. The method was also tested on the database of roll-call votes in the Russian parliament and on the mushrooms dataset. In both cases, a clear interpretation of the factors revealed by the method was found.

We conclude with the note that LANNIA could be effectively used to solve a wide range of real-world problems related to the study of binary data structure, but two issues hinder this. First, the method is not implemented as a program or library for a popular statistical or engineering package, and hence the method is not easily accessible to non-computer science researchers. Second, the BFA generative model has some assumptions that may not correspond to generative model of data. The most limiting assumption is the statistical independence of appearance of factors in observations stated by (7). For future work, it would be worthwhile to extend the model to the case of different types of relationships between factors: hierarchy of factors, causal relationships, and others.
References


**Author’s selected publications related to the topic of the thesis**


