ALGORITHMS FOR MOLECULAR BIOLOGY

RESEARCH Open Access

Using the message passing algorithm on discrete data to detect faults in boolean regulatory networks

Anwoy Kumar Mohanty[1], Aniruddha Datta[1*] and Vijayanagaram Venkatraj[2]

Abstract

Background: An important problem in systems biology is to model gene regulatory networks which can then be utilized to develop novel therapeutic methods for cancer treatment. Knowledge about which proteins/genes are dysregulated in a regulatory network, such as in the Mitogen Activated Protein Kinase (MAPK) Network, can be used not only to decide upon which therapy to use for a particular case of cancer, but also help in discovering effective targets for new drugs.

Results: In this work we demonstrate how one can start from a model signal transduction network derived from prior knowledge, and infer from gene expression data the probable locations of dysregulations in the network. Our model is based on Boolean networks, and the inference problem is solved using a version of the message passing algorithm. We have done simulation experiments on synthetic data to verify the efficacy of the algorithm as compared to the results from the much more computationally intensive Markov Chain Monte-Carlo methods. We also applied the model to analyze data collected from fibroblasts, thereby demonstrating how this model can be used on real world data.

Keywords: Message passing, Sum-product, Markov chain Monte-Carlo

Background

Modeling cellular behavior is a first step towards the holistic understanding of the multivariate interactions among various genes. One possible approach to do that is through gene regulatory networks. These networks could also help in developing better intervention strategies in order to shift the state of the cell or the tissue to a more favorable one. Many different approaches have been proposed in the literature for modeling the behavior of genetic regulatory networks. These include differential equations [1], deterministic and probabilistic Boolean networks [2,3], and Bayesian and dynamic Bayesian networks [4,5]. Some of these methods rely on the assumption that the transition probabilities are provided beforehand. Such an assumption may not be realistic since the sheer volume of data required to effectively estimate the transition probabilities makes it a practically difficult proposition. Some methods such as the REVEAL algorithm [6] provide approaches to learn deterministic Boolean networks from discretized time course data. However time course data from biological samples itself can be difficult to come by.

One way to get around the problem of insufficient data is to use prior knowledge about the regulatory interactions between the various biological molecules in a cell. In the biological literature, a lot of information is available regarding the various regulatory interactions. This information has been collected by biologists over a long period of time. These regulatory interactions, collectively referred to as pathway knowledge, are generally not incorporated into the various methods of modeling gene regulatory networks. Using this information, however, would result in models which describe cellular behavior more accurately.

A possible approach to use such prior information has been developed in [7]. In that reference, the authors use Boolean logic to model signal transduction networks. In [8], the authors have used boolean models derived from

*Correspondence: datta@ece.tamu.edu
[1]Department of Electrical and Computer Engineering, Texas A&M University, 77843 College Station, USA
Full list of author information is available at the end of the article

BioMed Central

© 2014 Mohanty et al.; licensee BioMed Central Ltd. This is an Open Access article distributed under the terms of the Creative Commons Attribution License (http://creativecommons.org/licenses/by/4.0), which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly credited. The Creative Commons Public Domain Dedication waiver (http://creativecommons.org/publicdomain/zero/1.0/) applies to the data made available in this article, unless otherwise stated.

prior information to model the heterogeneity of cancerous tissues. Furthermore, in [9] Boolean logic is used to model the Mitogen Activated Protein Kinase (MAPK) signal transduction network and the result of that modeling is shown in Figure 1. Here, each connecting wire corresponds to a variable which represents the state of the corresponding protein/gene. In this model each variable is assumed to have two states, an activated and a deactivated one. For example the state of EGFR will be upregulated or activated when the cell is exposed to EGF. The way the various variables are dependent on each other can be modeled using standard Boolean logic functions such as AND, OR, NOT, NAND, etc. This is shown in Figure 1. In [9] the authors presented a stuck-at fault model of the mutations which result in the neoplastic behavior of the tissue. A stuck-at-one fault corresponds to a variable permanently being in an activated state irrespective of the states of the variables upstream of it. Similarly a stuck-at-zero fault would mean a variable has a permanently downregulated state irrespective of the states of the other upstream variables. These "stuck" variables would however affect the variables downstream of them through the Boolean Logic gates which have these variables as inputs. To show how Boolean regulatory networks with stuck-at faults can be used to model cancerous tissue, we give the following examples. In 30% of human breast cancers there is an over expression of the ERBB2 gene [10]. This causes ligand independent firing translating to a stuck-at-one fault in the Boolean network. A stuck-at one fault at ERBB2 means that the variable corresponding to ERBB2 in the Boolean network shown in Figure 1 is always upregulated regardless of the activity status of the variables upstream of it. Similarly in 90% of the pancreatic cancer cases we see a mutated Ras gene which causes it to lose its gtpase activity [10]. In other words, we have a stuck-at-one fault associated with the Ras variable. Stuck-at faults could also be interpreted as points of dysregulation in the Boolean network brought about by certain genes irrespective of the presence of mutations.

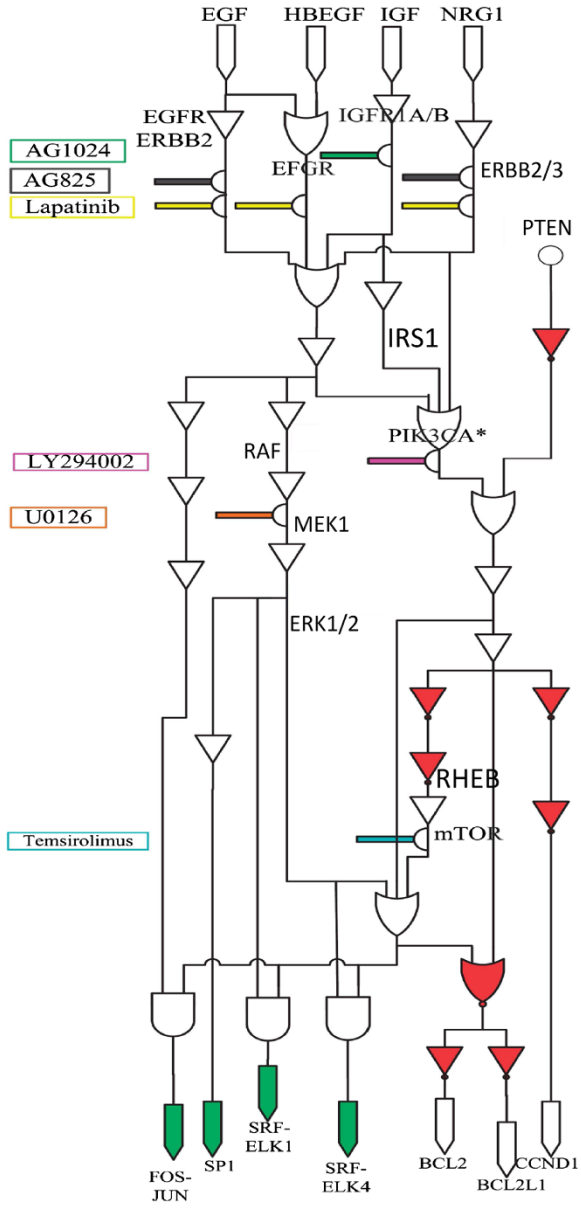

Figure 1 A Boolean network model of the MAPK signal transduction network with target locations of inhibitory drugs shown.

Figure 1 also shows the locations where some important kinase inhibitory drugs act. The inputs in the Boolean network corresponding to these drugs are '1' when the drugs are administered, '0' otherwise. When the input corresponding to a drug is '1', it causes the target variable to get downregulated irrespective of the variables upstream of it, which in turn affects other downstream variables. At the bottom of the network, various observable variables such as the transcription factors (FOS-JUN, SP1, SRF-ELK) are represented as arrows. Other proteins of interest (BCL2, BCL2L1, CCND1) are also represented as arrows. Activity of transcription factors can be determined by the use of appropriate reporter genes. Hence this network can be considered as a multi input multi output system with the inputs being the exposure to various drug combinations and the output being the activity of the observable variables. The growth factors can also be considered as inputs, but in this paper we are simply considering the drugs as inputs whereas the growth factors are all considered to be one, that is, constantly active.

Locating stuck-at faults in a given Boolean regulatory network could help in the identification of key dysregulated genes that have a strong impact on the observable variables. This in turn could be used to identify targets for

new drugs. Knowledge about the locations of the stuck-at faults along with knowledge about the targets of the kinase inhibitory drugs can be used to come up with optimal intervention strategies. A method to devise optimal intervention strategies using such Boolean regulatory networks with stuck-at faults is described in [9]. Accordingly, the problem we pose is this: given data points, where each data point consists of a combination of drugs used as the input and the activity of the observable variables as outputs, is it possible to locate the variables where stuck-at-faults have occurred? In the following sections we represent the problem as a statistical model with unknown parameters which are estimated from the data points using the message passing algorithm. This algorithm allows for rapid computation of the posterior probabilities of the parameters. The estimates obtained are evaluated by comparison with the results given by Markov Chain Monte Carlo methods.

Model description

There are many ways to model a gene regulatory network which describes the behavior of neoplastic tissue. The general rule is that the more the number of unknown parameters, the more the amount of data that is required to get an effective estimate of those parameters. Hence the modeling must be done keeping in mind the limited amount of data available from biological experiments.

Before we go into the details, we would like to point out that literature survey would enable us to know the most likely locations in the Boolean network where stuck-at faults can take place. As stated in the previous section, in 30% of human breast cancers there is an over expression of the ERBB2 gene, and in 90% of the pancreatic cancer cases we see a mutated Ras gene. These are among many examples where prior knowledge about locations of faults is available. This knowledge would allow us to limit the search space for faults in the network. For example we may provide a set of locations where we want to search for faults.

One important assumption made in the modeling of mutations is that they are random events that occur independently of each other [11-13]. We make use of this assumption in our model by assuming that the faults occur unconditionally independent from each other with certain unknown probabilities associated with them. These unknown probability parameters are to be estimated from the collected data. These estimated probabilities will indicate our confidence about where the faults have occurred in the Boolean regulatory network.

We now explain the key ideas through a simple example. Let us assume that we have narrowed down the set of locations where we want to search for faults to be composed of RAF, IRS1, and RHEB as shown in Figure 1 (we are assuming stuck-at-one faults). Let their probabilities of occurrences be ρ_1, ρ_2, and ρ_3 which are to be determined. Define $\rho = (\rho_1\ \rho_2\ \rho_3)^T$ as the vector of the three parameters. Three possible locations of faults implies that there are 2^3 different fault combinations and their associated networks corresponding to the binary numbers 000, 001, ..., 111. The first network is one with no faults and has a probability of

$$P(M = 0/\rho) = (1-\rho_1)(1-\rho_2)(1-\rho_3). \quad (1)$$

The second network has a single stuck-at-one fault at RHEB alone, and it's probability is given by $P(M = 1/\rho) = (1-\rho_1)(1-\rho_2)\rho_3$. Similarly, the third network has a single stuck-at-one fault at IRS1 alone, with a probability of $P(M = 2/\rho) = (1-\rho_1)\rho_2(1-\rho_3)$, and so on. The variable M is the decimal equivalent of the binary number representing the different fault combinations and could equivalently represent the particular faulty Boolean network being considered. Since there are three possible locations where stuck-at-faults can take place in this example, M can take $2^3 = 8$ different values. In our convention, we use integers from 0 to $2^3 - 1$ to represent the values taken by M. For example $M = 6$ corresponds to a network with faults at RAF and IRS1 but not at RHEB and has a corresponding probability of $\rho_1\rho_2(1-\rho_3)$.

In this example the dimension of ρ is three, but it can be any integer depending on the size of the search space. Determining the entries of ρ allows us to determine the most likely faulty networks. Let V be the dimension of ρ. Then it is clear that $P(M = m/\rho)$ has the following form:

$$P(M = m/\rho) = \prod_{v=1}^{V} \rho_v^{R_{v,m}} (1-\rho_v)^{1-R_{v,m}} \quad (2)$$

where $R_{v,m}$ is either 0 or 1 and m can vary from 0 to $2^V - 1$.

Consider any one of the variables represented as arrows at the bottom of Figure 1. Let us represent that variable by O_j. j varies from 1 to 7 in our example based on Figure 1. The behavior or O_j is determined by the network and what faults are in it. Let $o_{i,j}$ be an observation of that variable when the combination input is I_i. $o_{i,j}$ can be either 0 or 1 since we are dealing with a boolean network here. Given that the network M is any one of the 2^V possible networks and given that the drug combination input is I_i, the probability $P\left(O_j = o_{i,j}/M = m, I_i\right)$ can be either 0 or 1. It is 1 when $o_{i,j}$ matches the output of the j^{th} output variable of the m^{th} network for the the input drug combination I_i, and is 0 otherwise. Let us represent $P\left(O_j = o_{i,j}/M = m, I_i\right)$ by $S_{m,i,j}$. The probability $P(M = m/\rho)$ is a function of ρ as

described in equation (2). Therefore, by the theorem of total probability,

$$P\left(O_j = o_{i,j}/I_i, \rho\right) = \sum_{m=0}^{2^V-1} S_{m,i,j} P\left(M = m/\rho\right) \tag{3}$$

In our example, we will proceed by assuming that the observable variables (the O_j's) are independent given the faulty network and the drug combinations. This assumption can be easily relaxed for the case when the 7 observable variables represented as arrows at the bottom of Figure 1 are observed together for each drug combination used as the input. In this case, instead of $P\left(O_j/M\right)$, we will be working with $P\left(O_1, O_2, \ldots, O_7/M\right)$. This however does not affect our fundamental results and is a simple extension of our example.

Let O represent all of the observed data for all the observable variables and I represent the entire set of the corresponding inputs. Let J be the number of observable variables and N be the number of observations for each observable variable. Then we have

$$P\left(O/\rho, I\right) = \prod_{j=1}^{J}\prod_{i=1}^{N} P\left(O_j = o_{i,j}/I_i, \rho\right) \tag{4}$$

which is nothing but the likelihood function. In order to handle experimental repeats, we can have the the drug combinations I_i to be the same for more than one value of the index i.

An estimate of ρ can be obtained from equation 4, either by maximum likelihood estimation, or by calculating the posterior mean of the parameters. If the prior distributions of all the elements of ρ are assumed to be uniformly distributed between 0 and 1, the posterior distribution of ρ is directly proportional to $P\left(O/\rho, I\right)$. If $P(O/\rho, I)$ comes out to be zero for all values of ρ, then we have every reason to question the validity of the Boolean network used to model the behavior of the biological network, or the set of possible locations of faults. Various estimates of ρ, such as the posterior mean or the posterior mode (the value of ρ where the posterior distribution is maximal) can be obtained from $P\left(O/\rho, I\right)$. Now we can algebraically expand the right hand side of equation (4) to write $P\left(O/\rho, I\right)$ as

$$P\left(O/\rho, I\right) = \sum_{k}\prod_{v=1}^{V} \rho_v^{Q1_{v,k}} \left(1-\rho_v\right)^{Q2_{v,k}} \tag{5}$$

where $Q1_{v,k}$ and $Q2_{v,k}$ are non negative integers. Calculating $P\left(\rho/O, I\right)$ from $P\left(O/\rho, I\right)$ is now trivial since it only involves calculation of a multiplicative normalization constant.

$$P\left(\rho/O, I\right) = \frac{P\left(O/\rho, I\right)}{\int P\left(O/\rho, I\right) d\rho} \tag{6}$$

where in the denominator there is the normalization constant which turns out to be

$$\int P\left(O/\rho, I\right) d\rho = \sum_{k}\prod_{v=1}^{V} \beta\left(Q1_{v,k}+1, Q2_{v,k}+1\right) \tag{7}$$

where $\beta(*,*)$ is the beta function. This equation is derived by considering a uniform prior on all the elements of ρ. The integrations can be done easily because of the form of equation 5. Each variable ρ_v is integrated from 0 to 1. Equation 6 shows the joint posterior distribution of all the unknown parameters ρ_1 through ρ_V considered together. In order to find the marginal distribution of any given parameter of interest, we will need to integrate out the rest of the parameters. For example $P(\rho_l/O, I)$ for any given value of l can be found out to be

$$P(\rho_l/O, I) = \frac{\sum_k \rho_l^{Q1_{l,k}} (1-\rho_l)^{Q2_{l,k}} \prod_{\substack{v=1 \\ v\neq l}}^{V} \beta\left(Q1_{v,k}+1, Q2_{v,k}+1\right)}{\sum_k \prod_{v=1}^{V} \beta\left(Q1_{v,k}+1, Q2_{v,k}+1\right)} \tag{8}$$

Following this the posterior means can also be calculated.

However the number of additive terms in equation 5 represented by the summing variable k in general rises exponentially with the number of data points collected. In the worst case, the left hand side of equation (3) will contain 2^V terms. Since the number of multiplicative terms in equation 4 is NJ (the number of data points collected), upon expanding the right hand side of equation 4 we get 2^{VNJ} additive terms in equations 5, 7, and 8. Thus the computational cost to compute the mean of any given ρ_l is $O(2^{VNJ})$. Hence the total computation cost to compute the posterior means of all the elements of ρ (ρ_1 through ρ_V) is $O\left(V \times 2^{VJN}\right)$. Therefore the straightforward approach for calculating the posterior distributions of ρ_l's and their posterior means will get intractable as the amount of data collected increases.

To get around this difficulty we will use an iterative algorithm to obtain an approximation of the marginal distributions of the elements of the parameter vector ρ. From the marginal distribution it will be straightforward to obtain the posterior means and confidence intervals of the individual elements of ρ.

Factor graph representation of the model

Factor Graphs are an important tool used in various applications such as signal processing and telecommunications. Many algorithms can be easily understood and derived using the factor graph approach. These include Kalman Filters, the Viterbi Algorithm, the Forward-Backward algorithm and Turbo Codes to name a few. The approach involves first representing the probability

model as a factor graph and then applying the message passing algorithm along the edges. The reader is referred to [14,15] for an in-depth coverage of factor graphs and the message passing algorithm. Here we provide a short primer to the subjects and go into the details of only our particular example.

A simple example

Consider a simple function $g(x_1, x_2, x_3) = f_1(x_1, x_2) \times f_2(x_2, x_3) f_3(x_3)$, where x_i are discrete variables. Suppose we want to calculate $\sum_{x_1,x_3} g(x_1, x_2, x_3)$ for a particular value of x_2 (the marginal of x_2). In addition, suppose that each x_i can take A different values. Hence the straight forward approach would require us to sum $g(x_1, x_2, x_3)$ over A^2 different values. However $\sum_{x_1,x_3} g(x_1, x_2, x_3)$ can also be calculated as

$$\sum_{x_1,x_3} g(x_1, x_2, x_3) = \left(\sum_{x_1} f_1(x_1, x_2)\right)\left(\sum_{x_3} f_2(x_2, x_3) f_3(x_3)\right) \tag{9}$$

which sums over $2A$ different values. For continuous variables, the summation is replaced by integration. The optimal strategy for calculating the marginal of x_2 is straightforward to derive in this simple example. However a systematic approach to find the optimal strategy to calculate the marginal of any variable for any given probability function is given by the message passing algorithm which acts on the factor graph representation of the function.

The factorization of a function can be represented by a factor graph. A factor graph is a bipartite graph with a variable node corresponding to each variable x_i and a factor node corresponding to each independent factor f_j and has an undirected edge connecting a variable node of x_i to a factor node of f_j iff x_i is an argument of f_j [14,15]. The factor graph of $g(x_1, x_2, x_3)$ is shown in Figure 2.

Messages pass along the edges in both directions. Messages are functions of the variable whose node is associated with the edge. Let $\mu_{f_j \to x_i}(x_i)$ and $\mu_{x_i \to f_j}(x_i)$ denote the messages from f_j to x_i and vice versa. We simply write down the update equations below. For an in-depth discussion on their derivation, the reader is referred to [14,15]. The messages are calculated as follows:

$$\mu_{x_i \to f_j}(x_i) = \prod_{h \in n(x_i) \setminus \{f_j\}} \mu_{h \to x_i}(x_i) \tag{10}$$

$$\mu_{f_j \to x_i}(x_i) = \sum_{\sim\{x_i\}} \left(f_j(X) \prod_{y \in n(f_j) \setminus \{x_i\}} \mu_{y \to f_j}(y) \right) \tag{11}$$

where $n(x_i)$ and $n(f_j)$ denote the neighbors of x_i and f_j respectively in the factor graph. $n(x_i) \setminus \{f_j\}$ represents the set of all the neighbors of x_i except f_j. The definition of $n(f_j) \setminus \{x_i\}$ is similar. Since the factor graph is bipartite, the neighbors of a variable node can only be factor nodes, and the neighbors of a factor node can only be variable nodes. X denotes the set of arguments of f_j. $\sum_{\sim\{x_i\}}$ denotes summation over all local variables except x_i. The set of local variables will simply be the set X, since the factor node f_j is connected by undirected edges only to the variable nodes of its arguments. The message going away from a leaf variable node is the constant 1, while the message going away from a leaf factor node is the value of that local factor. Using these rules on the simple example, we have $\mu_{x_1 \to f_1}(x_1) = 1$ and $\mu_{f_3 \to x_3}(x_3) = f_3(x_3)$.

$$\begin{aligned} \mu_{f_1 \to x_2}(x_2) &= \sum_{\sim\{x_2\}} f_1(x_1, x_2)\, \mu_{x_1 \to f_1}(x_1) \\ &= \sum_{x_1} f_1(x_1, x_2) \end{aligned} \tag{12}$$

$$\begin{aligned} \mu_{x_3 \to f_2}(x_3) &= \prod_{h \in n(x_3) \setminus \{f_2\}} \mu_{h \to x_3}(x_3) = \mu_{f_3 \to x_3}(x_3) \\ &= f_3(x_3) \end{aligned} \tag{13}$$

$$\begin{aligned} \mu_{f_2 \to x_2}(x_2) &= \sum_{\sim\{x_2\}} f_2(x_2, x_3)\, \mu_{x_3 \to f_2}(x_3) \\ &= \sum_{x_3} f_2(x_2, x_3) f_3(x_3) \end{aligned} \tag{14}$$

The marginal distribution of a variable is simply the product of all the messages being received by the corresponding variable node. Hence $\sum_{x_1,x_3} g(x_1, x_2, x_3) = \mu_{f_1 \to x_2}(x_2) \times \mu_{f_2 \to x_2}(x_2)$ and thus equation (9) is derived using factor graphs and the message passing algorithm. Calculating the rest of the messages would allow us to calculate the marginals of x_1 and x_3 as well. The message passing algorithm would terminate when messages along both directions of all the edges in the graph have been calculated.

The message passing algorithm terminates and gives exact marginals for the cases where the factor graph has no cycles. But the most interesting applications are for those cases where the factor graph has cycles, where the

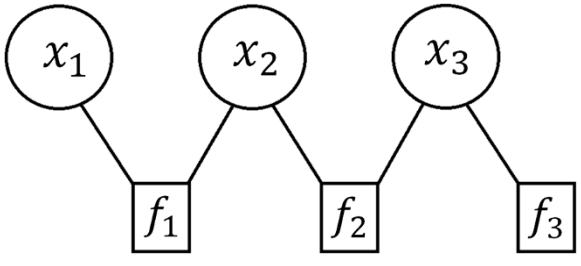

Figure 2 The factor graph representation of a factorizable function. The variable nodes are circular and the factor nodes are rectangular.

marginals are calculated by iteratively updating the messages (for example the iterative decoding of turbo codes). We similarly use an iterative version of the message passing algorithm in our model to approximate the marginal posterior distribution of the unknown parameters.

Using factor graphs and the message passing algorithm on the signal transduction network model

Now, $P(\rho/O,I) \propto P(O/\rho,I)$ as is evident from equation (6), while the expression for $P(O/\rho,I)$ is given in equation (4). Let $P_{i,j}$ represent the multiplicative factor $P\left(O_j = o_{i,j}/I_i,\rho\right)$ in equation (4). In a factor graph, each multiplicative factor is represented by a factor node and each element of ρ is represented by a variable node. Hence there are NJ number of factor nodes with each corresponding to one particular multiplicative term in equation 4, and there are V number of variable nodes with each corresponding to one particular unknown parameter (one out of ρ_1 through ρ_V). The purpose of this algorithm is to compute the posterior marginal distributions of the unknown parameters ρ_1 through ρ_V, which can then be used to compute their means and confidence intervals.

Figure 3 shows the factor graph of equation (4).

As we can see the factor graph in Figure 3 has cycles. In a factor graph with cycles, the message passing algorithm does not terminate and the messages are locally updated with every iteration. Every time a new message is calculated, it replaces the old message. The iterative message passing algorithm is as follows:

1. initialize all $\mu_{\rho_v \to P_{i,j}}(\rho_v) = 1$.
2. calculate all $\mu_{P_{i,j} \to \rho_v}(\rho_v)$ as per equation (11).
3. calculate all $\mu_{\rho_v \to P_{i,j}}(\rho_v)$ as per equation (10).
4. repeat steps 2 and 3 in that order.

Since we are dealing with continuous variables between 0 and 1, the summations are replaced by integrations. Every time $\mu_{P_{i,j} \to \rho_v}(\rho_v)$ are computed in step 2, they come out to be polynomials of degree one due to the multiplicatively separable nature of the integrands involved and that all the parameters ρ_v are being integrated from 0 to 1 (a rectangular integration region). Let them be represented as $b_{0,v,i,j} + b_{1,v,i,j} \times \rho_v$. Hence $\mu_{P_{i,j} \to \rho_v}(\rho_v)$ can be represented by a vector $b_{v,i,j} = \left(b_{0,v,i,j}\ b_{1,v,i,j} \right)^T$. Every time $\mu_{\rho_v \to P_{i,j}}(\rho_v)$ are computed in step 3, they will be polynomials of degree $NJ-1$ since they are simply the product of all incoming messages except one. Let them be represented as $\sum_{k=0}^{NJ-1} a_{k,v,i,j} \rho_v^k$. Hence $\mu_{\rho_v \to P_{i,j}}(\rho_v)$ can be represented by a vector $a_{v,i,j} = \left(a_{0,v,i,j}\ a_{1,v,i,j}\ \ldots\ a_{NJ-1,v,i,j} \right)^T$.

The values $b_{0,v,i,j}$ and $b_{1,v,i,j}$ can be updated in step 2 as follows.

$$b_{0,v,i,j} \leftarrow \sum_{m=0}^{2^V-1} S_{m,i,j}\left(1 - R_{v,m}\right) \times \prod_{\substack{l \in \{1 \ldots V\} \\ l \neq v}} \left(\sum_{k=0}^{NJ-1} \frac{a_{k,l,i,j}}{k+2} \right)^{R_{l,m}} \times \left(\sum_{k=0}^{NJ-1} \frac{a_{k,l,i,j}}{(k+1)(k+2)} \right)^{1-R_{l,m}} \tag{15}$$

$$b_{1,v,i,j} \leftarrow \sum_{m=0}^{2^V-1} S_{m,i,j}\left(2R_{v,m} - 1\right) \times \prod_{\substack{l \in \{1 \ldots V\} \\ l \neq v}} \left(\sum_{k=0}^{NJ-1} \frac{a_{k,l,i,j}}{k+2} \right)^{R_{l,m}} \times \left(\sum_{k=0}^{NJ-1} \frac{a_{k,l,i,j}}{(k+1)(k+2)} \right)^{1-R_{l,m}} \tag{16}$$

The $a_{v,i,j}$ can be updated in step 3 by performing polynomial multiplications of the $NJ-1$ incoming first degree polynomials to the $v'th$ variable node and comparing coefficients. That is, the following equation must be satisfied.

$$\sum_{k=0}^{NJ-1} a_{k,v,i,j} \rho_v^k = \prod_{g \neq i, h \neq j} \left(b_{0,v,g,h} + b_{1,v,g,h} \times \rho_v \right) \tag{17}$$

By comparing coefficients of either side of equation (17), the values of the elements of the vector $a_{v,i,j}$ are updated.

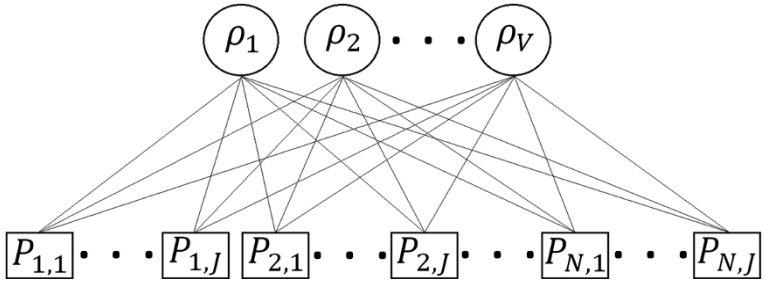

Figure 3 The factor graph representation of the probability model of the signal transduction network. The variable nodes are circular and the factor nodes are rectangular.

This is also equivalent to the convolution of the message vectors $b_{v,g,h}$ for $g \neq i, h \neq j$. At each iteration, the message vectors can be multiplied by constants so as to prevent overflow or underflow when implementing the algorithm on a digital computer with finite precision. In that case the final solutions we get are simply the required marginal distributions scaled by some unknown constant. If we are simply interested in the marginal distributions, then it is not necessary to keep track of the multiplied constants. We simply need to normalize the marginals so that their integrals from 0 to 1 give unity.

The message vectors $a_{v,i,j}$ and $b_{v,i,j}$ are iteratively updated until some convergence criteria is satisfied (for example if the Hellinger distance between the marginals of two successive iterations is below a certain threshold). In our simulations, we saw that as few as 2 iterations gave satisfactory results in terms of convergence. Hence the time complexity of the algorithm is dependent on steps 2 and 3 of the algorithm.

In order to calculate $\mu_{\rho_v \to P_{i,j}}(\rho_v)$ in step 3, first calculate the polynomial $U_v(\rho_v) = \prod_{g,h} \mu_{P_{g,h} \to \rho_v}(\rho_v)$ of degree NJ. Then find the quotient of the division operation $U_v(\rho_v) \div \mu_{P_{i,j} \to \rho_v}(\rho_v)$. This gives $\mu_{\rho_v \to P_{i,j}}(\rho_v)$. Along with that, we can also calculate and store the value of $\theta_{v,i,j,1} = \int_0^1 \rho_v \mu_{\rho_v \to P_{i,j}}(\rho_v) d\rho_v$ and $\theta_{v,i,j,0} = \int_0^1 (1-\rho_v) \mu_{\rho_v \to P_{i,j}}(\rho_v) d\rho_v$ which will be used in step 2. Note that $\theta_{v,i,j,1} = \sum_{k=0}^{NJ-1} \frac{a_{k,v,i,j}}{k+2}$ and $\theta_{v,i,j,0} = \sum_{k=0}^{NJ-1} \frac{a_{k,v,i,j}}{(k+1)(k+2)}$. Calculating the coefficients of $U_v(\rho_v)$ is of time complexity at most $O\left((NJ)^2\right)$. This is because it involves the convolution of NJ different first degree polynomials. Calculating the quotient of $U_v(\rho_v) \div \mu_{P_{i,j} \to \rho_v}$, and $\theta_{v,i,j,1}$ and $\theta_{v,i,j,0}$ are of time complexity $O(NJ)$. The last three operations of $O(NJ)$ have to be done for all NJ of the factor nodes for each variable node. Hence the time complexity of calculating the messages from one variable node to all factor nodes is of time complexity $O\left((NJ)^2\right)$. Repeating this action for all V variable nodes gives us the time complexity of step 3 of the algorithm to be $O\left((NJ)^2 V\right)$.

If we look at equations (15) and (16), the computation of $b_{v,i,j}$ seems to be of $O\left(NJV2^V\right)$ time complexity. Since there are NJV of $b_{v,i,j}$ to be computed, step 2 seems to be of $O\left((NJ)^2(V)^2 2^V\right)$ time complexity. However some of the computations are repeated and storing these computations for reuse can reduce the time complexity. Let $\kappa_{m,i,j} = \prod_{l=1}^{V} \theta_{l,i,j,1}^{R_{l,m}} \theta_{l,i,j,0}^{1-R_{l,m}}$. Then $\mu_{P_{i,j} \to \rho_v}(\rho_v) = \sum_{m=0}^{2^V-1} S_{m,i,j} \rho_v^{R_{v,m}} (1-\rho_v)^{1-R_{v,m}} \times \frac{\kappa_{m,i,j}}{\theta_{v,i,j,1}^{R_{v,m}} \theta_{v,i,j,0}^{1-R_{v,m}}}$. Computation of $\kappa_{m,i,j}$ for all m is of $O\left(V2^V\right)$ time complexity for a given factor node $P_{i,j}$. Computation of $\mu_{P_{i,j} \to \rho_v}(\rho_v)$ for all v is of $O\left(V2^V\right)$ time complexity for a given factor node $P_{i,j}$. Hence computation of $\mu_{P_{i,j} \to \rho_v}(\rho_v)$ from a single factor node to all variable nodes is of $O\left(V2^V\right)$ time complexity. Hence total computation for all factor nodes in step 2 comes out to be of $O\left(NJV2^V\right)$ time complexity.

Hence the complexity of each iteration of the algorithm comes out to be $O\left(NJV(2^V + CNJ)\right)$, where C is a constant. This is quadratic with respect to the number of data points NJ, as opposed to the exponential complexity of the straightforward approach discussed in section 'Model description'.

Once the convergence criteria is met and the algorithm is terminated, the marginal distribution of ρ_v is calculated as

$$P(\rho_v/O, I) = \gamma \prod_{i,j} \mu_{P_{i,j} \to \rho_v}(\rho_v) \tag{18}$$

where γ is a normalization constant which can be calculated to give $\int_0^1 P(\rho_v/O, I)\, d\rho_v = 1$.

Simulation experiments

We did simulations where the algorithm was tested on synthetic data as well as applied to real world data. The marginal posterior distributions estimated using the iterative message passing algorithm were compared with the marginal posteriors estimated using the time consuming and computationally intensive Markov Chain Monte Carlo (MCMC) methods and the estimates obtained using both methods came out to be close thereby verifying the iterative message passing algorithm's correctness.

Various literature on MCMC methods exist [16-18]. We will describe the details used in our simulations instead of going into a detailed discussion of MCMC methods. The Markov Chain Monte Carlo Method involves creating a Markov Chain whose stationary distribution is the required posterior distribution. The Metropolis-Hastings Algorithm will be used to generate such a Markov Chain since the samples need to be generated from a non standard probability distribution. This method will be used to generate samples from the posterior distribution of the unknown parameters of the vector ρ. These samples can then be used to get an estimate of the joint as well as the marginal posterior distributions of the unknown parameters using kernel density estimation.

Samples are drawn from the posterior distribution of ρ using the Metropolis-Hastings (MH) Algorithm in the following manner. Let the n^{th} sample drawn from the posterior distribution of ρ be $\rho^{(n)} = \left(\rho_1^{(n)} \ \rho_2^{(n)} \ \ldots \ \rho_V^{(n)} \right)$.

1. Initialize all elements of $\rho^{(0)}$ to be 0.5.
2. At the n^{th} iteration of the MH algorithm, generate ρ^* from the proposal distribution $U(\rho/\rho^{(n)}, \Delta)$. The proposal distribution and the tuning parameter Δ will be discussed in the next paragraph.

3. Calculate the acceptance ratio

$$D = \frac{P\left(O/\rho^*, I\right) U\left(\rho^{(n)}/\rho^*, \Delta\right)}{P\left(O/\rho^{(n)}, I\right) U\left(\rho^*/\rho^{(n)}, \Delta\right)}$$

(Recall that the prior of the parameter vector is constant). $P(O/\rho^*, I)$ and $P(O/\rho^{(n)}, I)$ can be easily calculated for known values of ρ^* and $\rho^{(n)}$ without the expansion of $P(O/\rho, I)$ described in equation (5). Accept ρ^* as the next sample $\rho^{(n+1)}$ with probability $min(1, D)$, or keep $\rho^{(n+1)}$ equal to $\rho^{(n)}$ with probability $1 - min(1, D)$.

4. Repeat steps 2 and 3 to generate samples from the posterior of $P(\rho/O, I)$.

The proposal distribution $U(\rho/\rho^{(n)}, \Delta)$ is such that ρ_i is Beta distributed with parameters $\frac{\rho_i^{(n)}}{\Delta}$ and $\frac{1-\rho_i^{(n)}}{\Delta}$, that is

$$U\left(\rho/\rho^{(n)}, \Delta\right) = \prod_{i=1}^{V} \frac{\rho_i^{\frac{\rho_i^{(n)}}{\Delta}-1}(1-\rho_i)^{\frac{1-\rho_i^{(n)}}{\Delta}-1}}{Beta\left(\frac{\rho_i^{(n)}}{\Delta}, \frac{1-\rho_i^{(n)}}{\Delta}\right)} \qquad (19)$$

where $Beta(x, y)$ is the beta function with parameters x and y and Δ is a scalar tuning parameter which controls the variance of the distributions of the ρ_i's. It can be adjusted to give autocorrelation properties of the Markov Chain within acceptable ranges.

Experiments with synthetic data

To demonstrate the working of the algorithm, we ran simulations of the message passing algorithm as well as the MH algorithm on synthetic data. We generated synthetic data from the example described in section 'Model description' which was derived from the MAPK signal transduction network, which is a well understood network.

The set of locations where faults can take place was taken to be composed of RAF, IRS1, and RHEB. The probabilities of stuck-at-one faults at these locations (The parameters ρ_1, ρ_2, and ρ_3) were taken as 0.7, 0.4, and 0.2. Synthetic observations of the observable variable (the variables shown at the bottom of Figure 1 as arrows) were generated for various drug combinations as inputs (the drugs being AG1024, AG825, Lapatinib, LY294002, U0126, and Temsirolimus, whose action on the Boolean network of the MAPK network is shown in Figure 1) according to the probability model described in the previous sections. The inputs at the top of the network corresponding to growth factors (EGF, HBEGF, IGF, and NRG1) were all taken as 1 (if the cells were being grown on petridishes, then this would be equivalent to the case where all the four growth factors have been supplied in the serum). Hence the data set $\left\{\left(o_{i,1}, o_{i,2}, \ldots, o_{i,J}\right), I_i\right\}$ is generated. There are 6 drugs in the Boolean model. All the $2^6 - 1$ drug combinations were used to generate the data points. Hence i varies from 1 to 63.

After the synthetic data set was generated, the marginal posterior distributions of the elements of ρ (The parameters ρ_1, ρ_2, and ρ_3) were estimated using both the message passing algorithm as well as the MCMC method. For the MCMC method, the tuning parameter Δ is set to 0.04 which gives an acceptance rate of 40%. The reader is referred to [18] for information on acceptance rates. Then the Markov Chain was run to generate 50,000 samples to attain stationarity (the burn in period). Following this, the Markov chain was run long enough to generate 250,000 samples and thinned by a factor of 50 (one in 50 samples generated was stored for each parameter) resulting in 5000 samples for each ρ_v. This resulted in effective sample sizes of atleast 4000 for each of the ρ_i's. the reader is referred to [18] for information on effective sample sizes. The algorithms were implemented in MATLAB. The message passing algorithm was terminated after 2 iterations which took about 4 seconds. For our purposes, we used the Hellinger Distance between the marginals of the first parameter ρ_1 calculated at consecutive iterations of the message passing algorithm to fall below a certain threshold to signal termination of the algorithm. However other convergence criterions could also be used. The MCMC samples were generated in 30 minutes after the initial burn in period. The marginal posterior distribution of ρ_1 through ρ_3 calculated using both the message passing algorithm and the MCMC approach are shown in Figure 4. Kernel density estimation with a Gaussian Kernel was used to estimate the marginals from the sample values generated using the MH algorithm. The estimate $\hat{P}(\rho_v/O, I)$ of $P(\rho_v/O, I)$ is calculated from the samples as follows

$$\hat{P}\left(\rho_v/O, I\right) = \frac{1}{L}\sum_{n=1}^{L} \frac{1}{\sqrt{2\pi}\sigma_v} exp\left(-\frac{\left(\rho_v - \rho_v^{(n)}\right)^2}{2\sigma_v^2}\right) \qquad (20)$$

where σ_v is the bandwidth of the Gaussian kernel which is set to $\frac{\delta_v}{L^{\frac{1}{5}}}$. L is the number of samples generated by the MH algorithm (5000 in our case) and δ_v is the standard deviation of the generated samples. This rule of thumb to calculate the bandwidth of the Gaussian kernel is discussed in [19].

As we can see in Figure 4, there is almost no difference in the inference of the marginal posterior distributions of the unknown parameters between the message passing algorithm and the MCMC approach. The posterior mean of ρ_v is calculated from the message passing algorithm as $\int_0^1 \rho_v \gamma \prod_{i,j} \mu_{P_{i,j}\to\rho_v}(\rho_v) d\rho_v$ and from the MCMC approach as $\frac{1}{L}\sum_n \rho_v^{(n)}$. These come out to

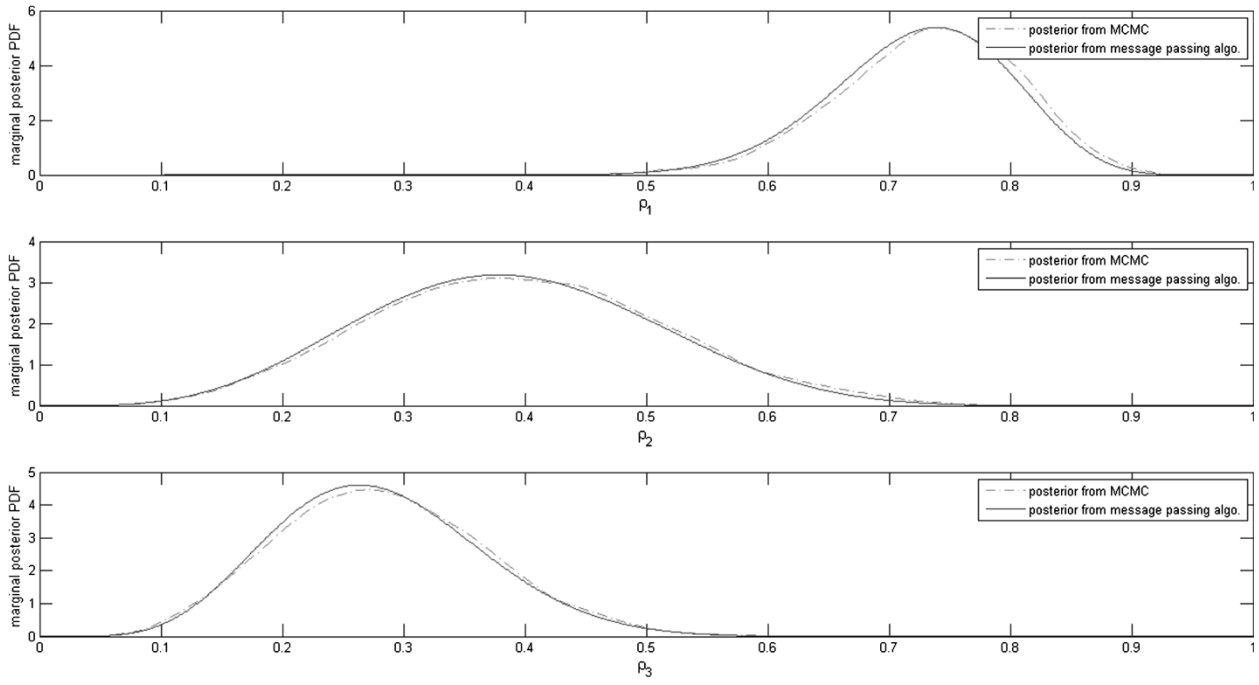

Figure 4 Marginal posterior distribution of ρ_1 through ρ_3 calculated using both the message passing algorithm and the MCMC approach.

be (0.7254 0.3891 0.2799) and (0.7326 0.3961 0.2830) respectively. These estimates are close to each other and to the actual values of (0.7 0.4 0.2).

This simulation shows that the message passing algorithm successfully calculates the posterior marginal distributions of the unknown parameters ρ_1 through ρ_3 and gives the same inferences as the Metropolis-Hastings algorithm. We did simulations with various values of ρ and for different sets of locations of faults. The iterative message passing algorithm gave estimates of the posterior marginal distributions of the parameters same as those estimated using the MCMC approach for all the test cases considered in our simulations.

Applications to real data

To test our model, we performed experiments on healthy adult fibroblasts where it is fair to assume that there are no cancer causing mutations present in the tissue. Hence it is fair to assume that a Boolean regulatory network with no faults would best model this tissue.

Adult fibroblasts were grown in petri-dishes till confluence and then maintained in 0.2% FBS (Fetal Bovine Serum) for four days. It is a general assumption that FBS contains most of the important growth factors. After this, the cells were exposed to 0.2% FBS and 100 μM Anisomycin for 30 minutes. Anisomycin is a protein synthesis inhibitor which activates the MAPK signal transduction network and keeps it responsive to kinase specific inhibitors [20,21]. That is, with the addition of Anisomycin, we anticipate the MAPK signal transduction network to respond to the addition of kinase inhibitors such as U0126. Anisomycin, being a protein synthesis inhibitor, would also cut off any feedback path which has a translation (protein synthesis) step in it. The cells were then grouped into three groups (group 0, group 1, and group 2). Group 0 was the control group which was exposed to 100 μM Anisomycin only. Group 1 was exposed to 100 μM Anisomycin and 50 μM of LY294002. Group 2 was exposed to 100 μM Anisomycin, 50 μM of LY294002, and 10 μM of U0126. All three groups were also exposed to 20% FBS along with the other chemicals. LY294002 and U0126 are highly specific inhibitors of PI3 Kinase (PI3K in Figure 1) and MEK1 respectively. The molecular targets of LY294002 and U0126 are shown in Figure 1. Genes having the SP1 and SRF-ELK response elements in their promoters were quantified through real time PCR and the delta-delta method [22]

Table 1 Gene expression levels and their discrete values for the gene EGR1

group 1	*normalized gene expression*	0.5987	0.7320	0.5586	0.6199
	discrete value	1	1	1	1
group 2	*normalized gene expression*	0.4796	0.2892	0.2535	0.2698
	discrete value	1	0	0	0

The threshold level using Otsu's method comes out to be 0.3824 for EGR1.

Table 2 Table showing the normalized gene expression ratios and their Reference Sequence (RefSeq) numbers

	EGR1	JUN	CMYC	DECORIN	IRF3	VEGFA
RefSeq	**NM_001964.2**	**NM_002228.3**	**NM_002467.4**	**NM_133503.2**	**NM_001571.5**	**NM_003376.5**
Group 1	0.5987	0.4931	0.3209	0.4353	0.5176	0.4444
	0.7320	0.6736	0.2852	0.4601	0.4204	0.4989
	0.5586	0.6598	0.3439	0.4147	0.3560	0.5176
	0.6199	0.7792	0.2994	0.4323	0.3345	0.5105
Group 2	0.4796	0.1550	0.2570	0.2793	0.2624	0.3164
	0.2892	0.2793	0.2059	0.3789	0.2553	0.4601
	0.2535	0.3015	0.2717	0.3737	0.2253	0.4633
	0.2698	0.3415	0.2679	0.3536	0.2031	0.3660

with GAPDH as the reference gene and group 0 as the control. The genes were measured in quadruplets for each experiment.

EGR1 is measured as a reporter gene of SRF-ELK transcription factor [23]. JUN, and cMYC are measured as reporters of SP1 [24,25]. Other genes having the SP1 response element in their promoters are Decorin, IRF3 and VEGFA [26-29]. These six genes were quantified in quadruplets for each experiment. The readings of each gene are discretized using Otsu's method [30]. As an example the readings of ERG1 and their corresponding discretized values are shown in Table 1. The threshold level for EGR1 came out to be 0.3824. the expressions above this level are labeled as 1 and those below are labeled as 0. The measured normalized gene expression ratios are shown in Table 2.

For demonstration purposes, we have taken the set of locations where to search for faults to be composed of ERK1/2 and IRS1 (shown in Figure 1). The marginal posterior probability distributions of the probabilities of faults associated with these two locations are shown in Figure 5.

As we can see in Figure 5, the posterior marginal distribution associated with ERK1/2 comes out to be quite tightly distributed with a mean of 0.1538 while that for IRS1 comes out to be uniformly distributed between 0 and 1. This is because the data does not contain any discriminating information about the occurrence of any fault at IRS1 under this MAPK Boolean model. But it does tell us

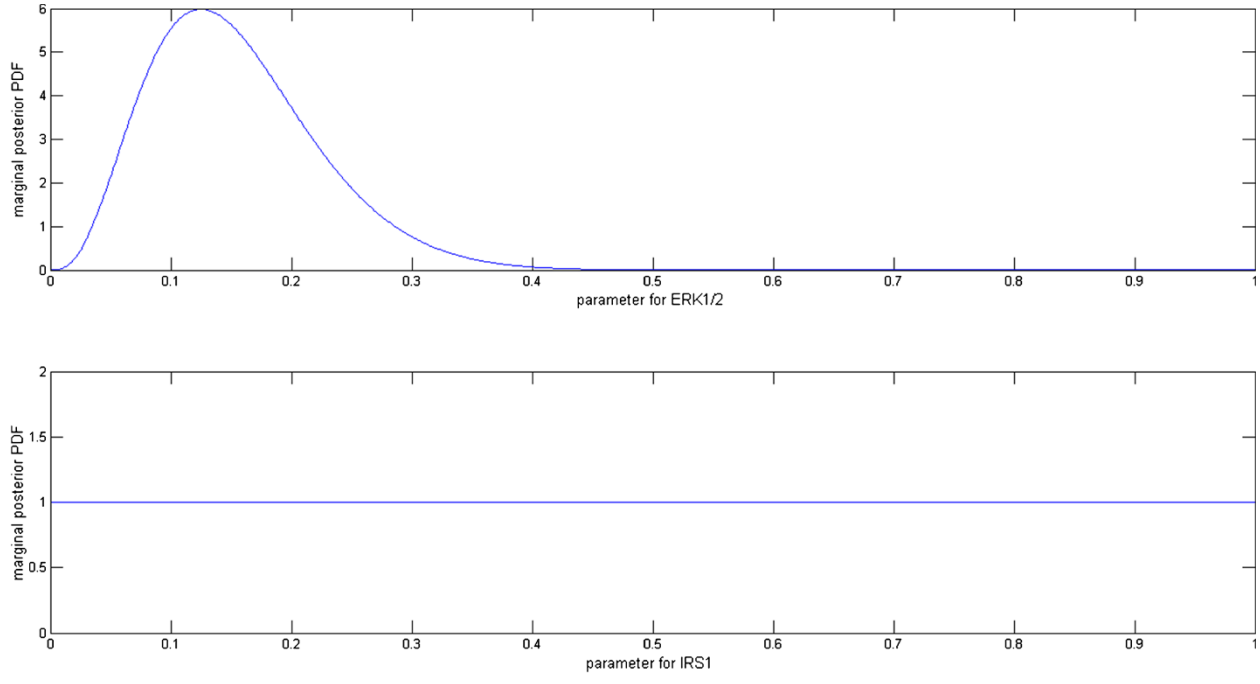

Figure 5 Marginal posterior distribution of the unknown parameters associated with ERK1/2 and IRS1.

that the probability of occurrence of a fault at the variable corresponding to ERK1/2 is pretty low, judging by its mean to be having a low value of close to 15%. This is expected since the data comes from adult fibroblasts, where we can be fairly sure that no cancer causing mutations are present. If data had been collected after exposure to other combinations of other drugs (for instance Lapatinib or Temsirolimus) then the data might have allowed the model to make meaningful inferences regarding occurrences of faults at locations besides ERK1/2 as well as give sharper confidence intervals than that shown in Figure 5.

Conclusion

In this paper we have described a method to estimate the probabilities with which certain faults have taken place in a given Boolean Regulatory network, provided we have the observations of the observable variables whose behavior is determined by the network. We have described the probability model and described a fast algorithm based on message passing to make the inferences about the posterior marginal probability distributions of the unknown parameters of the model (These parameters being the probabilities of the occurrences of the faults). We have compared the performance of the algorithm with Markov Chain Monte Carlo techniques (the Metropolis-Hastings Algorithm) through simulations, and we have shown that the message passing algorithm gives results comparable to those obtained using the MCMC methods with the added advantage of much smaller computation times. We also applied the model to analyze data collected from fibroblasts, thereby demonstrating how this model can be used on real world data. Such a computationally manageable approach has the potential to allow the inference of locations of faults in a Boolean regulatory network in a probabilistic setting from data, such as gene expression data.

Locating the points of dysregulations in a deterministic Boolean signal transduction network could be used to suggest therapies as described in [9]. Since we are locating faults in a probabilistic setting, the therapy could be designed keeping in mind the tradeoff between treating cancer and managing the side effects of the treatment. For example, consider a case where we have two possible locations of faults. Let the computed probability of the occurrence of a fault at the first location be smaller than that of the second location. Then we may only consider the second fault in our therapy design process, thereby reducing the exposure of the patient to excessive drugs which may have unwanted side effects.

Future work could focus on performing experiments on cancerous cell lines being exposed to various combinations of drugs and infer from the collected data the likely locations of dysregulations in the corresponding Boolean regulatory network. Also, algorithms could be developed to automate the process of selecting the set of locations of faults instead of having the user provide it to the algorithm.

Availability of codes and data

The MATLAB codes and the data can be obtained by sending a request to anwoy.rkl@gmail.com.

Competing interests

The authors declare that they have no competing interests.

Authors' contributions

AKM designed and implemented the algorithm and performed the computational experiments. AD conceived the study. VV set up the wet-lab experiments. All authors read and approved the final manuscript.

Acknowledgements

This work was supported by the National Science Foundation under Grant ECCS-1068628 and by a Texas A&M Engineering Genomics and Bioinformatics Seed Grant.

Author details

[1]Department of Electrical and Computer Engineering, Texas A&M University, 77843 College Station, USA. [2]Department of Veterinary Integrated Biosciences, College of Veterinary Medicine, Texas A&M University, 77845 College Station, USA.

Received: 16 April 2014 Accepted: 9 July 2014
Published: 16 August 2014

References

1. Bower JM, Bolouri H: *Computational Modeling of Genetic and Biochemical Networks, 1st edition*. Boston: MIT Press; 2001.
2. Shmulevich I, Dougherty ER, Kim S, Zhang W: **Probabilistic Boolean networks: a rule-based uncertainty model for gene regulatory networks.** *Bioinformatics* 2002, **18**(2):261–274.
3. Datta A, Dougherty E: *Introduction to Genomic Signal Processing with Control*. New York: CRC Press; 2007.
4. Friedman N, Linial M, Nachman I, Pe'er D: **Using Bayesian networks to analyze expression data.** *J Comput Biol* 2000, **7**(3–4):601–620.
5. Zou M, Conzen SD: **A new dynamic Bayesian network (DBN) approach for identifying gene regulatory networks from time course microarray data.** *Bioinformatics* 2005, **21:**71–79.
6. Liang S, Fuhrman S, Somogyi R: **REVEAL, a general reverse engineering algorithm for inference of genetic network architectures.** *Pac Symp Biocomput* 1998, **3**(3):18–29.
7. Layek RK, Datta A, Dougherty ER: **From biological pathways to regulatory networks.** *Mol BioSyst* 2011, **7:**843–851.
8. Mohanty AK, Datta A, Venkatraj V: **A model for cancer tissue heterogeneity.** *IEEE T Bio-Med Eng* 2014, **61**(3):966–974.
9. Layek RK, Datta A, Bittner M, Dougherty ER: **Cancer therapy design based on pathway logic.** *Bioinformatics* 2011, **27**(4):548–555.
10. Weinberg RA: *The Biology of Cancer, 1st edition*. Princeton: Garland Science; 2006.
11. Greenman C, Wooster R, Futreal PA, Stratton MR, Easton DF: **Statistical analysis of pathogenicity of somatic mutations in cancer.** *Genetics* 2006, **173**(4):2187–2198.
12. Goldman N, Yang Z: **A codon-based model of nucleotide substitution for protein-coding DNA sequences.** *Mol Biol Evol* 1994, **11**(5):725–736.
13. Yang Z, Ro S, Rannala B: **Likelihood models of somatic mutation and codon substitution in cancer genes.** *Genetics* 2003, **165:**695–705.
14. Kschischang FR, Frey BJ, Loeliger HA: **Factor graphs and the sum-product algorithm.** *IEEE T Inform Theory* 2001, **47**(2):498–519.
15. Wymeersch H: *Iterative Receiver Design*. New York: Cambridge University Press; 2007.
16. Gelman A, Carlin JB, Stern HS, Rubin DB: *Bayesian Data Analysis, 2nd edition*. Boca Raton, London, New York, Washington D.C.: Chapman and Hall/CRC; 2004.

17. Gelman A, Hill J: *Data Analysis Using Regression and Multi-level/hierarchical Models*. New York: Cambridge University Press; 2007.
18. Hoff PD: *A First Course in Bayesian Statistical Methods*. Dordrecht, Heidelberg, London, New York: Springer Texts in Statistics; 2009.
19. Scott DW: *Multivariate Density Estimation: Theory, Practice, and Visualization*. New York, Chichester, Brisbane, Toronto, Singapore: John Wiley & Sons; 1992.
20. Bébien M, Salinas S, Becamel C, Richard V, Linares L, Hipskind RA: **Immediate-early gene induction by the stresses anisomycin and arsenite in human osteosarcoma cells involves MAPK cascade signaling to Elk-1, CREB and SRF.** *Oncogene* 2003, **22**(12):1836–1847.
21. Dhawan P, Bell A, Kumar A, Golden C, Mehta KD: **Critical role of p42/44(MAPK) activation in anisomycin and hepatocyte growth factor-induced LDL receptor expression: activation of Raf-1/Mek-1/p42/44(MAPK) cascade alone is sufficient to induce LDL receptor expression.** *J Lipid Res* 1999, **40**(10):1911–1919.
22. Livak KJ, Schmittgen TD: **Analysis of relative gene expression data using real-time quantitative PCR and the $2^{-\Delta\Delta C_t}$ method.** *Methods* 2001, **25**(4):402–408.
23. Clarkson RW, Shang CA, Levitt LK, Howard T, Waters MJ: **Ternary complex factors Elk-1 and Sap-1a mediate growth hormone induced transcription of Egr-1 (early growth response factor-1) in 3T3-F442A Preadipocytes.** *Mol Endocrinol* 1999, **13**(4):619–631.
24. Rozek D, Pfeifer GP: **In vivo protein-DNA interactions at the c jun promoter: preformed complexes mediate the UV response.** *Mol Cell Biol* 1993, **13**(9):5490–5499.
25. Levens D: **How the c-myc promoter works and why it sometimes does not.** *J Natl Cancer I Monographs* 2008, **39:**41–43.
26. Verrecchia F, Rossert J, Mauviel A: **Blocking sp1 transcription factor broadly inhibits extracellular matrix gene expression in vitro and in vivo: implications for the treatment of tissue fibrosis.** *J Invest Dermatol* 2001, **116**(5):755–763.
27. Xu HG, Jin R, Ren W, Zou L, Wang Y, Zhou GP: **Transcription factors Sp1 and Sp3 regulate basal transcription of the human IRF-3 gene.** *Biochimie* 2012, **94**(6):1390–1397.
28. Samson SL, Wong NC: **Role of Sp1 in insulin regulation of gene expression.** *J Mol Endocrinol* 2002, **29**(3):265–279.
29. Pagés G, Pouysségur J: **Transcriptional regulation of the vascular endothelial growth factor gene-a concert of activating factors.** *Cardiovasc Res* 2005, **65**(3):564–573.
30. Otsu N: **A threshold selection method from gray-level histograms.** *Automatica* 1975, **11**(285–296):23–27.

doi:10.1186/s13015-014-0020-6
Cite this article as: Mohanty *et al.*: Using the message passing algorithm on discrete data to detect faults in boolean regulatory networks. *Algorithms for Molecular Biology* 2014 **9**:20.

Submit your next manuscript to BioMed Central and take full advantage of:

- **Convenient online submission**
- **Thorough peer review**
- **No space constraints or color figure charges**
- **Immediate publication on acceptance**
- **Inclusion in PubMed, CAS, Scopus and Google Scholar**
- **Research which is freely available for redistribution**

Submit your manuscript at
www.biomedcentral.com/submit

BioMed Central