East Asian Journal on Applied Mathematics
doi: 10.4208/eajam.080310.200910a

Vol. **1**, No. 2, pp. 132-154
May 2011

Construction of Probabilistic Boolean Networks from a Prescribed Transition Probability Matrix: A Maximum Entropy Rate Approach

Xi Chen[1], Wai-Ki Ching[*,1], Xiao-Shan Chen[2], Yang Cong[1] and Nam-Kiu Tsing[1]

[1] *Advanced Modeling and Applied Computing Laboratory, Department of Mathematics, The University of Hong Kong, Pokfulam Road, Hong Kong.*
[2] *School of Mathematical Sciences, South China Normal University, Guangzhou, China.*

Received 8 March 2010; Accepted (in revised version) 20 September 2010
Available online 7 April 2011

Abstract. Modeling genetic regulatory networks is an important problem in genomic research. Boolean Networks (BNs) and their extensions Probabilistic Boolean Networks (PBNs) have been proposed for modeling genetic regulatory interactions. In a PBN, its steady-state distribution gives very important information about the long-run behavior of the whole network. However, one is also interested in system synthesis which requires the construction of networks. The inverse problem is ill-posed and challenging, as there may be many networks or no network having the given properties, and the size of the problem is huge. The construction of PBNs from a given transition-probability matrix and a given set of BNs is an inverse problem of huge size. We propose a maximum entropy approach for the above problem. Newton's method in conjunction with the Conjugate Gradient (CG) method is then applied to solving the inverse problem. We investigate the convergence rate of the proposed method. Numerical examples are also given to demonstrate the effectiveness of our proposed method.

Key words: Boolean networks, conjugate gradient method, genetic regulatory networks, inverse problem, Markov chains, Newton's method, probabilistic Boolean networks, transition-probability matrix.

1. Introduction

Building mathematical models and developing efficient numerical algorithms for studying regulatory interactions among DNA, RNA, proteins, and small molecules are important research issues in computational systems biology [7, 26]. In fact, many formalisms and

[*]Corresponding author. *Email addresses:* dlkcissy@hotmail.com (X. Chen), wching@hkusua.hku.hk (W.-K. Ching), cxs333@21cn.com (X.-S. Chen), congyang0305@yahoo.com.cn (Y. Cong), nktsing@hku.hk (N.-K. Tsing)

mathematical models have been proposed in the literature to study genetic regulatory networks such as Bayesian networks [25], Boolean Networks (BNs) [21, 22], multivariate Markov chain models [9], regression models [45], Probabilistic Boolean Networks (PBNs) [32–35], and a review on other mathematical models can also be found in [16,36]. Among these models, BNs and their extensions PBNs have received much attention as they are able to capture the switching behavior of biological processes [26].

Boolean logic owes its name to George Boole who devised a mathematical framework for logical reasoning [4, 5]. BN models were first introduced by Kauffman [21–24]. Reviews of BN models can be found in [26, 37]. In a BN, the gene expression states are quantized to only two levels: on and off (represented as 1 and 0). The target gene is determined by several genes called its input genes via a Boolean function. When the input genes and the Boolean functions are given, then we say that a BN is defined. We remark that a BN is a deterministic model and the only randomness comes from its initial state. Given an initial state, the BN will eventually enter into a cycle of states called its attractor cycle. Since genetic regulation processes exhibit uncertainty and microarray data sets used to infer the model have errors due to experimental noise in the complex measurement processes, it is more realistic to consider stochastic models. The idea of extending the concept of a BN (a deterministic model) to a PBN (a probabilistic model) is as follows. For each gene, there can be more than one Boolean function and corresponding selection probabilities are assigned to the Boolean functions. The dynamics (transitions) of a PBN can be studied using Markov chain theory [10, 32, 35].

Given a PBN, the network behavior is characterized by its steady-state probability distribution which gives the first-order statistical information of a PBN. One can understand a genetic regulatory network and identify the influence of different genes via such a network. In [44], an efficient method has been used to construct the transition probability matrix and the standard iterative power method for computing the resulting steady-state probability distribution. Later, also a matrix approximation method has been proposed in [11] to get an approximation of the steady-state probability distribution efficiently. Furthermore, it is possible to control some genes in a network so as to drive the whole network into a desirable state or a steady-state probability distribution (a mixture of states). Therapeutic gene intervention or gene control policy [12, 15, 33, 35] can therefore be developed and studied.

Here we study the problem of constructing a PBN based on a given transition-probability matrix and a set of BNs. This is an inverse problem of huge size. The inverse problem is ill-posed, meaning that there can be many networks or no network having the desirable properties. Pal et al. [29] have presented two algorithms to solve the inverse problem of finding attractors constituting a BN. Network inference from steady-state data is a very important problem as most microarray data sets are assumed to be obtained from sampling the steady-state. In fact, the inverse problem can be split into two different tasks. The first task is to construct a sparse transition-probability matrix from a given network steady-state probability distribution. A maximum entropy rate approach has been proposed for this purpose [13]. The second task is to construct a PBN (the BNs and the selection probabilities) from a given steady-state probability distribution. Here, we propose to apply Newton's

method together with the conjugate gradient method to solving the inverse problem.

The remainder of the paper is structured as follows. Section 2 gives a brief review on BNs and PBNs. In Section 3, we present the inverse problem. Some interesting properties of the model are also discussed. In Section 4, we consider a maximum entropy approach for the inverse problem. Section 5 presents numerical examples to demonstrate our proposed algorithm. Finally, concluding remarks are given in Section 6.

2. A Review on Boolean Networks and Probabilistic Boolean Networks

The BN model is a particular case of discrete dynamical systems [2]. In this section, we first define a dynamical system (M,D). Then, we will present the definition of the BN model based on (M,D). Let (M,D) be a discrete dynamical system consisting a set M and a map $D: M \to M$, where M is a product of sets $M_1, M_2, \cdots, M_n$. Then, the map D can be specified in terms of its components [27]:

$$D_i : M_1 \times M_2 \times \cdots \times M_n \to M_i\,.$$

Here if we define m_i ($m_i \in M_i$) to be the state (0 or 1) of gene i, then the map D will represent the set of rules of the regulatory interactions among the genes (set of Boolean functions):

$$D(\mathbf{m}) = (D_1(\mathbf{m}), D_2(\mathbf{m}), \cdots, D_n(\mathbf{m}))^T$$

where $\mathbf{m} = (m_1, m_2, \cdots, m_n)^T$ and $D_i : \{0,1\}^n \to \{0,1\}$.

Now, we are ready to give the definition of a BN having n genes. A Boolean Network (BN) $G(V,F)$ consists of a set of vertices:

$$V = \{v_1, v_2, \cdots, v_n\}$$

and a list of Boolean functions:

$$F = \{f_1, f_2, \cdots, f_n\}$$

($f_i : \{0,1\}^n \to \{0,1\}$). We define $v_i(t)$ to be the state (0 or 1) of the vertex v_i at time t. Boolean functions are used to represent the rules of the regulatory interactions among the genes:

$$v_i(t+1) = f_i(\mathbf{v}(t)), \quad i = 1, \cdots, n,$$

where

$$\mathbf{v}(t) = (v_1(t), v_2(t), \cdots, v_n(t))^T$$

is called the Gene Activity Profile (GAP). The GAP can take any possible form (state) from the set:

$$S = \{(v_1, v_2, \ldots, v_n)^T : v_i \in \{0,1\}\} \tag{2.1}$$

and thus totally there are 2^n possible states.

Considering the inherent deterministic directionality in BNs as well as the fact that it can assume only a finite number of states, it is easy to see that some states will be re-visited

Table 1: The Truth Table.

State	$v_1(t)$	$v_2(t)$	$f^{(1)}$	$f^{(2)}$
1	0	0	0	0
2	0	1	0	0
3	1	0	1	1
4	1	1	1	0

infinitely often, depending on the initial starting state. Such states are called attractors and the states leading to comprise their basins of attraction. The number of transitions needed to return to a given state in an attractor is called the cycle length [1, 21, 22]. It is also well-known that eventually a BN will enter into an attractor cycle and stay there forever. The cycles can have biological significance [19] such as states of cell proliferation or cell apoptosis. For more details we refer interested readers to [24].

Here we give an example of a BN having two genes with the truth table given in Table 1. From the truth table, there are four states and they are $(0,0),(0,1),(1,0)$ and $(1,1)$. Let us label them by $1,2,3$ and 4 respectively. We note that if the current state of the network is 1, the network will stay with State 1 in the next step (with probability one). If the current state is 2, the network will go to State 1 in the next step (with probability one). Similarly, if the current state is 3, the network will go to State 4 in the next step (with probability one). Finally, if the current network state is 4, the network will go to State 3 in next step (with probability one). The transition-probability matrix (Boolean network matrix) of the 2-gene BN is then given by

$$B = \begin{pmatrix} 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}. \tag{2.2}$$

The truth table provides the one-step transition probability (0 or 1 in the case of BN) between any two states. Let column vectors **a** and **b** be any two states in the set S. By letting **a** and **b** take all the possible states in S, one can get the transition-probability matrix of the 2-gene BN. Since the network is deterministic, each column in B (the Boolean network matrix) has only one non-zero element and the column sum is one. We observe that there are two cycles and they are given as follows: (i) $(0,0) \longleftrightarrow (0,0)$, and (ii) $(1,0) \longleftrightarrow (1,1)$. Thus State 1 is an attractor cycle of period (length) one and States 3 and 4 form an attractor cycle of period two. We remark that there is a one-to-one relation between a BN and its corresponding BN matrix.

Since BN is a deterministic model, the following are some reasons why this is not favorable. First, the biological system has its stochastic nature. Second, the microarray data sets used to infer the network structure are usually not accurate because of the experimental noise in the complex measurement process. To overcome the deterministic rigidity, extension to a probabilistic model is natural. To extend the concept of a BN to a stochastic model, for each vertex v_i in a PBN, instead of having only one Boolean function (as in the case of

a BN) there are a number of Boolean functions (predictor functions) $f_j^{(i)}(j=1,\cdots,l(i))$ to be chosen for determining the state of gene v_i. The probability of choosing $f_j^{(i)}$ as the predictor function is $c_j^{(i)}$, where

$$0 \le c_j^{(i)} \le 1 \quad \text{and} \quad \sum_{j=1}^{l(i)} c_j^{(i)} = 1 \quad \text{for} \quad i=1,\cdots,n.$$

The probability $c_j^{(i)}$ can be estimated by using the statistical method Coefficient of Determination (COD) [18] with real gene expression data sets.

Let f_j be the jth possible realization,

$$f_j = \left(f_{j_1}^{(1)}, f_{j_2}^{(2)}, \cdots, f_{j_n}^{(n)}\right), \quad 1 \le j_i \le l(i), \quad i=1,\cdots,n,$$

where $l(i) \le 2^{2^n}$ is the total number of possible Boolean functions of gene i. Then in an independent PBN (the selection of the Boolean function for each gene is assumed to be independent), the probability of choosing the corresponding BN is given by

$$q_j = \prod_{i=1}^{n} c_{j_i}^{(i)}, \quad j=1,\cdots,N.$$

There are at most

$$N = \prod_{i=1}^{n} l(i)$$

different possible realizations of BNs. We note that the transitions among the states in the set S is a Markov chain process [10]. Let $\mathbf{a}$ and $\mathbf{b}$ be any two column vectors in the set S. Then the transition probability

$$\begin{aligned} &\text{Prob}\left\{\mathbf{v}(t+1)=\mathbf{a} \mid \mathbf{v}(t)=\mathbf{b}\right\} \\ =\ &\sum_{j=1}^{N} \text{Prob}\left\{\mathbf{v}(t+1)=\mathbf{a} \mid \mathbf{v}(t)=\mathbf{b}, \text{the } j\text{th network is selected}\right\} \cdot q_j. \end{aligned}$$

The transition probability matrix A of a PBN (Markov chain) can then be obtained by computing the above probabilities for all the possible states in the set S in (2.1). In fact, it can be shown that the transition probability matrix A can be written as the sum of the Boolean network matrices A_i ([11]):

$$A = \sum_{i=1}^{N} q_i A_i, \tag{2.3}$$

where q_i is the probability of choosing the BN with the BN matrix A_i. Here we will focus on estimating q_i when A and A_i are given.

3. The Inverse Problem

In this section, we propose the inverse problem of constructing a PBN from a given transition-probability matrix A and a set of Boolean networks $\{A_i\}$. Suppose that the possible BNs constituting the PBN are known and their BN matrices are denoted by $\{A_1, \cdots, A_N\}$. Moreover, the transition-probability matrix A is observed and its relation to A_i is given in (2.3). We are interested in the estimation of the parameters q_i, $i = 1, \cdots, N$ when A is given. Since the problem size is huge and A is usually very sparse, here we assume that each column of A has m non-zero entries. In this case, we have $N = m^{2^n}$ and we can order $A_1, \cdots, A_{m^{2^n}}$ systematically. We note that q_i and A_i are non-negative and there are only $m \cdot 2^n$ non-zero entries in A. Thus we have $m \cdot 2^n$ equations for m^{2^n} unknowns.

To reconstruct the PBN, one possible way to get q_i is to consider the following minimization problem:

$$\min_{\mathbf{q}} \left\| A - \sum_{i=1}^{m^{2^n}} q_i A_i \right\|_F^2 \tag{3.1}$$

subject to

$$0 \le q_i \le 1 \quad \text{and} \quad \sum_{i=1}^{m^{2^n}} q_i = 1.$$

Here $\|\cdot\|_F$ is the Frobenius norm of a matrix. Let us define a mapping F from the set of $l \times l$ square matrices to the set of $l^2 \times 1$ vectors by

$$F\left(\begin{pmatrix} a_{11} & \cdots & a_{1l} \\ \vdots & \vdots & \vdots \\ a_{l1} & \cdots & a_{ll} \end{pmatrix}\right) = (a_{11}, \ldots, a_{l1}, a_{12}, \ldots, a_{l2}, \ldots, \ldots, a_{1l}, \ldots, a_{ll})^T.$$

If we let

$$U = [F(A_1), F(A_2), \cdots, F(A_{m^{2^n}})] \quad \text{and} \quad \mathbf{p} = F(A), \tag{3.2}$$

then (3.1) becomes

$$\min_{\mathbf{q}} \|U\mathbf{q} - \mathbf{p}\|_2^2 \tag{3.3}$$

subject to

$$0 \le q_i \le 1 \quad \text{and} \quad \sum_{i=1}^{m^{2^n}} q_i = 1.$$

We note that in practice the matrix U is very large, so it is not possible to store the whole matrix and therefore we seek iterative methods for solving the above minimization problem. One possibility is the Conjugate Gradient (CG) method [3]. Since

$$||U\mathbf{q} - \mathbf{p}||_2^2 = \mathbf{q}^T U^T U \mathbf{q} - 2\mathbf{q}^T U^T \mathbf{p} + \mathbf{p}^T \mathbf{p},$$

the minimization problem (3.3) without the constraints is then equivalent to

$$\min_{\mathbf{q}}\{\mathbf{q}^T U^T U\mathbf{q} - 2\mathbf{q}^T U^T \mathbf{p}\}.$$

If U is a full column rank matrix then $U^T U$ is a symmetric positive definite matrix. The minimization problem without constraints is equivalent to solving $U^T U\mathbf{q} = U^T \mathbf{p}$ with the CG method. We note that if there is a vector $\mathbf{q}$ satisfying the equation $U\mathbf{q} = \mathbf{p}$ with $\mathbf{1}^T\mathbf{q} = 1$ and $\mathbf{0} \le \mathbf{q} \le \mathbf{1}$, then the CG method can yield the solution. Thus to ensure that $\mathbf{1}^T\mathbf{q} = 1$, we add a row of $(1, 1, \cdots, 1)$ to the bottom of the matrix U and form a new matrix $\bar{U}$. At the same time, we add an entry 1 at the end of the vector $\mathbf{p}$ to get a new vector $\bar{\mathbf{p}}$. We then consider the revised system of linear equations

$$\bar{U}^T \bar{U}\mathbf{q} = \bar{U}^T \bar{\mathbf{p}}\,.$$

However, the CG algorithm has to be modified to ensure the second constraint $\mathbf{0} \le \mathbf{q} \le \mathbf{1}$. This method can give a solution of the inverse problem. However, usually there are too many solutions – so extra criterion has to be introduced in order to narrow down the set of solutions, or to obtain a unique solution.

One possible and reasonable approach is to consider the solution which gives the largest entropy, as $\mathbf{q}$ itself can be considered to be a probability distribution. This means we are to find a distribution vector $\mathbf{q}$ that maximizes

$$-\sum_{i=1}^{m^{2^n}} q_i \log(q_i).$$

Entropy is a measure of the uncertainty associated with a random variable [31]. It measures, in the sense of an expected value, the information contained in a message. Entropy can also be regarded as a measure of the multiplicity associated with the state of the objects. A state which can be accomplished in many ways is more probable than one that can be accomplished in fewer ways. The entropy method was adopted by Wilson for traffic demand estimation in transportation networks [41] – cf. also [8, 39, 42, 43]. Before addressing our application in the next section, we present some interesting properties of the matrices UU^T and $\bar{U}\bar{U}^T$ that will be useful in the convergence analysis later.

3.1. Eigenvalues of the Matrices UU^T and $\bar{U}\bar{U}^T$

We analyze the eigenvalues of the matrices UU^T and $\bar{U}\bar{U}^T$. We note that when the zero rows of U and $\bar{U}$ are removed, it will not affect the nonzero eigenvalues of UU^T and $\bar{U}\bar{U}^T$, so we assume that U and $\bar{U}$ do not contain any zero row. We observe that U has $m \cdot 2^n$ nonzero rows, and it is straightforward to check that

$$UU^T = m^{2^n-1} I_{m\cdot 2^n} + m^{2^n-2} T, \tag{3.4}$$

where

$$T = \begin{pmatrix} 0_m & E_m & \cdots & E_m & E_m \\ E_m & 0_m & \cdots & E_m & E_m \\ \vdots & \vdots & & \vdots & \vdots \\ E_m & E_m & \cdots & 0_m & E_m \\ E_m & E_m & \cdots & E_m & 0_m \end{pmatrix}, \tag{3.5}$$

0_m is an $m \times m$ zero matrix and E_m is an $m \times m$ matrix with all entries being equal to 1. The matrix T can be written as follows:

$$T = E - F, \tag{3.6}$$

where E is an $m \cdot 2^n \times m \cdot 2^n$ matrix with all entries being equal to 1 and

$$F = \operatorname{diag}\left(E_m, E_m, \cdots, E_m\right). \tag{3.7}$$

The eigenvalues of E_m are m and 0, with multiplicity 1 and $m-1$ respectively. Thus the eigenvalues of the matrix F are m and 0, with multiplicity 2^n and $(m-1)\cdot 2^n$ respectively. The eigenvalues of E are $m \cdot 2^n$ with multiplicity 1 and 0 with multiplicity $m \cdot 2^n - 1$. It is easy to see that $m \cdot (2^n - 1)$ is an eigenvalue of T. We note that E and F commute. Therefore the eigenvalues of the matrix T are $m \cdot (2^n - 1)$ with multiplicity 1, as well as $-m$ and 0 with multiplicity $2^n - 1$ and $(m-1)2^n$, respectively. In fact, from (3.4)-(3.7) it is straightforward to obtain the following theorem related to the eigenvalues of the matrix UU^T.

Theorem 3.1. *The eigenvalues of the matrix UU^T are $2^n \cdot m^{2^n-1}$ with multiplicity 1, m^{2^n-1} with multiplicity $(m-1)\cdot 2^n$ and 0 with multiplicity $2^n - 1$.*

For eigenvalues of the matrix $\bar{U}\bar{U}^T$, we have the following result and the proof can be found in the Appendix.

Theorem 3.2. *Let $s = rank(\bar{U})$. Then we have*
(a) $s = rank(U) = (m-1)\cdot 2^n + 1$;
(b) The matrix $\bar{U}\bar{U}^T$ has at most four different eigenvalues. The largest eigenvalue lies in an interval

$$\left[2^n \cdot m^{2^n-1},\ \left(\sqrt{2^n \cdot m^{2^n-1}} + \sqrt{m^{2^n}}\right)^2\right],$$

the second largest one lies in an interval $\left[m^{2^n-1},\ 2^n \cdot m^{2^n-1}\right]$; other eigenvalues are m^{2^n-1} and 0.
(c)

$$2^n \le \frac{\lambda_1(\bar{U}\bar{U}^T)}{\lambda_s(\bar{U}\bar{U}^T)} \le \left(\sqrt{2^n} + \sqrt{m}\right)^2,$$

where $\lambda_i(H)$ is the i-th largest eigenvalue of a positive semi-definite matrix H.

4. The Inverse Problem and Its Dual Problem

In this section, we first propose the inverse problem for the construction of a PBN with the entropy maximization approach. We then apply Newton's method with the CG method to solving the inverse problem.

4.1. The Inverse Problem

For the inverse problem, we have $m \cdot 2^n$ equations for m^{2^n} unknowns. Thus one may have infinitely many solutions. Since $\mathbf{q}$ can be viewed as a probability distribution, one possible way to get a better choice of q_i is to consider maximizing the entropy of $\mathbf{q}$ subject to the given constraints – i.e. consider the following maximization problem:

$$\max_{\mathbf{q}} \sum_{i=1}^{m^{2^n}} (-q_i \log q_i) \tag{4.1}$$

subject to

$$\bar{U}\mathbf{q} = \bar{\mathbf{p}} \quad \text{and} \quad 0 \le q_i \quad i = 1, \cdots, m^{2^n}.$$

We remark that the constraint $q_i \le 1$ can be discarded as we require that

$$\sum_{i=1}^{m^{2^n}} q_i = 1 \quad \text{and} \quad 0 \le q_i \quad i = 1, \cdots, m^{2^n}.$$

As in [8], the dual problem of (4.1) is therefore of the type

$$\min_{\mathbf{y}} \max_{\mathbf{q}} L(\mathbf{q}, \mathbf{y}) \tag{4.2}$$

where $\mathbf{y}$ is the multiplier and $L(\cdot,\cdot)$ is the Lagrangian function

$$L(\mathbf{q}, \mathbf{y}) = \sum_{i=1}^{m^{2^n}} (-q_i \log q_i) + \mathbf{y}^T(\bar{\mathbf{p}} - \bar{U}\mathbf{q}). \tag{4.3}$$

The optimal solution $\mathbf{q}^*(\mathbf{y})$ of the inner maximization problem of (4.2) solves the equations

$$\nabla_{q_i} L(\mathbf{q}, \mathbf{y}) = -\log q_i - 1 - \mathbf{y}^T \bar{U}_{\cdot i} = 0, \quad i = 1, \cdots, m^{2^n}$$

and is thus of the form

$$q_i^*(\mathbf{y}) = e^{-1-\mathbf{y}^T \bar{U}_{\cdot i}}, \quad i = 1, \cdots, m^{2^n} \tag{4.4}$$

where $\bar{U}_{\cdot i}$ is the ith column of the matrix $\bar{U}$. After substituting $\mathbf{q}^*(\mathbf{y})$ back into (4.3) the dual problem (4.2) can be simplified to

$$\min_{\mathbf{y}} \left\{ \sum_{i=1}^{m^{2^n}} e^{-1-\mathbf{y}^T \bar{U}_{\cdot i}} + \mathbf{y}^T \bar{\mathbf{p}} \right\}. \tag{4.5}$$

The solution of the primal problem (4.1) can be obtained from the solution of the dual problem (4.5) through (4.4). Thus we have transformed a constrained maximization problem with m^{2^n} variables into an unconstrained minimization problem of $m \cdot 2^n + 1$ variables.

4.2. Newton's Method for the Dual Problem

A variety of numerical methods are available for solving problem (4.5). In the following, we will explain how Newton's method in conjunction with the CG method can be used, and give a theoretical justification for the efficiency of the CG method in this setting. To this end we denote

$$f(\mathbf{y}) = \sum_{i=1}^{m^{2^n}} e^{-1-\mathbf{y}^T \bar{U}_{\cdot i}} + \mathbf{y}^T \bar{\mathbf{p}}$$

as the function to be minimized. The gradient and the Hessian of f are respectively

$$\nabla f(\mathbf{y}) = -\bar{U}\mathbf{q}^*(\mathbf{y}) + \bar{\mathbf{p}} \tag{4.6}$$

and

$$\nabla^2 f(\mathbf{y}) = \bar{U} \cdot \mathrm{diag}(\mathbf{q}^*(\mathbf{y})) \cdot \bar{U}^T \tag{4.7}$$

where $\mathbf{q}^*(\mathbf{y})$ is as defined in (4.4) and $\mathrm{diag}(\mathbf{q}^*(\mathbf{y}))$ is the diagonal matrix with diagonal entries $(\mathbf{q}^*(\mathbf{y}))$.

Newton's Method (see Nocedal and Wright [28]):
Choose a starting vector $\mathbf{y}_0 \in \mathrm{Im}(\bar{U})$ (Here $\mathrm{Im}(\bar{U})$ denotes the column space of matrix $\bar{U}$);
$k = 1$;
while $||\nabla f(\mathbf{y}_k)||_2 > tolerance$
find $\mathbf{p}_k$ with $\nabla^2 f(\mathbf{y}_{k-1})\mathbf{p}_k = -\nabla f(\mathbf{y}_{k-1})$;
set $\mathbf{y}_k = \mathbf{y}_{k-1} + \mathbf{p}_k$;
$k = k + 1$;
end.

From Eq. (4.7), we observe that f is strictly convex on the subspace $\mathrm{Im}(\bar{U})$. Newton's method will produce a sequence of points $\mathbf{y}_k$ according to the iteration $\mathbf{y}_k = \mathbf{y}_{k-1} + \mathbf{p}_k$, where the Newton step $\mathbf{p}_k$ is the solution of the Hessian matrix system

$$\nabla^2 f(\mathbf{y}_{k-1})\mathbf{p}_k = -\nabla f(\mathbf{y}_{k-1}). \tag{4.8}$$

We note that $\nabla^2 f(\mathbf{y}_{k-1})$ is a one-to-one mapping of the concerned subspace concerned onto itself. Moreover, from (4.6) we have $\nabla f(\mathbf{y}) \in \mathrm{Im}(\bar{U})$ as $\bar{\mathbf{p}} \in \mathrm{Im}(\bar{U})$. Hence, Eq. (4.8) has a unique solution and therefore Newton's method for minimizing f is well defined. If we start with $\mathbf{y}_0 \in \mathrm{Im}(\bar{U})$, the Newton's sequence will remain in the subspace. Moreover, it will converge locally at a quadratic rate. To enforce global convergence one may apply

line search or trust region techniques [14, 17]. However, we did not find this necessary in our computational experiments.

In each iteration of the Newton's method, one has to solve the linear system of the form (4.8), which we propose to do by the Conjugate Gradient (CG) method. The convergence rate of the CG method depends on the effective condition number

$$\frac{\lambda_1(\nabla^2 f(\mathbf{y}))}{\lambda_s(\nabla^2 f(\mathbf{y}))}$$

of $\nabla^2 f(\mathbf{y})$. The following theorem gives a theoretical estimate of this number and the proof can be found in the Appendix.

Theorem 4.1. *For the Hessian matrix* $\nabla^2 f(\mathbf{y})$*,*

$$2^n \cdot e^{-2(m\cdot 2^n+1)\cdot\|y\|_\infty} \leq \frac{\lambda_1(\nabla^2 f(\mathbf{y}))}{\lambda_s(\nabla^2 f(\mathbf{y}))} \leq \left(\sqrt{2^n}+\sqrt{m}\right)^2 \cdot e^{2(m\cdot 2^n+1)\cdot\|y\|_\infty}. \tag{4.9}$$

Remark: The main computational cost in each Newton iteration comes from the cost of solving the Hessian matrix system (4.8) by the CG method. In each CG iteration, the main computational cost comes from the matrix-vector multiplication of size $(m2^n+1)$ and the complexity is therefore of $O(m^3 2^{3n})$. Thus our algorithm is not computationally efficient for large value of m – e.g. for context-sensitive PBNs or PBNs with gene perturbations [30]. This is because our method is not efficient for direct application to the dense transition probability matrices in such PBNs.

5. Numerical Results

In this section, we first present some numerical examples of PBNs to demonstrate the proposed method. We then discuss the application of our method to the case of a BN with a small perturbation by a numerical example. Finally we consider a PBN of three genes with small perturbations. Regarding the numerical methods, for Newton's method we set the tolerance to be 10^{-7} while the tolerance of the CG method is 10^{-10}.

5.1. A Numerical Example of a PBN

In this subsection, we present an example to demonstrate the computational performance of our proposed method. In particular, the example gives a detailed explanation of our construction method.

Example: We consider the case $n = 2$ and $m = 2$ and suppose that the observed transition-probability matrix of the PBN is

$$A_{2,2} = \begin{pmatrix} 0.1 & 0.3 & 0.5 & 0.6 \\ 0.0 & 0.7 & 0.0 & 0.0 \\ 0.0 & 0.0 & 0.5 & 0.0 \\ 0.9 & 0.0 & 0.0 & 0.4 \end{pmatrix}.$$

We would like to find the following decomposition

$$A_{2,2} = \sum_{k \in S} q_k A_k \quad \text{where} \quad \sum_{k \in S} q_k = 1 \quad \text{and} \quad q_k \geq 0 \tag{5.1}$$

for some Boolean network matrices A_k and some set S. From the structure of $A_{2,2}$, it is straightforward to check that there are at most 16 (i.e. $|S| = 16$) possible Boolean network matrices for constituting the PBN, because the non-zero positions of a Boolean network matrix constituting the given transition-probability matrix should match with $A_{2,2}$. Thus any Boolean network matrix other than those 16 Boolean network matrices above will take no part in (5.1). Moreover, the transition-probability matrix $A_{2,2}$ can be written as a weighted sum of the 16 Boolean network matrices. In fact, there are many possible ways to do so as the number of variables q_i is more than the number of constraints (see Eq. (5.2)). All 16 possible Boolean matrices are listed below:

$$A_1 = \begin{pmatrix} 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}, A_2 = \begin{pmatrix} 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}, A_3 = \begin{pmatrix} 1 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}, A_4 = \begin{pmatrix} 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix},$$

$$A_5 = \begin{pmatrix} 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}, A_6 = \begin{pmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}, A_7 = \begin{pmatrix} 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}, A_8 = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix},$$

$$A_9 = \begin{pmatrix} 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{pmatrix}, A_{10} = \begin{pmatrix} 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 \end{pmatrix}, A_{11} = \begin{pmatrix} 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \end{pmatrix}, A_{12} = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 \end{pmatrix},$$

$$A_{13} = \begin{pmatrix} 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{pmatrix}, A_{14} = \begin{pmatrix} 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 \end{pmatrix}, A_{15} = \begin{pmatrix} 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \end{pmatrix}, A_{16} = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 \end{pmatrix}.$$

Now we have

$$A = \sum_{i=1}^{16} q_i A_i$$

and the 8 equations governing q_i (cf. (3.2)) are as follows:

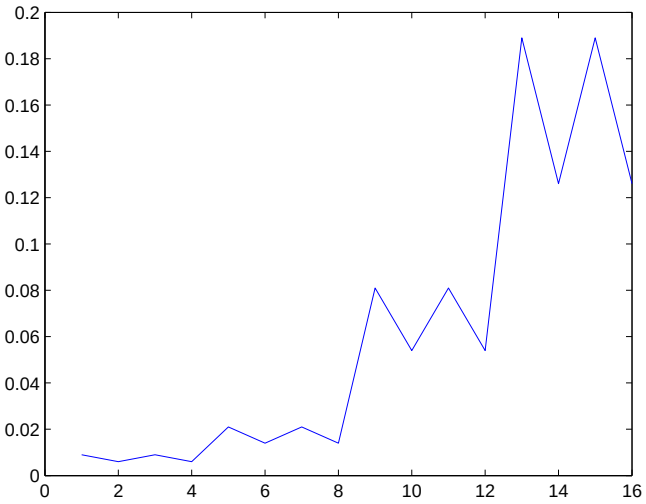

Figure 1: The Probability Distribution $\mathbf{q}$ for the Case of $A_{2,2}$.

$$\left(\begin{array}{cccc|cccc|cccc|cccc}
1&1&1&1&1&1&1&1&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&1&1&1&1&1&1&1&1\\
\hline
1&1&1&1&0&0&0&0&1&1&1&1&0&0&0&0\\
0&0&0&0&1&1&1&1&0&0&0&0&1&1&1&1\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
\hline
1&1&0&0&1&1&0&0&1&1&0&0&1&1&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&1&1&0&0&1&1&0&0&1&1&0&0&1&1\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
\hline
1&0&1&0&1&0&1&0&1&0&1&0&1&0&1&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&1&0&1&0&1&0&1&0&1&0&1&0&1&0&1
\end{array}\right)
\left(\begin{array}{c}
q_1\\q_2\\q_3\\q_4\\q_5\\q_6\\q_7\\q_8\\q_9\\q_{10}\\q_{11}\\q_{12}\\q_{13}\\q_{14}\\q_{15}\\q_{16}
\end{array}\right)
=
\left(\begin{array}{c}
0.1\\0.0\\0.0\\0.9\\\hline 0.3\\0.7\\0.0\\0.0\\\hline 0.5\\0.0\\0.5\\0.0\\\hline 0.6\\0.0\\0.0\\0.4
\end{array}\right). \quad (5.2)$$

Using our maximum entropy approach, we obtain the solution as shown in Fig. 1. From the solution, we note that the re-constructed PBN is supposed to be dominated (over 78%) by the 9th, the 11st and the last 4 BNs. From the dominant BNs, one can therefore construct the underlying regulatory rules – i.e. their truth tables. Here we see that our method can be used to identify the major components of the BNs constituting the PBN. From the truth tables, we may infer that (i) $(0,0) \longleftrightarrow (1,1)$ is an attractor cycle of period two. Moreover, it is possible that (ii) $(0,1) \longleftrightarrow (0,1)$ and (iii) $(1,0) \longleftrightarrow (1,0)$ and (iv) $(1,1) \longleftrightarrow (1,1)$ are attractors of period one.

The steady-state probability distribution of $A_{2,2}$ is $(0.4000, 0.0000, 0.0000, 0.6000)^t$. If we approximate $A_{2,2}$ by using the six major BNs A_9, A_{11}, A_{13}, A_{14}, A_{15} and A_{16} with a

Table 2: The steady-state probability distributions with different approximations.

Number of BNs dropped	$\|\|\tilde{p}-p\|\|_2^2$	$\tilde{p}$
0	7.9585×10^{-10}	$(0.4000, 0.0000, 0.0000, 0.6000)^t$
2	4.2487×10^{-10}	$(0.4000, 0.0000, 0.0000, 0.6000)^t$
4	0.0103	$(0.3927, 0.0000, 0.0000, 0.6073)^t$
6	0.0103	$(0.3927, 0.0000, 0.0000, 0.6073)^t$
8	0.0354	$(0.3750, 0.0000, 0.0000, 0.6250)^t$
10	0.0076	$(0.4054, 0.0000, 0.0000, 0.5946)^t$
12	1.2329	$(0.0000, 1.0000, 0.0000, 0.0000)^t$

normalization – i.e. we approximate $A_{2,2}$ by

$$\tilde{A}_{2,2} = \frac{1}{0.78}(0.08A_9 + 0.08A_{11} + 0.19A_{13} + 0.12A_{14} + 0.19A_{15} + 0.12A_{16}).$$

Then the steady-state probability distribution of $\tilde{A}_{2,2}$ is $(0.4054, 0.0000, 0.0000, 0.5946)^t$ with the initial probability distribution $(0.25, 0.25, 0.25, 0.25)^t$. This shows that our method gives a good approximation of the PBN. Furthermore, in Table 2 we give the approximations of the steady-state probability distributions obtained when we drop a different number of BNs – viz. from the smallest selection probability to the highest selection probability. Here we observe that the threshold of choosing the dropped BNs can directly influence the sensitivity of the performance with respect to both steady-state probability distribution and dominated BNs. To determine the dominant BNs, one can consider the value of $\sum_{i=1}^{K} q_i$ when the K least important BNs are dropped. In our example, we can choose the threshold to be 20%; and we see that even when we dropped the first ten least important BNs (22%), the approximation obtained by our method is still reasonably good.

Table 3: Number of Iterations.

n	m	Number of BNs	Newton's Iterations	Average Number of CG Iterations
2	2	16	9	9
2	3	81	7	9
3	2	256	7	7
3	3	6561	11	13

Finally, we present the number of Newton's iterations required for convergence and the average number of CG iterations in each Newton's iteration in Table 3.

5.2. BNs with Small Perturbations

Here we consider a special class of PBNs – viz. BNs with small perturbation [6]. Brun et al. [6] have derived a relation between the steady-state probability distribution of a PBN and the structure of attractors in a BN. Let $\mathbf{u}$ and $\mathbf{w}$ be states of a BN at time t and $t+1$,

respectively. Since in a BN the state at time $t+1$ is given deterministically from the state at time t, we can write $\text{Prob}(\mathbf{v}(t+1)=\mathbf{w}|\mathbf{v}(t)=\mathbf{u})=1$. For any other state $\mathbf{w}'$, one can write $\text{Prob}(\mathbf{v}(t+1)=\mathbf{w}'|\mathbf{v}(t)=\mathbf{u})=0$. The dynamics of the BN can be represented by a $2^n\times 2^n$ matrix B. Here we construct a variant PBN by introducing slight perturbation to this BN as follows. If state $\mathbf{w}$ is the next state to $\mathbf{u}$ in the above BN, we let

$$\text{Prob}(\mathbf{v}(t+1)=\mathbf{w}|\mathbf{v}(t)=\mathbf{u})=1-\frac{2^n-1}{2^n}\epsilon.$$

For other $\mathbf{w}'$, we let

$$\text{Prob}(\mathbf{v}(t+1)=\mathbf{w}'|\mathbf{v}(t)=\mathbf{u})=\frac{\epsilon}{2^n},$$

where $\epsilon\in(0,1]$. We consider the case when ϵ is positive but small. Let $B(\epsilon)$ be the $2^n\times 2^n$ matrix corresponding to these transition probabilities. Then we have

$$B(\epsilon)=(1-\epsilon)B+\epsilon\mathbf{u}\mathbf{1}^t$$

where

$$\mathbf{u}=\frac{1}{2^n}(1,1,\cdots,1)^t \quad \text{and} \quad \mathbf{1}^t=(1,1,\cdots,1).$$

Here we re-visit the example of a 2-gene PBN in Section 2. The transition-probability matrix of the BN is given in (2.2). There are two attractors. One is {State 3, State 4} and the other is State 1, with basin {State 1, State 2}. By introducing slight perturbation to this BN, we can get a variant PBN. Note that the attractors in the PBN are the same as those in the original BN. The transition-probability matrix of the PBN obtained by small perturbation of the BN is

$$B(\epsilon)=\begin{pmatrix} 1-\frac{3\epsilon}{4} & 1-\frac{3\epsilon}{4} & \frac{\epsilon}{4} & \frac{\epsilon}{4} \\ \frac{\epsilon}{4} & \frac{\epsilon}{4} & \frac{\epsilon}{4} & \frac{\epsilon}{4} \\ \frac{\epsilon}{4} & \frac{\epsilon}{4} & \frac{\epsilon}{4} & 1-\frac{3\epsilon}{4} \\ \frac{\epsilon}{4} & \frac{\epsilon}{4} & 1-\frac{3\epsilon}{4} & \frac{\epsilon}{4} \end{pmatrix}.$$

We then apply our algorithm to this example. Since each column of the transition matrix has one non-zero entry, a BN matrix can be represented by giving the position of the non-zero entry. We then adopt this representation in Table 4. For $\epsilon=0.01,0.02,0.03,0.04$, we apply our algorithm and get only one major BN (the original BN, the 6th BN in Table 4) in the solution – the other BNs have probability less than 0.01. While for $\epsilon=0.05,0.06,0.07,\cdots,0.20$, apart from the original BN we find 12 other BNs in our solution (see Table 4). The numerical results indicate that our proposed method can recover the major BNs of a PBN when the perturbation is reasonably small in this example.

Table 4: The 13 Major BNs.

BNs					$q_i(\epsilon=0.01)$	$q_i(\epsilon=0.03)$	$q_i(\epsilon=0.05)$	$q_i(\epsilon=0.10)$	$q_i(\epsilon=0.20)$
1	1	1	1	3	0.00	0.00	0.01	0.02	0.03
2	1	1	2	3	0.00	0.00	0.01	0.02	0.03
3	1	1	3	3	0.00	0.00	0.01	0.02	0.03
4	1	1	4	1	0.00	0.00	0.01	0.02	0.03
5	1	1	4	2	0.00	0.00	0.01	0.02	0.03
6	**1**	**1**	**4**	**3**	**0.97**	**0.91**	**0.86**	**0.73**	**0.52**
7	1	1	4	4	0.00	0.00	0.01	0.02	0.03
8	1	2	4	3	0.00	0.00	0.01	0.02	0.03
9	1	3	4	3	0.00	0.00	0.01	0.02	0.03
10	1	4	4	3	0.00	0.00	0.01	0.02	0.03
11	2	1	4	3	0.00	0.00	0.01	0.02	0.03
12	3	1	4	3	0.00	0.00	0.01	0.02	0.03
13	4	1	4	3	0.00	0.00	0.01	0.02	0.03

Table 5: Truth Table (Taken from [32]).

$x_1x_2x_3$	$f_1^{(1)}$	$f_2^{(1)}$	$f_1^{(2)}$	$f_1^{(3)}$	$f_2^{(3)}$
000	0	0	0	0	0
001	1	1	1	0	0
010	1	1	1	0	0
011	1	0	0	1	0
100	0	0	1	0	0
101	1	1	1	1	0
110	1	1	0	1	0
111	1	1	1	1	1
$c_j^{(i)}$	0.6	0.4	1	0.5	0.5

Table 6: The Four BNs.

BN_1	1	7	7	6	3	8	6	8
BN_2	1	7	7	5	3	7	5	8
BN_3	1	7	7	2	3	8	6	8
BN_4	1	7	7	1	3	7	5	8

5.3. A Three-gene Network with Small Perturbations

Shmulevich, et al. [32] proposed a PBN consisting of three genes $V = (x_1, x_2, x_3)$ and the function set $F = (F_1, F_2, F_3)$, where

$$F_1 = \left\{f_1^{(1)}, f_2^{(1)}\right\}, \quad F_2 = \left\{f_1^{(2)}\right\}, \quad \text{and } F_3 = \left\{f_1^{(3)}, f_2^{(3)}\right\}.$$

The functions are shown in Table 5, and the corresponding BNs are then given in Table 6.

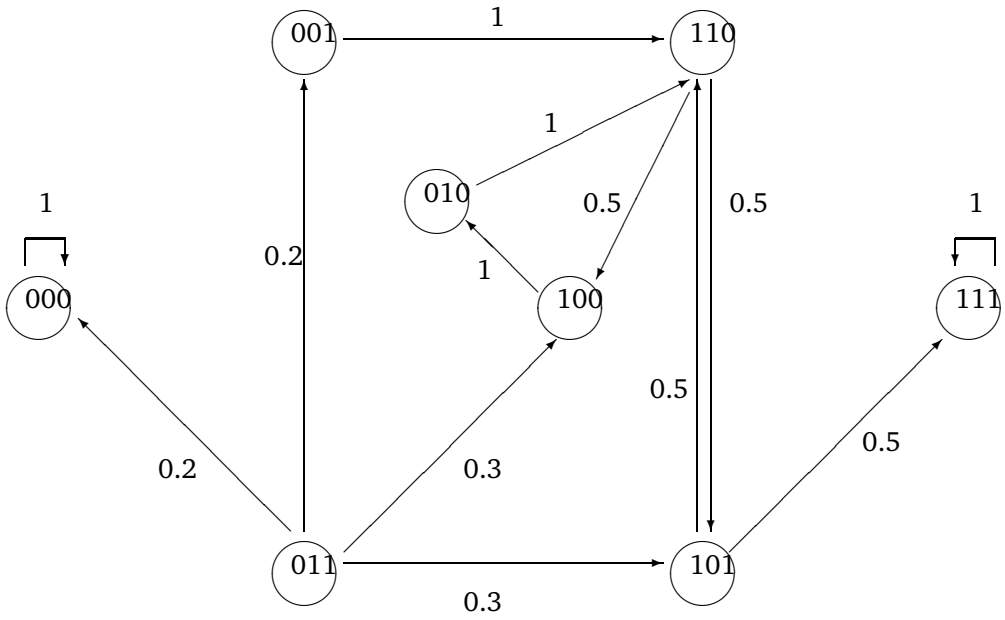

Figure 2: State Transition Diagram (taken from [32]).

The transition-probability matrix is given by

$$A_{4,4} = \begin{pmatrix} 1.0 & 0.0 & 0.0 & 0.2 & 0.0 & 0.0 & 0.0 & 0.0 \\ 0.0 & 0.0 & 0.0 & 0.2 & 0.0 & 0.0 & 0.0 & 0.0 \\ 0.0 & 0.0 & 0.0 & 0.0 & 1.0 & 0.0 & 0.0 & 0.0 \\ 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\ 0.0 & 0.0 & 0.0 & 0.3 & 0.0 & 0.0 & 0.5 & 0.0 \\ 0.0 & 0.0 & 0.0 & 0.3 & 0.0 & 0.0 & 0.5 & 0.0 \\ 0.0 & 1.0 & 1.0 & 0.0 & 0.0 & 0.5 & 0.0 & 0.0 \\ 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.5 & 0.0 & 1.0 \end{pmatrix}.$$

The state transition diagram corresponding to the transition-probability matrix $A_{4,4}$ is shown in Fig. 2.

We then consider adding some perturbations to the first two rows and the non-zeros entries of the transition probability as follows:

$$A_{4,4} = \begin{pmatrix} 1.0-\delta & \delta & \delta & 0.2+\delta & \delta & \delta & \delta & \delta \\ \delta & \delta & \delta & 0.2+\delta & \delta & \delta & \delta & \delta \\ 0.0 & 0.0 & 0.0 & 0.0 & 1.0-2\delta & 0.0 & 0.0 & 0.0 \\ 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\ 0.0 & 0.0 & 0.0 & 0.3-\delta & 0.0 & 0.0 & 0.5-\delta & 0.0 \\ 0.0 & 0.0 & 0.0 & 0.3-\delta & 0.0 & 0.0 & 0.5-\delta & 0.0 \\ 0.0 & 1.0-2\delta & 1.0-2\delta & 0.0 & 0.0 & 0.5-\delta & 0.0 & 0.0 \\ 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.5-\delta & 0.0 & 1.0-2\delta \end{pmatrix}.$$

Table 7: The 16 Major BNs.

BNs									q_i ($\delta = 0.01$)	q_i ($\delta = 0.02$)	q_i ($\delta = 0.03$)	q_i ($\delta = 0.04$)
1	1	7	7	1	3	7	5	8	0.047	0.045	0.042	0.040
2	1	7	7	1	3	7	6	8	0.047	0.045	0.042	0.040
3	1	7	7	1	3	8	5	8	0.047	0.045	0.042	0.040
4	1	7	7	1	3	8	6	8	0.047	0.045	0.042	0.040
5	1	7	7	2	3	7	5	8	0.047	0.045	0.042	0.040
6	1	7	7	2	3	7	6	8	0.047	0.045	0.042	0.040
7	1	7	7	2	3	8	5	8	0.047	0.045	0.042	0.040
8	1	7	7	2	3	8	6	8	0.047	0.045	0.042	0.040
9	1	7	7	5	3	7	5	8	0.071	0.067	0.063	0.059
10	1	7	7	5	3	7	6	8	0.071	0.067	0.063	0.059
11	1	7	7	5	3	8	5	8	0.071	0.067	0.063	0.059
12	1	7	7	5	3	8	6	8	0.071	0.067	0.063	0.059
13	1	7	7	6	3	7	5	8	0.071	0.067	0.063	0.059
14	1	7	7	6	3	7	6	8	0.071	0.067	0.063	0.059
15	1	7	7	6	3	8	5	8	0.071	0.067	0.063	0.059
16	1	7	7	6	3	8	6	8	0.071	0.067	0.063	0.059

For $\delta = 0.01, 0.02, 0.03$ and 0.04, we apply our algorithm and obtain 16 major BNs (Table 7) and these BNs actually contribute around respectively 94%, 90%, 84% and 79% of the network. We note that the 1st, 8th, 9th and the last major BNs match with the four BNs (BN_1, BN_2, BN_3, BN_4) in Table 6.

6. Discussion

In this paper, we present the problem of constructing a PBN from a given transition-probability matrix and a given set of BNs. It is an inverse problem of huge size. We propose a maximum entropy approach for solving the problem. Newton's method is then applied in combination with the CG method, to solve the inverse problem involving the Hessian matrix system. We also give a convergence rate analysis for the proposed method. Further research on preconditioning the CG method to accelerate its convergence might be done, or to apply the proposed method to more real genetic regulatory networks. Finally, one can also extend the proposed model by using the relative entropy approach as discussed in the following subsection.

6.1. Extension of the Model by Relative Entropy Approach

Our method may be extended if there is further known information. In particular, we may have an initial guess of $\mathbf{q}$ denoted by $\tilde{\mathbf{q}}$ and would like to improve this estimation. One possible way is consider minimizing the relative entropy of $\tilde{\mathbf{q}}$ from $\mathbf{q}$ [8, 39, 42, 43] –

viz.

$$-\sum_{i=1}^{m^{2^n}} q_i \log \tilde{q}_i + \sum_{i=1}^{m^{2^n}} q_i \log q_i.$$

The problem can be formulated as a maximization problem – viz.

$$\max_{\mathbf{q}} \left\{ \sum_{i=1}^{m^{2^n}} q_i \log \tilde{q}_i - \sum_{i=1}^{m^{2^n}} q_i \log q_i \right\}$$

subject to

$$\bar{U}\mathbf{q} = \bar{\mathbf{p}} \quad \text{and} \quad 0 \le q_i.$$

In steps similar to Section 4, one can solve this maximization problem via its dual problem

$$\min_{\mathbf{y}} \left\{ \sum_{i=1}^{m^{2^n}} e^{\log \tilde{q}_i - 1 - \mathbf{y}^T \bar{U}_{\cdot i}} + \mathbf{y}^T \bar{\mathbf{p}} \right\}. \tag{6.1}$$

Thus we can again transform a constrained maximization problem with m^{2^n} variables into an unconstrained minimization problem of $m \cdot 2^n + 1$ variables, and apply Newton's method together with the CG method to solve (6.1). Since the Hessian matrix in this case shares the same structure as before, the convergence analysis in Section 4.2 can also be applied.

7. Appendix

7.1. Proof of Theorem 3.2

Let $\mathbf{1}$ be a $m^{2^n} \times 1$ column vector where entries are 1. Then

$$\bar{U} = \begin{pmatrix} U \\ \mathbf{1}^T \end{pmatrix}.$$

From perturbation theory of singular values [38] and Theorem 3.1, we have

$$\sqrt{\lambda_1(\bar{U}\bar{U}^T)} \le \sqrt{\lambda_1(UU^T)} + \|\mathbf{1}\|_2 = \sqrt{2^n \cdot m^{2^n - 1}} + \sqrt{m^{2^n}}.$$

In terms of the definitions of U and $\bar{U}$, we know that

$$\text{rank}(U) = \text{rank}(\bar{U}) = (m-1) \cdot 2^n + 1.$$

Since

$$\bar{U}\bar{U}^T = \begin{pmatrix} UU^T & U\mathbf{1} \\ \mathbf{1}^T U^T & \mathbf{1}^T \mathbf{1} \end{pmatrix},$$

applying the Interlace Theorem [20, p.184] to the above matrix we get the result (b) such that

$$\frac{2^n \cdot m^{2^n-1}}{m^{2^n-1}} \le \frac{\lambda_1(\bar{U}\bar{U}^T)}{\lambda_s(\bar{U}\bar{U}^T)} \le \frac{\left(\sqrt{2^n \cdot m^{2^n-1}} + \sqrt{m^{2^n}}\right)^2}{m^{2^n-1}},$$

which yields (c) so the proof is complete.

7.2. Proof of Theorem 4.1

In view of (4.4) and the definition of $\bar{U}$, we have

$$e^{-1-(m\cdot 2^n+1)\cdot\|\mathbf{y}\|_\infty} \cdot \mathbf{1} \le \mathbf{q}^*(\mathbf{y}) \le e^{-1+(m\cdot 2^n+1)\cdot\|\mathbf{y}\|_\infty} \cdot \mathbf{1},$$

where $\mathbf{1}$ is again the vector with all entries 1, and hence

$$e^{-1-(m\cdot 2^n+1)\cdot\|\mathbf{y}\|_\infty}\bar{U}\bar{U}^T \le \bar{U}\mathrm{diag}(\mathbf{q}^*(\mathbf{y}))\bar{U}^T \le e^{-1+(m\cdot 2^n+1)\cdot\|\mathbf{y}\|_\infty}\bar{U}\bar{U}^T.$$

By Theorem 3.2 and Corollary 4.3.3 in [20] we can get

$$\frac{\lambda_1(\nabla^2 f(\mathbf{y}))}{\lambda_s(\nabla^2 f(\mathbf{y}))} \le \frac{\lambda_1\left(\bar{U}\bar{U}^T\right)}{\lambda_s\left(\bar{U}\bar{U}^T\right)} \cdot e^{2(m\cdot 2^n+1)\cdot\|y\|_\infty} \le \left(\sqrt{2^n}+\sqrt{m}\right)^2 \cdot e^{2(m\cdot 2^n+1)\cdot\|y\|_\infty}$$

and

$$\frac{\lambda_1(\nabla^2 f(\mathbf{y}))}{\lambda_s(\nabla^2 f(\mathbf{y}))} \ge \frac{e^{-1-(m\cdot 2^n+1)\cdot\|\mathbf{y}\|_\infty} \cdot 2^n \cdot m^{2^n-1}}{e^{-1+(m\cdot 2^n+1)\cdot\|\mathbf{y}\|_\infty} \cdot m^{2^n-1}} = 2^n \cdot e^{-2(m\cdot 2^n+1)\cdot\|y\|_\infty}.$$

This yields (4.9) and the proof is complete.

Acknowledgments

This paper was presented at the Workshop on Scientific Computing and Matrix Analysis, held on the occasion of the tenth anniversary of the Macao handover at the University of Macao, and dedicated to the 20th anniversary of the TOTS Group founded by Prof. Raymond H. Chan. The research was supported in part by HKRGC Grant No. 7017/07P, HKUCRGC Grants, the HKU Strategy Research Theme fund on Computational Sciences, a Hung Hing Ying Physical Research Sciences Research Grant, the National Natural Science Foundation of China Grant No. 10971075 and Guangdong Provincial Natural Science Grant No. 9151063101000021.

References

[1] T. Akutsu, M. Hayasida, W. Ching and M. Ng, (2007), Control of Boolean Networks: Hardness Results and Algorithms for Tree Structured Networks, Journal of Theoretical Biology, (244), 670-679.

[2] M. Aldana, (2003), Boolean Dynamics of Networks with Scale-free Topology, Physica D (185), 45-66.

[3] O. Axelsson, (1996), Iterative Solution Methods, Cambridge University Press, Cambridge, UK.

[4] G. Boole, (1847), Mathematical Analysis of Logic, Being an Essay Towards a Calculus of Deductive Reasoning, Reprint: Blackwell, Oxford, 1948.

[5] G. Boole, (1854), An Investigation of the Laws of Thought, on Which are Founded the Mathematical Theories of Logic and Probabilities, Reprint: Dover Publications, New York, 1958.

[6] M. Brun, E. Dougherty and I. Shmulevich, (2005), Steady-State Probabilities for Attractors in Probabilistic Boolean Networks, Signal Processing, 85 (4), 1993-2013.

[7] J. E. Celis, M. Kruhøfferm, I. Gromova, C. Frederiksen, M. Østergaard and T. F. Ørntoft, (2000), Gene Expression Profiling: Monitoring Transcription and Translation Products Using DNA Microarrays and Proteomics, FEBS Lett. 480 (1), 2-16.

[8] W. Ching, S. Scholtes and S. Zhang, (2004), Numerical Algorithms for Estimating Traffic Between Zones in a Network, Engineering Optimisation, (36), 379-400.

[9] W. Ching, E. Fung, M. Ng and T. Akutsu, (2005), On Construction of Stochastic Genetic Networks Based on Gene Expression Sequences, International Journal of Neural Systems, (15), 297-310.

[10] W. Ching and M. Ng, (2006), Markov Chains : Models, Algorithms and Applications, International Series on Operations Research and Management Science, Springer: New York.

[11] W. Ching, S. Zhang, M. Ng and T. Akutsu, (2007), An Approximation Method for Solving the Steady-state Probability Distribution of Probabilistic Boolean Networks, Bioinformatics, (23), 1511-1518.

[12] W. Ching, S. Zhang, Y. Jiao, T. Akutsu and A. Wong, (2009), Optimal Control Policy for Probabilistic Boolean Networks with Hard Constraints. IET on Systems Biology, (3), 90-99.

[13] W. Ching and Y. Cong, (2009), A New Optimization Model for the Construction of Markov Chains, CSO2009, Hainan, IEEE Computer Society Proceedings, 551-555.

[14] A. Conn, B. Gould and P. Toint, (2000), Trust-region Methods, SIAM Publications, Philadelphia.

[15] A. Datta, A. Choudhary, M. Bitter and E. R. Dougherty, (2003), External Control in Markovian Genetic Regulatory Networks, Machine Learning, (52), 169-191.

[16] H. de Jong, (2002), Modeling and Simulation of Genetic Regulatory Systems: A Literature Review, Journal of Computational Biology, (9), 69-103.

[17] J. Dennis and R. Schnabel, (1983), Numerical Methods for Unconstrained Optimisation and Nonlinear Equations, Prentice Hall, Englewood Cliffs.

[18] E. Dougherty, S. Kim and Y. Chen, (2000), Coefficient of Determination in Nonlinear Signal Processing, Signal Processing, (80), 2219-2235.

[19] S. Huang and D. E. Ingber, (2000), Shape-dependent Control of Cell Growth, Differentiation, and Apoptosis: Switching Between Attractors in Cell Regulatory Networks, Experimental Cell Research, (261), 91-103.

[20] R. Horn and C. Johnson, (1985), Matrix Analysis, Cambridge University Press, Cambridge.

[21] S. Kauffman, (1969), Metabolic Stability and Epigenesis in Randomly Constructed Gene Nets, Journal of Theoretical Biology, (22), 437-467.

[22] S. Kauffman, (1969), Homeostasis and Differentiation in Random Genetic Control Networks, Nature, (224), 177-178.
[23] S. Kauffman, (1974), The Large Scale Structure and Dynamics of Genetic Control Circuits: An Ensemble Approach, Journal of Theoretical Biology, (44), 167-190.
[24] S. Kauffman, (1993), The Origins of Order: Self-organization and Selection in Evolution, New York: Oxford University Press.
[25] S. Kim, S. Imoto and S. Miyano, (2003), Dynamic Bayesian Network and Nonparametric Regression for Nonlinear Modeling of Gene Networks from time Series Gene Expression Data, Proc. 1st Computational Methods in Systems Biology, Lecture Note in Computer Science, (2602), 104-113.
[26] S. Huang, (1999), Gene Expression Profiling, Genetic Networks and Cellular States: An Integrating Concept for Tumorigenesis and Drug Discovery, Journal of Molecular Medicine, (77), 469-480.
[27] H. Mortveit and M. Reidys, (2007), An Introduction to Sequential Dynamical Systems, Springer Verlag, New York.
[28] J. Nocedal and S. Wright, (1999), Numerical Optimisation, Springer-Verlag, New York.
[29] R. Pal, I. Ivanov, A. Datta, M. Bittner and E. Dougherty, (2005), Generating Boolean Networks with a Prescribed Attractor Structure, Bioinformatics, (21), 4021-4025.
[30] R. Pal, A. Datta, M. Bittner and E. Dougherty, (2005), Intervention in Context-sensitive Probabilistic Boolean Networks, Bioinformatics, (21) 1211-1218.
[31] C. E. Shannon, (1948), A Mathematical Theory of Communication, Bell System Technical Journal, (27), 379-423.
[32] I. Shmulevich, E. Dougherty, S. Kim and W. Zhang, (2002), Probabilistic Boolean Networks: A Rule-based Uncertainty Model for Gene Regulatory Networks, Bioinformatics, (18), 261-274.
[33] I. Shmulevich, E. Dougherty, S. Kim and W. Zhang, (2002), Control of Stationary Behavior in Probabilistic Boolean Networks by Means of Structural Intervention, Journal of Biological Systems, (10), 431-445.
[34] I. Shmulevich, E. Dougherty, S. Kim and W. Zhang, (2002), From Boolean to Probabilistic Boolean Networks as Models of Genetic Regulatory Networks, Proceedings of the IEEE, (90), 1778-1792.
[35] I. Shmulevich and E. Dougherty, (2007), Genomic Signal Processing, Princeton University Press, U.S.
[36] P. Smolen, D. Baxter and J. Byrne, (2000), Mathematical Modeling of Gene Network, Neuron, (26), 567-580.
[37] R. Somogyi and C. Sniegoski, (1996), Modeling the Complexity of Gene Networks: Understanding Multigenic and Pleiotropic Regulation, Complexity, (1), 45-63.
[38] G. W. Stewart and J. G. Sun, (1990), Matrix Perturbation Theory, Academic Press, Boston.
[39] H. Van Zuylen and L. Willumsen, (1980), The Most Likely Trip Matrix Estimated from Traffic Counts, Transportation Research, (14)(B), 281-293.
[40] G. Vahedi, I. Ivanov and E. Dougherty, (2009), Inference of Boolean Networks Under Constraint on Bidirectional Gene Relationships, IET Systems Biology, (3), 191-202.
[41] A. Wilson, (1970), Entropy in Urban and Regional Modelling, Pion, London.
[42] H. Yang, T. Sasaki, Y. Iida and Y. Asakura, (1992), Estimation of Origin-destination Matrices from Link Traffic Counts on Congested Network, Transportation Research, (26)(B), 417-434.
[43] H. Yang, Y. Iida and T. Sasaki, (1994), The Equilibrium-based Origin-destination Estimation Problem, Transportation Research, (28)(B), 23-33.
[44] S. Zhang, W. Ching, M. Ng and T. Akutsu, (2007), Simulation Study in Probabilistic Boolean Network Models for Genetic Regulatory Networks, Journal of Data Mining and Bioinformat-

ics, (1), 217-240.

[45] S. Zhang, W. Ching, N. Tsing, H. Leung and D. Guo, (2008), A Multiple Regression Approach for Building Genetic Networks, Proceedings of the International Conference on BioMedical Engineering and Informatics (BMEI2008) Sanya, China (in CD-ROM).