A Modified Newton's Method for Inverse Problem of Probabilistic Boolean Networks with Gene Perturbations

Wen Li
School of Mathematical Sciences,
South China Normal University,
Guangzhou, China, 510631.
Email : liwen@scnu.edu.cn

Wai-Ki Ching
Advanced Modeling and Applied Computing Laboratory,
Department of Mathematics,
The University of Hong Kong,
Pokfulam Road, Hong Kong.
E-mail: wching@hku.hk.

Lu-Bin Cui
School of Mathematical Sciences,
South China Normal University,
Guangzhou, China, 510631.
E-mail: hnzkc@163.com.

***Abstract*—Modeling genetic regulatory networks is an important research issue in systems biology. Many mathematical models have been proposed, and among these models, Boolean Network (BN) and its extension Probabilistic Boolean Network (PBN) are popular. In this paper we consider the problem constructing PBNs with gene perturbations. We propose a modified Newton's method to get the gene perturbation probability of the captured problem. Numerical experiments are given to demonstrate both effectiveness and efficiency of our proposed method.**

Keywords: Inversion Problem, Gene Perturbation, Modify Newton's Method, Probabilistic Boolean Networks.

I. Introduction

Mathematical modeling and computational study of genetic regulatory network are important topics in bioinformatics and have been studied in a number of literature [3], [7], [10]. Many mathematical models have been proposed in the literatures, e.g. directed graphs, Boolean networks (BNs) [1], [2], Probabilistic Boolean Networks (PBNs) [12], [13], Markov chain model [4], [6], multivariate Markov model [5] and many other models [15]. Among these models, BN and its extension PBN have attracted much attention [14].

BN was first introduced by Kauffman [8], [9]. In a BN model, the gene expression states are quantized to only two levels: on and off (represented as 1 and 0). The target gene is predicted by several genes called its input genes via a Boolean function. When the input genes and the Boolean functions are given, we say that a BN is defined. Later BNs have been extended to PBNs (stochastic models). In a PBN, for each gene, there can be more than one Boolean function and selection probabilities are assigned to the Boolean functions. The network dynamics of a PBN can be studied in a Markov chains framework [12]. The rich theory of Markov chains are then applicable to the analysis of a PBN. PBNs also provide a natural way to quantify the relative influence and sensitivity of genes in their interactions with other genes. The random gene perturbations are introduced into the PBN model in [13], where the perturbation describes the random inputs from the outside. The effect of introducing the random gene perturbations is to make the network stable and ensure the existence of the steady-state distribution. Extension to the context-sensitive PBN model has been introduced by Pal et al. [11].

The gene perturbation can be applied to gene control analysis. One can regard the perturbation probability p is the proportion of time that a control is applied [17]. Suppose we know the steady-state probability distribution $\tilde{\mathbf{x}}$ of a PBN with gene perturbation. And we would like to perturb the PBN to another PBN with the steady-state probability distributions $\tilde{\mathbf{x}}$, then one would be interested in knowing the gene perturbation probability p (or the proportion of time that a control was applied). Here we consider the inverse problem of PBNs with gene perturbations, that is to find the gene perturbation probability p when the transition probability matrix A and the steady-state probability distribution $\tilde{x}$ of PBNs with gene perturbations are given. We propose an effective method to find the perturbation probability p.

The rest of the paper is organized as follows. In Section 2, we give a brief review on the mathematical formulation of PBNs and PBNs with gene perturbations. We then introduce inverse problem of PBNs with gene perturbations in Section 3. Section 4 presents a modified Newton's method for the computation of perturbation probability p. Numerical experiments are given to demonstrate the effectiveness of the proposed method in Section 5. Finally we give a brief summary to conclude the paper.

II. Review on PBNs and PBNs with Gene Perturbations

In this section, we give a brief introduction to PBN and PBN with Gene Perturbations with some examples.

A. A Review of PBNs

In a PBN, each target gene has a number of Boolean functions can be chosen to be a prediction function. All

978-1-4577-1666-9/11/$26.00 ©2011 IEEE

TABLE I
THE TRUTH TABLE OF THE PBN

State	$v_1 v_2 v_3$	$f_1^{(1)}$	$f_2^{(1)}$	$f_1^{(2)}$	$f_1^{(3)}$	$f_2^{(3)}$
1	000	0	0	0	0	0
2	001	1	1	1	0	0
3	010	1	1	1	0	0
4	011	1	0	0	1	0
5	100	0	0	1	0	0
6	101	1	1	1	1	0
7	110	1	1	0	1	0
8	111	1	1	1	1	1
	$c_j^{(i)}$	0.6	0.4	1	0.5	0.5

these Boolean functions can be selected randomly with some probabilities. We assume that for the kth gene, there are $l(k)$ possible Boolean functions:

$$F^{(k)} = \{f_j^{(k)} \ : for \ j = 1, \ldots, l(k)\}$$

and the probability of choosing function $f_j^{(k)}$ is $c_j^{(k)}$, where $f_j^{(k)}$ is a function with respect to the activity levels of n genes. A PBN is said to be independent if the elements from different $F^{(k)}$ are independent. For an independent PBN of n genes, there are at most $N = \prod_{k=1}^{n} l(k)$ different possible BNs. This means that there are totally N possible realizations of the PBN. Suppose f_j is the jth possible realization,

$$f_j = [f_{j_1}^{(1)}, f_{j_2}^{(2)}, \ldots, f_{j_n}^{(n)}], \quad 1 \le j_k \le l(k), \ k = 1, 2, \ldots, n.$$

Assuming the selection process is independent, the probability to choose the jth realization is given by

$$p_j = \prod_{k=1}^{n} c_{j_k}^{(k)}, \ j = 1, 2, \ldots, N. \tag{II.1}$$

In fact, it can be show that transition probability matrix of the network states is

$$A = \sum_{j=1}^{N} p_j A_j$$

where A_j is the transition matrix corresponding to the jth BN.

Example II.1. *[12] Suppose we are given a PBN consisting of three genes $V = (v_1, v_2, v_3)$ and the function sets $F^{(1)} = \{f_1^{(1)}, \ f_2^{(1)}\}$, $F^{(2)} = \{f_1^{(1)}\}$ and $F^{(3)} = \{f_1^{(3)}, \ f_2^{(3)}\}$. Let the functions be given in Table I. The transition probability matrix is given by*

$$A = \begin{pmatrix} 1 & 0 & 0 & 0.2 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0.2 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0.3 & 0 & 0 & 0.5 & 0 \\ 0 & 0 & 0 & 0.3 & 0 & 0 & 0.5 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0.5 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0.5 & 0 & 1 \end{pmatrix}.$$

B. A Review of PBNs with Gene Perturbations

In this subsection, we introduce the PBN model with gene perturbations [13]. Random gene perturbation is the description of the random inputs from the outside due to external stimuli and this is meaningful in an open genome system. The effect of the random gene perturbations is to make the genes flip from state 1 to state 0 or vice versa. It makes the underlying Markov chain of the PBN ergodic and therefore all the 2^n states in the system are communicated. When random gene perturbation is introduced, the transition probability matrix $\tilde{A}$ is then given by

$$\tilde{A}(i,j) = (1-p)^n A(i,j) + \tilde{P}_n \equiv \hat{A}(i,j) + \tilde{P}_n, \tag{II.2}$$

where

$$[\tilde{P}_n]_{ij} = p^{h(v(i),v(j))}(1-p)^{n-h(v(i),v(j))} I_{v(i)\neq v(j)} \tag{II.3}$$

is the perturbation matrix. Here $h(v(i), v(j))$ is Hamming distance between the two vectors $v(i)$ and $v(j)$, p is the perturbation probability of each gene and $I_{v(i)\neq v(j)}$ is the indicator function:

$$I_{v(i)\neq v(j)} = \begin{cases} 1 & \text{if } \ v(i) \neq v(j), \\ 0 & \text{if } \ v(i) = v(j). \end{cases}$$

From Equation II.2, one can see that the final transition matrix $\tilde{A}$ is the sum of the transition matrix without perturbation A multiplied by $(1-p)^n$ and the perturbation matrix $\tilde{P}_n$. Given a PBN model, the Hamming distance $h(v(i), v(j))$ between the two vectors $v(i)$ and $v(j)$ is defined. Thus from Equation II.3, we know that the perturbation matrix $\tilde{P}_n$ only depends on the number of genes and the random gene perturbation probability. When the number of genes and the gene perturbation probability in different PBNs are same, the perturbation matrices are same. Xu et al.[17] studied the properties of the perturbed matrix and proposed expression of the perturbed matrix.

Theorem II.1. *[17] Let $\tilde{P}_n$ be the $2^n \times 2^n$ perturbed matrix of a PBN with n genes. Then we have for $n = 1, 2, \ldots$*

$$\tilde{P}_n = \underbrace{Q_1 \otimes Q_1 \otimes \cdots \otimes Q_1}_{n \ terms} - (1-p)^n I_{2^n} \tag{II.4}$$

where

$$Q_1 = \begin{pmatrix} 1-p & p \\ p & 1-p \end{pmatrix}.$$

The Kronecker product $\otimes$ for two matrices H $(n \times m)$ and B $(r \times s)$ is defined as an $(rn \times sm)$ matrix

$$H \otimes B = \begin{pmatrix} H_{11}B & \cdots & H_{1m}B \\ H_{21}B & \cdots & H_{2m}B \\ \vdots & \vdots & \vdots \\ H_{n1}B & \cdots & H_{nm}B \end{pmatrix}.$$

III. INVERSE PROBLEM OF PBNS WITH GENE PERTURBATIONS AND MODIFIED NEWTON'S METHOD

In this section, we first introduce the inverse problem of PBNs with gene perturbations, and then propose a modified Newton's method for computing the perturbation probability p. The gene perturbation can be applied to gene control analysis. And the perturbation probability p is the proportion of time that a control is applied [17]. The inverse problem is to find the perturbation probability p when the transition probability matrix A and the steady-state probability distribution $\tilde{\mathbf{x}}$ of PBNs with gene perturbations are known. That is we want to know the proportion of time that a control is applied in gene control problem if we want to control a gene changing to another gene with the steady-state probability distributions $\tilde{\mathbf{x}}$.

978-1-4577-1666-9/11/$26.00 ©2011 IEEE

From Theorem II.1, when random gene perturbation is introduced, the transition probability matrix is given by

$$\tilde{A} = A + \tilde{P}_n = (1-p)^n (A - I_{2^n}) + \underbrace{Q_1 \otimes \cdots \otimes Q_1}_{n \ terms}.$$

From $\tilde{A}\tilde{\mathbf{x}} = \tilde{\mathbf{x}}$, we have

$$\tilde{A}\tilde{\mathbf{x}} - \tilde{\mathbf{x}} = [(1-p)^n (A - I_{2^n}) + (\underbrace{Q_1 \otimes \cdots \otimes Q_1}_{n \ terms} - I_{2^n})]\tilde{\mathbf{x}} = \mathbf{0}.$$

Furthermore the perturbation probability satisfies $0 \le p \le 1$. The inverse problem for PBNs with gene perturbation can be modeled as follows: Let

$$Q_n = \underbrace{Q_1 \otimes \cdots \otimes Q_1}_{n \ terms} = Q_1 \otimes Q_{n-1}.$$

Then we have

$$\begin{gathered}[(1-p)^n (A - I_{2^n}) + (Q_n - I_{2^n})]\tilde{\mathbf{x}} = \mathbf{0} \\ \text{and} \quad 0 \le p \le 1.\end{gathered} \tag{III.1}$$

It is noted that to show that both existence and uniqueness of the solution for the inverse problem is difficult in theory. As we know that the inverse problem may have no solution because the distribution $\tilde{x}$ cannot be the steady-state distribution of the PBN with gene perturbation for any $0 \le p \le 1$. For example, suppose transition probability matrix A and the steady-state probability distribution $\tilde{x}$ of the PBN with gene perturbation are given as follow:

$$A = \begin{pmatrix} 1 & 0 & 0.5 & 0 \\ 0 & 0.2 & 0 & 0 \\ 0 & 0 & 0.5 & 1 \\ 0 & 0.8 & 0 & 0 \end{pmatrix}, \ \tilde{\mathbf{x}} = (0.2,\ 0.1,\ 0.4,\ 0.5)^T.$$

Our inverse problem is to solve the following equations,

$$\begin{cases} 0.20 - 0.30p + 0.40p^2 &= 0 \\ -0.08 + 0.66p - 0.28p^2 &= 0 \\ 0.30 - 0.70p + 0.10p^2 &= 0 \\ -0.42 + 0.34p - 0.22p^2 &= 0 \\ 0 \le p \le 1. \end{cases} \tag{III.2}$$

Because the first equation of (III.2) has no real root, so the system (III.2) has no real root too. But if we choose $\tilde{\mathbf{x}} = (0.9991, 0.0124, 0.0395, 0.0104)^T$ in the above example, one can get the solution $p = 0.01$ by using our proposed algorithm, the modified Newton's method, which will be introduced later. The nonlinear system may have more than one solution. It is very difficulty to show that the existence of the solution.

One possible way to solve the system (III.1) is to solve every equations in (III.1), then find the common solution(s) of every equations in (III.1). If these equations have no common solution(s), then we can conclude that Equations (III.1) have no solution. Here we call this method a direct method. But, there are 2^n nonlinear equations in (III.1). If the number of genes, n, is large, this method may require a lot of computational times. Therefore we have to find an efficient method to solve (III.1). Before we propose the method, we give some properties of the nonlinear equations (III.1).

Property III.1. *Denote $g_i(p) = 0,\ i = 1, 2, 3, \ldots, 2^n$ as the ith equation in (III.1). Let S be a nonempty proper subset of the set $\{1, 2, 3, \ldots, 2^n\}$, and*

$$G(p) = \sum_{i \in S} g_i(p).$$

If the polynomials in (III.1) have a solution p^* and $G(p)$ is not always equal to zero for arbitrary $0 \le p \le 1$, then the equation $G(p) = 0,\ 0 \le p \le 1$ have a solution p^*.

Now the converse-negative proposition of Property III.1 is the following.

Property III.2. *If $G(p) = 0 (0 \le p \le 1)$ has no real solution, then system (III.1) has no solution.*

So if $G(p) = 0,\ 0 \le p \le 1$ have solution, we could solve the inverse problem by the following step. We choose $j \notin S$ and solve $g_j(p) = 0$. Then, if $G(p) = 0,\ 0 \le p \le 1$ and $g_j(p) = 0$ have no common solution then the inverse have no solution. Else we choose another set $S' \subseteq \{1, 2, \ldots, N\}$ and do the similar step again.

In order to reduce the computational cost and use the information of the equations in (III.1), one may sum up all equations in (III.1) and then solve any one of the equations in (III.1) and find the common solution(s) of the two equations. But if we sum all equations in (III.1) together, we get nothing due to the following property.

Property III.3. *Let S be a nonempty proper subset of the set $\{1, 2, 3, \ldots, 2^n\}$,*

$$G_1(p) = \sum_{i \in S} g_i(p) \quad \text{and} \quad G_2(p) = \sum_{i \notin S} g_i(p).$$

Then we have $G_1(p) = -G_2(p)$. That is to say the equation $G_1(p) = 0$ is equivalent to $G_2(p) = 0$.

Proof: For simplicity, we use some mathematical symbols in MATLAB. For an arbitrary matrix $B \in \mathbb{R}^{m \times n}$, we denote by

$$B(r, :),\ 1 \le r \le m \quad \text{and}$$

$$\sum_{r=i}^{j} B(r,\ :),\ 1 \le i \le j \le m$$

the rth row of matrix B and the summation of row vectors of matrix B from ith to jth row, respectively.

Since A is the transition probability matrix, we have

$$\sum_{r=1}^{2^n} A(r,\ :) = (1, 1, \ldots, 1).$$

By the following the recursive relation

$$Q_n = Q_1 \otimes Q_{n-1} = \begin{pmatrix} (1-p)Q_{n-1} & pQ_{n-1} \\ pQ_{n-1} & (1-p)Q_{n-1} \end{pmatrix}$$

978-1-4577-1666-9/11/$26.00 ©2011 IEEE

we have

$$\begin{aligned}\sum_{r=1}^{2^n} Q_n(r,\ :) &= (1,\ 1) \otimes \sum_{r=1}^{2^{n-1}} Q_{n-1}(r,\ :) \\ &\vdots \\ &\overbrace{\qquad\qquad\qquad\qquad}^{n\ terms} \\ &= (1,\ 1) \otimes \cdots \otimes (1,\ 1) \otimes \sum_{r=1}^{2} Q_1(r,\ :) \\ &= (1, 1, \ldots, 1).\end{aligned} \tag{III.3}$$

Therefore we have

$$\begin{aligned}&G_1(p) + G_2(p) \\ &= \left(\sum_{r=1}^{2^n} [(1-p)^n (A - I_{2^n}) + (Q_n - I_{2^n})](r,:) \right) \tilde{\mathbf{x}} \\ &= [(1-p)^n ((1,1,\ldots,1) - (1,1,\ldots,1)) + \\ &\quad ((1,1,\ldots,1) - (1,1,\ldots,1))]\, \tilde{\mathbf{x}} = 0.\end{aligned} \tag{III.4}$$

Thus we have $G_1(p) = -G_2(p)$. ■

Remark III.1. *If we sum up all equations in (III.1), we have*

$$\sum_{i=1}^{2^n} g_i(p) = G_1(p) + G_2(p) = 0,$$

then we get zero on both sides and this is not useful for solving p in the inverse problem of PBNs with gene perturbation.

To reduce the computational cost and use the information of equations (III.1), the best number of equations to be added together is 2^{n-1}. Because if the number $m > 2^{n-1}$, we only use the information of the rest $2^n - m (< 2^{n-1})$ equations of (III.1) by Property III.3. We denote $F_1(p) = 0$ as sum 2^{n-1} equations of III.1. The summation of the rest 2^{n-1} equations is equivalent to $F_1(p) = 0$. So, and then, we sum $m = 2^{n-1}/2$ equations of the rest 2^{n-1} equations and denote it by $F_2(p) = 0$. We do this until there are only two equations and we denote them by $F_3(p) = 0$,...,$F_n(p) = 0$, $F_{n+1}(p) = 0$. Every nonlinear equation could be solved by the Newton's method. We know that if the initial iterate point is near the true solution, the Newton's method is convergent and it converges quadratically [16]. But it is very difficult to know the approximation of p. So we choose $0,\ 0.1,\ 0.2,\ 0.3, \ldots, 0.9$ as initial iterate point for every $F_i(p) = 0$, respectively. It is worth noting that we could randomly choose equations of equations (III.1) to get $F_i(p) = 0$. In the method we choose the equations by the order of the equation in (III.1) in general. This is the step of the modified Newton's method.

TABLE II
THE TRUTH TABLE OF THE PBN

State	$v_1 v_2 v_3$	$f_1^{(1)}$	$f_2^{(1)}$	$f_1^{(2)}$	$f_1^{(3)}$	$f_2^{(3)}$
1	(0,0,0)	0	0	0	1	0
2	(0,0,1)	1	0	1	1	0
3	(0,1,0)	1	0	0	0	1
4	(0,1,1)	1	1	1	1	0
5	(1,0,0)	0	1	1	0	0
6	(1,0,1)	0	1	1	0	1
7	(1,1,0)	0	1	0	0	1
8	(1,1,1)	1	1	0	1	0
	$c_j^{(i)}$	0.25	0.75	1	0.6	0.4

Algorithm (The Modified Newton's Method)

MNewton(A, $\tilde{x}$, ε)
% A is the state transition probability matrix without gene perturbations;
% $\tilde{x}$ is the steady-state probability distributions of PBNs with gene perturbations;
% ε is the termination tolerance of Newton method;
$m = 0$;
$F_{n+1}(p) = g_{2^n}(p)$; % n is gene number;
tag=1;m=0;i=1;
% Solve $F_{n+1}(p) = 0$ using different initial points
by Newton's method.
Let $p = Newton(F_{n+1}(p) = 0, \varepsilon)$
While tag==1 and $i <= n$
$F_i(p) = \sum_{m+1}^{m+2^{n-i}} g_i(p)$;
$m = m + 2^{n-i}$;
$q = Newton(F_i(p) = 0, \varepsilon)$;
If p and q have no community values
tag=0;
Return(The solution of $F_i(p) = 0$ and p have no community value!)
Stop;
End if
Replace p with community values of p and q.
$i = i + 1$;
End while
If tag==0
Print(The inverse have no solution!)
else
Test the points of p is whether the solution of equations (III.1) or not.
Print the result.
End if
END ALGORITHM

If the inverse problem have solutions, the algorithm iterates n times and could get the solutions. If the inverse problem have no solution, the iterate numbers must be smaller than n. We test the points of p is whether the solution of equations (III.1) or not in the last 'If' estimation since the solutions of $F_i(p) = 0$ may be not the solution of equations (III.1).

IV. NUMERICAL EXAMPLES

In this section we demonstrate the efficiency of our proposed algorithm with some numerical examples. We first give a numerical test by the proposed algorithm in [18], [19] and then consider a example of a 3-genes network given by Shmulevich et al. [12]. Finally, we generate a random 128×128 sparse nonnegative matrix as the transition probability matrix of a PBN to show the effectiveness of the modified Newton's method. We compare the modified Newton's method with the direct method. All the runs were done in Matlab 2008a on a CPU 2.1GHZ and 1.7GB memory computer, and the termination tolerance $\varepsilon = 1.0e-12$.

In the first numerical example (taken from [19]), we consider a PBN with three genes. The truth table of the Boolean functions are given in Table II. The state transition probability matrix is then given by

$$A = \begin{pmatrix} 0.40 & 0 & 0.45 & 0 & 0 & 0 & 0.15 & 0 \\ 0.60 & 0 & 0.30 & 0 & 0 & 0 & 0.10 & 0 \\ 0 & 0.30 & 0 & 0 & 0.25 & 0.15 & 0 & 0 \\ 0 & 0.45 & 0 & 0 & 0 & 0.10 & 0 & 0 \\ 0 & 0 & 0.15 & 0 & 0 & 0 & 0.45 & 0.40 \\ 0 & 0 & 0.10 & 0 & 0 & 0 & 0.30 & 0.60 \\ 0 & 0.10 & 0 & 0.40 & 0.75 & 0.45 & 0 & 0 \\ 0 & 0.15 & 0 & 0.60 & 0 & 0.30 & 0 & 0 \end{pmatrix}.$$

Suppose the perturbation probability $p = 0.015$, we apply

978-1-4577-1666-9/11/$26.00 ©2011 IEEE

TABLE VIII
COMPARE MODIFIED NEWTON'S METHOD WITH DIRECT METHOD

Method			Modified Newton's		Direct Method	
n	size	True	CPU	p^M	CPU	p^D
4	16×16	0.032	1.7031	0.0320	5.7656	0.0320
5	32×32	0.450	2.4375	0.4500	12.9531	0.4500
6	64×64	0.040	3.2812	0.0400	37.9687	0.0400
7	128×128	0.130	3.5313	0.1300	80.2656	0.1300
8	256×256	0.050	5.6875	0.0500	227.6094	0.0500

the modified Newton's method to get p. The numerical solution p_{ij} of $F_i(p)$ by the Newton's method are given in Table IV, and the iterate numbers of every equation by the Newton's method are given Table IV. Table V gives the solutions of every equations in equations III.1 by the Newton's method directly. From the Table IV, we know that the numerical solution by the modified Newton's method is 0.0150. Comparing Table IV with Table V, we know that the modified Newton's method is effectiveness.

In the second example, we consider a example of a 3-genes network proposed by Shmulevich et al.[12]. The truth table and the state transition probability matrix of the PBN are also given above. Suppose the perturbation probability $p = 0.021$, we use the modified Newton's method to get p.

The numerical solution p_{ij} of every $F_i(p) = 0$ obtained by using the modified Newton's method are given in Table VI. From Table VI, the numerical solution of the inverse problem of PBNs with gene perturbation is 0.0210 and it recovers p.

In the third example, we generate a random 128×128 sparse nonnegative matrix as the transition probability matrix of a 7 genes PBN, in this example. The perturbation probability $p = 0.05$, the steady-state distribution of the PBN with gene perturbation can be obtained by the power method. We apply the modified Newton's method to get p pretending that p are not known. The numerical solutions of every $F_i(p) = 0$ using different initial iterate points by modified Newton's method are given in Table VII. The CPU time of the modified Newton's method is 3.8906 seconds. By using the modified Newton's method, the gene perturbation probability obtained is 0.0500.

Finally, we generate several random $2^n \times 2^n$ sparse nonnegative matrices as the transition probability matrices of n genes PBNs and solve the inverse problem of PBNs with gene perturbation by the modified Newton's method or the direct method, in order to compare numerically the feasibility and effectiveness of these two methods in the sense of elapsed CPU time in seconds (denoted as "CPU"). Table VIII gives the numerical solutions and CPU time by two methods. We denote "True" as the true gene perturbation probability, "p^M" as the numerical solutions obtained by the modified Newton's method, and "p^D" as the numerical solutions obtained by the direct method.

V. CONCLUDING REMARKS

In this paper we consider the inverse problem of PBNs with gene perturbation and give an effect method for solving the perturbation probability p. Numerical examples show that our modified Newton's method is efficient and effective.

ACKNOWLEDGMENT

Research support in part by HKU strategic theme grant on computational sciences, National Natural Science Foundation of China Grant No. 10971075 and Guangdong Provincial Natural Science Grant No. 9151063101000021, Research Fund for the Doctoral Program of Higher Education of China Grant No. 20104407110001.

REFERENCES

[1] T. Akutsu, S. Miyano, and S. Kuhara, *Identification of genetic networks from a small number of gene expression patterns under the Boolean network model*, Pac. Symp. Biocomput., 4 (1999), 17-28.

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

[3] J. E. Celis, M. KruhØfer, I. Gromova, C. Frederiksen, M. Ostergaard, T. Thykjaer, P. Gromov, J. Yu, H. Plsdttir, N. Magnusson, and T. F. Orntoft, *Gene expression profiling: monitoring transcription and translation products using DNA microarrays and proteomics*, FEBS Lett, 480 (2000), 2-16.

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

[5] W. Ching, M. Ng, E. Fung, and T. Akutsu, *On construction of stochastic genetic networks based on gene expression sequences*, Int. J. Neu. Sys., 15 (2005), 297-310.

[6] W. Ching, S. Zhang, M. Ng, and T. Akutsu, *An approximation method for solving the steady-state probability distribution of probabilistic Boolean networks*, Bioinformatics, 12 (2007), 1511-1518.

[7] T. R. Hughes, M. Mao, A. R. Jones, J. Burchard, M. J. Marton, K. W. Shannon, S. M. Lefkowitz, M. Ziman, J. M. Schelter, M. R. Meyer, S. Kobayashi, C. Davis, H. Dai, Y. D. He, S. B. Stephaniants, G. Cavet, W. L. Walker, A. West, E. Coffey, D. D. Shoemaker, R. Stoughton, A. P. Blanchard, S. H. Friend, P. S. Linsley, *Expression profiling using microarrays fabricated by an ink-jet oligonucleotide synthesizer*, Nat. Biotechnol., 19 (2001), 342-347.

[8] S. Kauffman, *Metabolic stability and epigenesis in randomly constructed genetic nets*, J. Theor. Biol., 22 (1969), 437-467.

[9] S. Kauffman, *The origins of order: self-organization and selection in evolution*, Oxford University Press, Oxford, 1993.

[10] K. Murphy, S. Mian, *Modelling gene expression data using dynamic Bayesian networks*, Technical Report, Berkeley, 1999.

[11] R. Pal, I. Ivanov, A. Datta, M. Bittner, E. Dougherty, *Generating Boolean networks with a prescribed attractor structure*, Bioinformatics, 21 (2005), 4021-4025.

[12] I. Shmulevich, E. Dougherty, S. Kim and W. Zhang, *Probabilistic Boolean networks: a rule-based uncertainty model for gene regulatory networks*, Bioinformatics, 18 (2002) 261-274.

[13] I. Shmulevich, E. R. Dougherty, and W. Zhang, *Gene perturbation and intervention in probabilistic Boolean networks*, Bioinformatics, 18 (2002), 1319-1331.

[14] I. Shmulevich, *Probabilistic boolean networks : the modeling and control of gene regulatory networks*, SIAM, Philadelphia, 2010.

[15] P. Smolen, D. Baxter, and J. Byrne, *Mathematical modeling of gene network*, Neuron. 26 (2000), 567-580.

[16] C. Kelley, *Iterative methods for linear and nonlinear equation*, SIAM, Philadelphia, 1995.

[17] W. Xu, W. Ching, S. Zhang, W. Li. *Matrix perturbation method for computing the steady-state probability distributions of probabilistic Boolean networks with gene perturbations*, Journal of Computational and Applied Mathematics, 235 (2011) 2242 - 2251.

[18] S. Zhang, W. Ching, M. Ng, and T. Akutsu, *Simulation study in probabilistic Boolean network models for genetic regulatory networks*, International Journal of Data Mining and Bioinformatics, 1 (2007), 217-240.

[19] S. Zhang, W. Ching, X. Chen, N. Tsing, *Generating probabilistic Boolean networks from a prescribed stationary distribution*, Information Sciences, 180 (2010) 2560-2570.

978-1-4577-1666-9/11/$26.00 ©2011 IEEE

TABLE III
THE SOLUTION OF EQUATION $F_i(p)$ BY NEWTON'S METHOD.

Initial	0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9
$F_1(p)$	0.0150	0.0150	0.0150	0.0150	0.0150	0.0150	0.0150	0.0150	0.0150	0.0150
$F_2(p)$	0.0150	0.0150	0.0150	0.0150	0.0150	0.0150	0.0150	0.0150	0.0150	0.0150
$F_3(p)$	0.0150	0.0150	0.0150	0.0150	0.0150	0.0150	0.0150	0.0150	0.0150	0.0150
$F_4(p)$	0.0150	0.0150	0.0150	0.0150	0.0150	0.0150	0.0150	0.0150	0.0150	0.0150

TABLE IV
THE ITERATIVE NUMBER OF EQUATION $F_i(p)$ BY NEWTON'S METHOD.

Initial	0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9
$F_1(p)$	4	4	4	4	4	4	4	5	5	5
$F_2(p)$	5	7	7	8	8	9	9	9	9	9
$F_3(p)$	4	5	6	6	7	7	9	11	17	14
$F_4(p)$	5	7	7	8	8	9	9	9	9	9

TABLE V
THE SOLUTION OF EVERY EQUATION IN EQUATIONS III.1 BY NEWTON'S METHOD

Initial	0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9
$g_1(p)$	0.0150	0.0670	0.0670	0.0670	0.0150	0.6368	0.6368	0.6368	0.6368	0.6368
$g_2(p)$	0.0150	0.0150	0.0150	0.4584	0.4584	0.4584	0.4584	0.4584	0.4584	0.4584
$g_3(p)$	0.0150	0.0150	0.0150	0.0150	0.0150	0.0150	0.0150	0.0150	0.0150	0.0150
$g_4(p)$	0.0150	0.0150	0.0150	0.0150	0.0150	0.0150	0.0150	0.0150	-1.1572	-1.1572
$g_5(p)$	0.0150	0.1024	0.1024	0.1024	0.1024	0.1024	0.1024	0.1024	0.1024	0.1024
$g_6(p)$	0.0150	0.0150	0.0150	0.0150	0.0150	0.0150	0.0150	0.0150	0.0150	0.0150
$g_7(p)$	0.0150	0.0150	0.0150	0.0150	0.0150	0.0150	0.0150	0.0150	0.0150	0.0150
$g_8(p)$	0.0150	0.0150	0.0150	0.0150	0.0150	0.0150	0.0150	0.0150	0.0150	0.0150

TABLE VI
THE SOLUTION p_{ij} OF $F_i(p) = 0$ OF THE SECOND EXAMPLE IN SECTION IV

Initial	0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9
$F_1(p)$	0.0210	0.0210	0.0210	0.0210	0.0210	0.0210	0.0210	0.0210	0.0210	0.0210
$F_2(p)$	0.0210	0.0210	0.0210	0.0210	0.0210	0.9059	0.9059	0.9059	0.9059	0.9059
$F_3(p)$	0.0210	0.0210	0.0210	1.5370	0.5569	0.5569	0.5569	0.5569	0.5569	0.5569
$F_4(p)$	0.0210	0.0210	0.0210	0.0210	0.0210	0.9059	0.9059	0.9059	0.9059	0.9059

TABLE VII
THE SOLUTION p_{ij} OF $F_i(p) = 0$ OF THE THIRD EXAMPLE IN SECTION IV

Initial	0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9
$F_1(p)$	0.0500	0.0500	0.0500	0.0500	0.0500	0.0500	0.0500	0.0500	0.0500	0.0500
$F_2(p)$	0.0500	0.0500	0.0500	0.0500	0.0500	0.0500	0.0500	0.0500	0.0500	0.0500
$F_3(p)$	0.0500	0.0500	0.0500	0.0500	0.0500	0.0500	0.0500	2.7004	1.2801	1.2801
$F_4(p)$	0.0500	0.0500	0.0500	0.0500	0.0500	0.0500	0.0500	0.0500	1.0887	1.0887
$F_5(p)$	0.0500	0.0500	0.0500	0.0500	0.0500	0.0500	3.7374	1.0934	1.0934	1.0934
$F_6(p)$	0.0500	0.0500	0.0500	0.0500	0.0500	0.0500	0.9899	0.9899	0.9899	0.9899
$F_7(p)$	0.0500	0.0500	0.0500	0.0500	0.0500	0.0500	0.0500	0.0500	0.0500	1.0871
$F_8(p)$	0.0500	0.0500	0.0500	0.0500	0.0500	0.0500	0.9899	0.9899	0.9899	0.9899

978-1-4577-1666-9/11/$26.00 ©2011 IEEE