Technical Report MS-CIS-07-20

Genetic Network Identification Using Convex Programming

A. Agung Julius*, Michael Zavlanos*, Stephen Boyd†, and George J. Pappas*

* Department of Electrical and Systems Engineering
University of Pennsylvania
Philadelphia, PA 19104
{agung,zavlanos,pappasg}@seas.upenn.edu

† Department of Electrical Engineering
Stanford University
Stanford, CA 94305
boyd@stanford.edu

July 26, 2007

Abstract

Gene regulatory networks capture interactions between genes and other cell substances, resulting in various models for the fundamental biological process of transcription and translation. The expression levels of the genes are typically measured as mRNA concentration in micro-array experiments. In a so called genetic perturbation experiment, small perturbations are applied to equilibrium states and the resulting changes in expression activity are measured. One of the most important problems in systems biology is to use these data to identify the interaction pattern between genes in a regulatory network, especially in a large scale network.

In this paper, we develop a novel algorithm for identifying the *smallest* genetic network that explains genetic perturbation experimental data. By construction, our identification algorithm is able to incorporate and respect any *a priori knowledge* known about the network structure. A priori biological knowledge is typically *qualitative*, encoding whether one gene affects another gene or not, or whether the effect is positive or negative. Our method is based on a convex programming relaxation of the combinatorially hard problem of L_0 minimization, so it can efficiently handle large scale problems. We apply the proposed method to the identification of a subnetwork of the SOS pathway in *Escherichia coli*, the segmentation polarity network in *Drosophila melanogaster*, and a larger artificial network for measuring the performance of the method. In all cases, we show that our method performs better than prior methods.

1 Introduction

Genes in living cells regulate various cellular biochemical processes through genetic regulatory networks. In such a network, the genes produce (through transcription and translation) proteins that act as *transcription factors* for other genes or themselves. Transcription factors are proteins that upregulate or downregulate the genetic transcription activities. A gene, therefore, can regulate the activities of other genes by proxy. When the increase in the expression activity of gene X results in the increase of that of gene Y, then gene X is said to *activate* gene Y. On the other hand, if the expression activity of gene Y is decreased, then gene X is said to *inhibit* gene Y.

Various activation and inhibition patterns in genetic circuits result in various system behaviors, such as feedback, oscillation, and switch like behavior. When the network's interconnection pattern is known, it is possible to develop quantitative models that can predict its behavior. The effectiveness of such quantitative models has been verified through experimental observation of naturally existing systems (see, for example, [20]) and the construction of various synthetic genetic networks that function according to the design model (see, for example, [12, 10]).

The use of RNA microarray has made it possible to have an expression profile for a large number of genes when exposed to different conditions. One of the most important problems in systems biology is to use these data to identify the interaction pattern between genes in a regulatory network, especially in a large scale network. In the literature, this is sometimes called *reverse engineering* the genetic network. Genetic network identification has important potential applications, for example in drugs discovery where a systems wide understanding of the regulatory network is crucial for identifying the targeted pathways.

Genetic network identification is a very active research field. For an overview on existing results and methodologies, we refer the reader to [4, 16, 14, 11, 2] and the references therein. Based on the technique, we identify two classes of methods for network identification. The first class consists of methods that infer the network by clustering the genes based on their expression profiles. These methods do not view the network as a dynamical system, and the inferred network typically lacks causality information.

The second class consists of methods that infer the cause-effect relation between genes through various representations. Several typical representations are information-theoretic network, Bayesian network, and dynamical network described by ordinary differential equations (ODE) (see the survey in [2]). Information-theoretic network based methods typically lack the causality information, as they identify the network as undirected graphs. Bayesian network based methods identify the genetic network as directed graph and thus convey some causality information. However, they typically do not accommodate cycles in the network graph. This limitation can be significant, as feedback motifs are very common in genetic regulatory networks. Both causality and feedback motives limitation are not present in methods where that model the network as the network is modelled as a set of differential equations ([13, 22, 21, 3]). The method that we propose in this paper belongs to this class.

Based on the type of data used in the identification, there are two classes of methods. The first class deals with data obtained from dynamic time-series measurement of the expression profiles. The second class deals with steady state data, obtained by measuring the expression profiles when the network reaches an equilibrium. Our method belongs to the second class, where the identification of the interconnection pattern is done *locally* by perturbing the network around a given equilibrium. It is generally known that a regulatory network can have multiple stable equilibria.

The method that we propose aims at providing a minimal model that explains given genetic perturbation data. Obtaining such a minimal model is computationally very hard, as it involves combinatorial optimization problems that are known to be extremely hard to solve in large scale. Pioneering works by Gardner *et al* [13] and Tegner *et al* [22] provided an important step towards solving this problem. The method that they use introduces an *a priori* limitation on the connectivity of the network, and perform a combinatorial search on this limited set. The connectivity limitation is that each gene in the network has the same number of inputs. Another different approach where the connectivity is imposed on the number of outputs is reported in [23]. For every combination, the parameters of the model are deduced through a least-square fitting. Similar approach that uses least-square fitting but without minimization of the model is also reported in [1].

In solving this problem, we take a novel and different approach. Instead of imposing a connectivity limitation on the network, we do not have any limit on how many inputs each gene should have. The hard combinatorial problem is solved using a mathematical technique called *convex optimization* ([7]) after relaxing it as an ℓ_1 optimization problem ([6]). The same technique has been applied successfully in various fields where sparsity optimization is needed such as, portfolio optimization in finance ([18]), controller design in engineering ([15]), and electric power network design ([8]). Posing the problem as a convex optimization problem is actually very advantageous from the complexity point of view, as convex optimization algorithms can be implemented reliably to solve large scale problems.

Compared to the existing methods of the same class, our method has an advantage of being able to incorporate *a priori* knowledge about the network structure, encoding whether one gene affects another gene or not, or whether the effect is positive or negative. The identified model is then constructed to satisfy the *a priori* knowledge by default.

In this paper, we apply our method to two networks that have been previously identified in the literature: the SOS pathway in *Escherichia coli* ([13]) and the segmentation polarity network in *Drosophila melanogaster* ([22]). We also present a comparison between the performance of our method and those of the above mentioned papers. We also apply our method to an artificial network for its performance analysis.

2 Gene Network Modeling and Identification

A genetic regulatory network consisting of n genes in a genetic perturbation experiment can be modeled as an $n-$dimensional dynamical systems ([13, 22]). In general, such a model assumes the following form:

$$\frac{d\hat{x}}{dt} = f(\hat{x}, u), \ \hat{x} \in \mathbb{R}^n, \ u \in \mathbb{R}^p, \tag{1}$$

where $\hat{x}_i \in \mathbb{R}$ denotes the transcription activity (typically measured as mRNA concentration) of gene i in the network, and u_i is the so called transcription perturbation. In very large networks, we can typically assume that not all genes can be perturbed in the experiment, resulting in $p \leq n$.

Such nonlinear genetic networks can have multiple stable equilibria. Each equilibrium typically corresponds to a phenotypical state of the system. The dynamics close to a given equilibrium x_{eq} can be approximated by the set of linear differential equations,

$$\frac{dx}{dt} = Ax + Bu, \tag{2}$$

where $x := \hat{x} - x_{eq}$ ([21, 1]). The matrix $A \in \mathbb{R}^{n \times n}$ encodes pairwise interactions between the individual genes in the network at the given equilibrium or phenotypical state, while matrix $B \in \mathbb{R}^{n \times p}$ indicates which genes are affected by the transcriptional perturbations. Given that the system is stable around the equilibrium $x = 0$, if u is small enough, the system will move to a new equilibrium x, for which

$$Ax + Bu = 0. \tag{3}$$

Let $U = [U_1 \ \dots \ U_m] \in \mathbb{R}^{p \times m}$ denote the stack matrix of the transcription perturbations for different m experiments and $X = [X_1 \ \dots \ X_m] \in \mathbb{R}^{n \times m}$ denote the stack matrix of the corresponding steady state mRNA concentrations. In large networks or in cases where experiments are costly, we can typically assume that the experimental data set is smaller than the network size ([16]). That is, we assume $m < n$.

By collecting all m experiments at steady state, the equilibrium conditions (3) can be written as

$$AX + BU = 0. \tag{4}$$

In Equation 4, the matrices X and U are known, since they are measured (possibly with noise), and B is typically known. The goal of the method we propose in this paper is to find unknown matrix A, which models genetic network interactions and best explains the genetic perturbation experiments.

We aim at constructing a model that not only explains the perturbation data, but also incorporates some *a priori* knowledge about the system. Furthermore we would like to develop a model that constitutes a minimal network.

A priori biological knowledge is typically *qualitative*, encoding whether one gene affects another gene or not, or whether the effect is positive or negative. This knowledge is then manifested as pre-specified signs of some entries of the matrix A.

We quantify the size of the model as the number of connections in the network, i.e., the number of nonzeros entries in the matrix A. This is known as the L_0 norm of the matrix A. (Even though it is not a norm, it is common to refer to it as the L_0 norm.) The minimal model then corresponds to a network with as few connections as possible, or equivalently, the sparsest possible matrix A.

Obtaining a minimal model is clearly beneficial, as it reduces the complexity the model. A non-minimal model might be able to explain the data slightly better than a minimal one, for example due to measurement noise. However, this can lead to a phenomenon called *overfitting*, where a model includes unnecessary features to accommodate the noise. A minimal model is also desirable when the identified model is used in a pathway knockout. A non-minimal model might contain falsely identified spurious pathways.

When the available data for network identification is scarce, which is the case in large scale networks, constraints based on *a priori* knowledge and minimality can supplement the available data to obtain a good model. This is demonstrated in the segmentation polarity network of *Drosophila melanogaster* that we analyze in this paper.

3 Identification algorithm

Recall equation (4). The matrix B is known in advance, since it encodes the genes that are perturbed in each experiment. Without any loss of generality, we can take B to be an identity matrix of appropriate dimensions and encode the perturbation inputs in U.

In the absence of noise, a straightforward approach to this problem is to solve the linear matrix equation $AX = -U$, for the unknown matrix $A \in \mathbb{R}^{n\times n}$. If we have a sufficient number $m = n$ of independent experiments, resulting in X being an invertible matrix, we could solve uniquely for A using $A = -UX^{-1}$. However, as genetic networks grow dramatically in size as we reach genome-scale networks, having n independent experiments can be costly, both financially and timewise. Furthermore, the absence of noise is not a realistic assumption, except when we are dealing with *in silico* model. Consequently, we are not going to assume that right hand side of (4) is zero. Instead, we define identification error as

$$\eta := AX + U, \tag{5}$$

and try to minimize η (as a function of A) with respect to some metric, while obtaining a minimal model for A and satisfying the *a priori* constraints that might be imposed on A.

Notations. In this paper, we use the following matrix notation. If X is a matrix with n rows and m columns, we write $X \in \mathbb{R}^{n\times m}$. The symbol X_{ij} refers to the the entry of X at the i-th row, j-th column. A single index such as X_j refers to the column vector corresponding to the j-th column of X. The operator $\mathbb{E}[X_j]$ is the probabilistic expectation of X_j. The operator $\mathrm{Var}[X_j]$ is the covariance of X_j.

Error criterion. In this paper, we use the total squared error as the error criterion.

$$\mathrm{Err} = \sum_{j=1,\ldots,m} \sum_{i=1,\ldots,n} \eta_{ij}^2, \tag{6}$$

which can be compactly written as a Frobenius norm ($\|\cdot\|_F$)

$$\mathrm{Err} = \|\eta\|_F^2 . \tag{7}$$

However, if the covariance of the error in the j-th experiment is known, then the sum can be replaced by a more accurate weighted sum

$$\mathrm{Err} = \sum_{j=1,\ldots,m} \sum_{i=1,\ldots,n} \sum_{k=1,\ldots,n} \eta_{ij}\eta_{kj}R_{ik}^{j}, \tag{8}$$

or in a more compact notation

$$\mathrm{Err} = \sum_{j=1,\ldots,m} \eta_j^T R^j \eta_j, \tag{9}$$

where R^j is the inverse of the error covariance matrix in the j-th experiment. The intuition behind this weight is that the identification error in the experiments with more reliable data (smaller variance) weights more than that coming from less reliable data.

Model minimality criterion. We define the size of a model given by the matrix A as the number of nonzero entries in A, which is denoted by $\|A\|_0$. This is the number of connections in the model.

A priori knowledge constraint. Such knowledge typically has the form of (partial) sign pattern of the matrix A. We encode this by a matrix $S \in \{0, +, -, ?\}^{n\times n}$, where

$$\begin{bmatrix} S_{ij} = + \\ S_{ij} = - \\ S_{ij} = 0 \\ S_{ij} = ? \end{bmatrix} \Leftrightarrow \begin{bmatrix} A_{ij} \geq \varepsilon \\ A_{ij} \leq -\varepsilon \\ -\frac{\varepsilon}{2} \leq A_{ij} \leq \frac{\varepsilon}{2} \\ A_{ij} \in \mathbb{R} \end{bmatrix}. \tag{10}$$

Here ε is a small number, below which a connection is considered negligible. In this paper, we set $\varepsilon = 10^{-3}$. In short, such a pattern encodes known positive interactions (+), negative interactions (−), the absence of interactions (0), or simply lack of knowledge (?) between any two genes (or other substances) in the network. For example, a matrix consisting of only (?) indicates no *a priori* knowledge about the network.

Bias due to mRNA decay. The mRNA molecules produced in the transcription process decay. In reference to (1), a model for the dynamics of the mRNA concentration that explicitly takes into account the decay process can be written as

$$\frac{d\hat{x}}{dt} = f(\hat{x}, u) - \Lambda\hat{x}, \tag{11}$$

where $f(\hat{x}, u)$ denotes the part of the dynamics due to genetic regulation (activation and repression) and Λ is a diagonal matrix that contains the decay rate of each mRNA. Obviously, any network model that results from identification using genetic perturbation data will have a negative bias on the diagonal terms. When the decay rates are known, which is the case for the *in silico* model of the Drosophila segmentation polarity network, we can remove the bias. Otherwise, we have to take this into account when making statements about genes autoregulation.

Convex programming. The method that we propose in this paper is based on a mathematical technique called *convex programming.* Basically, convex programming is a mathematical theory for minimization of a convex cost function over a convex set of feasible solutions. Formulating the identification problem as convex programming is attractive because there are techniques for solving convex programming problems efficiently; see, e.g., [7].

Convex function: A function $f : D \to \mathbb{R}$ is called convex if for any $x, y \in D$ and $\lambda \in [0, 1]$, we have the following inequality

$$f(\lambda x + (1-\lambda)y) \leq \lambda f(x) + (1-\lambda)f(y).$$

Convex set: A set D is called convex if for any $x, y \in D$ and $\lambda \in [0, 1]$, $(\lambda x + (1-\lambda)y)$ is also included in D. It can be shown that the set of all matrices A that satisfies a given *a priori* knowledge constraint S is a convex set. Hereafter, we shall denote this set as $\mathcal{S}$.

Convex optimization solver. The convex optimization problems that we pose in this paper are solved using MATLAB with the toolbox `cvx` (see [5]) running on an Intel Xeon 2.8Ghz processor with 4GB RAM. `cvx` makes forming and solving the problem easy, but at the cost of efficiency. However, custom made implementations of convex optimization algorithms can easily handle problems with thousands of variables. Such efficient implementation of our algorithm can allow us in the future to handle genome scale problems.

The method that we use in this paper is detailed in two steps.

Step 1: Establishing baseline error level

In this step, we establish the least error level that a model can attain, while disregarding the model minimality criterion. As discussed above, when we assume no *a priori* knowledge about the statistics of the error, we can simply use the total squared error as the error criterion. Consequently, finding the baseline error level E_{bs} amounts to solving the following convex programming problem

$$\begin{array}{ll} \text{minimize} & \sum_{j=1,\ldots,m} \sum_{i=1,\ldots,n} \eta_{ij}^2 \\ \text{subject to} & \eta = AX + U, \qquad A \in \mathcal{S}, \end{array} \tag{12}$$

with optimization variable A.

When the statistics of the measurement errors in each experiment for X and U are known, we can compute the associated covariance of the error criterion as

$$\text{Var}\left[\eta_j\right] = A\text{Var}\left[X_j\right]A^T + \text{Var}[U_j]. \tag{13}$$

As discussed above, the error criterion that we use is given in (9), where

$$R^j = \left(\mathrm{Var}\left[\eta_j\right]\right)^{-1}. \tag{14}$$

Consequently, finding the baseline error level E_{bs} amounts to solving the following optimization problem

$$\begin{array}{ll} \text{minimize} & \sum_{j=1,\dots,m} \eta_j^T R^j \eta_j \\ \text{subject to} & \eta = AX + U, \qquad A \in \mathcal{S}, \\ & R^j = \left(A\mathrm{Var}\left[X_j\right] A^T + \mathrm{Var}[U_j]\right)^{-1}, \end{array} \tag{15}$$

where A is the variable.

However, this formulation is not convex. In order to solve it efficiently, we relax the problem by approximating the covariance matrices. First, assuming that the covariance matrices are identity matrices, we find the best model that minimizes the error criterion (12). Denote this model as $\tilde{A}$. The weight matrices R^j are then given by

$$R^j = \left(\tilde{A}\mathrm{Var}\left[X_j\right] \tilde{A}^T + \mathrm{Var}[U_j]\right)^{-1}. \tag{16}$$

The baseline error level E_{bs} is then computed by solving the following convex optimization problem.

$$\begin{array}{ll} \text{minimize} & \sum_{j=1,\dots,m} \eta_j^T R^j \eta_j \\ \text{subject to} & \eta = AX + U, \qquad A \in \mathcal{S}, \end{array} \tag{17}$$

with A as the variable.

Step 2. Minimizing the model

The baseline error level and the approximated error covariance matrix that we obtain in Step 1 above are used in finding a minimal model that can explain the data reasonably well. That is, we search for a minimal model that results in an error level of at most βE_{bs}, where $\beta \geq 1$ is a predetermined parameter. If $\beta = 1$, we are interested in models that result in the same error level as the one obtained in Step 1. The bigger β is, the more variation we allow for the identified model, and thereby possibly obtain a smaller model at the cost of higher error level. Thus, β allows us to control the tradeoff between model accuracy and model minimality.

Mathematically, Step 2 can be formulated as the following optimization problem

$$\begin{array}{ll} \text{minimize} & \|A\|_0 \\ \text{subject to} & \eta = AX + U, \qquad A \in \mathcal{S}, \\ & \sum_{j=1,\dots,m} \eta_j^T R^j \eta_j \leq \beta E_{\text{bs}}, \end{array} \tag{18}$$

where A is the variable. We denote the solution of this problem as $A_{\min}$. Although the constraints in the problem above defines a convex feasible set, the cost function itself is not convex. In fact, the problem has combinatorial complexity as we have to search in the set of all possible interconnection patterns. This means that the complexity of the problem increases very rapidly with the its size and thus makes it practically impossible to solve it in a large scale. In order to solve this problem with convex programming, we relax the L_0 minimization problem as a weighted ℓ_1 minimization. The same technique has been applied successfully in various fields where sparsity optimization is needed such as, portfolio optimization in finance ([18]), controller design in engineering ([15]), and electric power network design ([8]).

Step 2.1: Initiate $A_{\text{old}} = 0$ and $W_{ij} = 1$, $i = 1, 2, \dots, n$, $j = 1, 2, \dots m$.

Step 2.2: Find A_{update} by solving the convex optimization problem

$$\begin{array}{ll} \text{minimize} & \sum_{i=1,\dots,n} \sum_{j=1,\dots,m} W_{ij} \, |A_{ij}| \\ \text{subject to} & \eta = AX + U, \qquad A \in \mathcal{S}, \\ & \sum_{j=1,\dots,m} \eta_j^T R^j \eta_j \leq \beta E_{\text{bs}}, \end{array} \tag{19}$$

where A is the variable.

Step 2.3: Update $W_{ij} = f(A_{ij})$. Here $f(A_{ij})$ is a function that assigns a weight matrix for the convex cost function in Step 2.2. The choice for the function is explained in more detail later.

Step 2.4: If $\|A_{\text{old}} - A_{\text{update}}\|_F \geq \varepsilon$, then update $A_{\text{old}} = A_{\text{update}}$ and go to step 2.2, otherwise stop the iteration and return the current solution as the optimal value.

$$A_{\min} = A_{\text{update}}. \tag{20}$$

Here ε is a small number that we choose to indicate the convergence of the iteration. Throughout this paper, we use $\varepsilon = 10^{-3}$, unless stated otherwise.

Choosing the weight function. The weight function $f(A_{ij})$ is designed such that the entries of A that are small are given larger weight than the larger entries ([6]). This is because we are interested in maximizing the number of zero entries in A. We thus emphasize more on

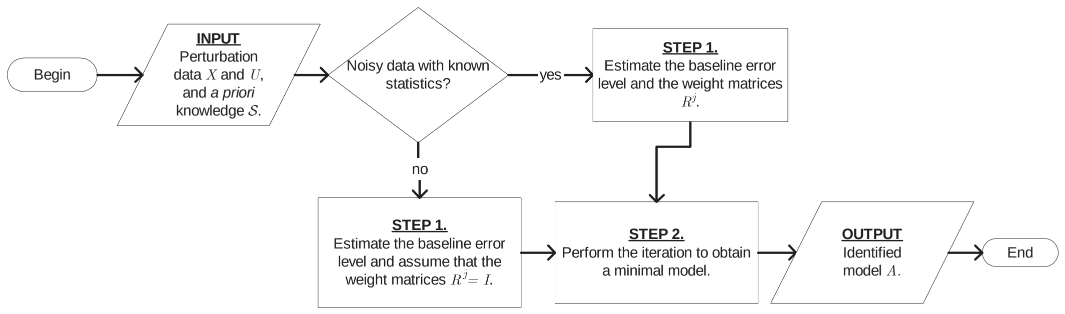
Begin
INPUT
Perturbation data X and U, and *a priori* knowledge $\mathcal{S}$.
Noisy data with known statistics?
yes
no
STEP 1.
Estimate the baseline error level and the weight matrices R^j.
STEP 1.
Estimate the baseline error level and assume that the weight matrices $R^j = I$.
STEP 2.
Perform the iteration to obtain a minimal model.
OUTPUT
Identified model A.
End

Figure 1: The flowchart summarizing the method proposed in this paper. The inputs to the algorithm are the perturbation data represented by the matrices X and U, as well as potential *a priori* structural knowledge about the network, which is represented by $\mathcal{S}$. The output of the algorithm is the identified model, represented by the matrix A.

reducing the entries that are already small. The generic form of $f(A_{ij})$ that we use in this paper is as follows.

$$f(A_{ij}) = \frac{\delta^p}{\delta^p + |A_{ij}|^p}, \tag{21}$$

where δ is a small number that acts a treshold, below which a number is considered "small". The exponent p determines the shape of the function. Throughout this paper, we use $\delta = 10^{-2}$ and $p = 1$, unless stated otherwise.

The overall identification procedure can be summarized in the flowchart in Figure 1.

4 Results and discussion

4.1 The segmentation polarity network of *Drosophila melanogaster*

We apply our proposed method to genetic perturbation data obtained from an *in numero* experiment based on the model given in [24, 22]. The network model depicts interaction between five genes and five transcription factors, as shown in Figure 2.

The gene *ci* produces the *Cubitus interruptus* protein that further undergoes a post-translational modification into the activator form (CI) or the repressor form (CN). *Cubitus interruptus* activator acts as an activator for the genes *ptc* and *wg*, while the *Cubitus interruptus* repressor represses the genes *ptc*, *wg*, *en* and *hh*. The gene *wg* produces the protein *wingless* (WG) that in turn acts as a self-activator. The gene *ptc* produces the protein *patched* (PTC) that inhibits the modification of *Cubitus interruptus* into the activator form and promotes the modification

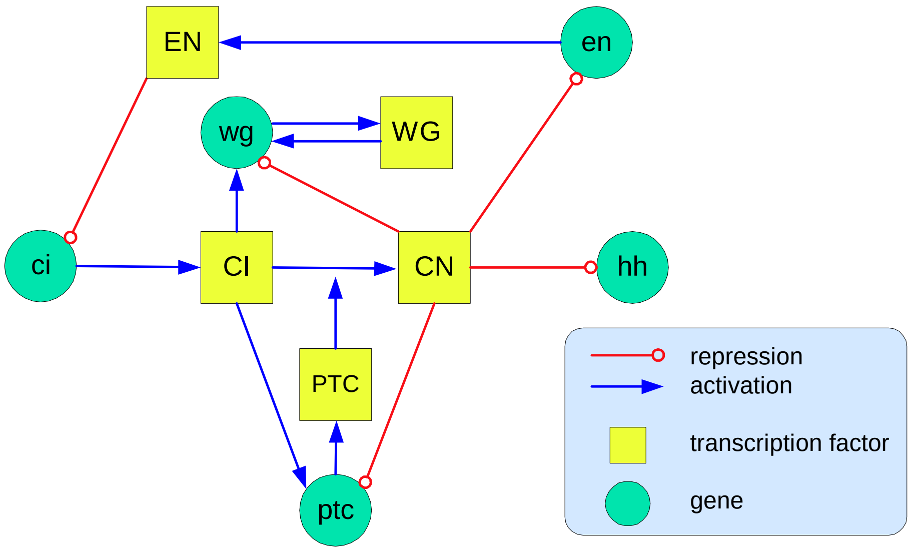

Figure 2: The segmentation polarity network of *Drosophila melanogaster* ([22]).

into the repressor form, effectively forming a negative feedback loop. The gene *en* produces the protein *engrailed* (EN) that represses the transcription of *ci* and promotes the transcription of *hh*.

We obtain the perturbation data X and U by numerically integrating the model provided in [22]. We first simulate the model without any perturbation to obtain an equilibrium. We then perform simulations with constant perturbation to each of the five genes in the model. The perturbation is set at 10^{-3}. Thus, $U = 10^{-3} \cdot I$, where I is an identity matrix. The deviations from the unperturbed equilibrium are recorded in the X matrix.

Since the data is noiseless, we obtain the baseline error level by solving (12). We do not estimate any error covariance matrices, and the performance measure is thus given by (6). We then proceed with step 2, and use $\beta = 1.1$.

From the numerical model, we have precise knowledge about the mRNA decay rate. Using this information, we eliminate the negative (auto repression) bias in the identified network by adding a diagonal matrix Λ (see (11)).

Given the matrix A of the identified model, we denote the strongest intergene interaction as

$$\mu := \max_{i \neq j} \|A_{ij}\|. \tag{22}$$

An interaction represented by A_{ij} is considered strong if $\|A_{ij}\| > 0.1 \cdot \mu$, and weak if $10^{-3} \leq$

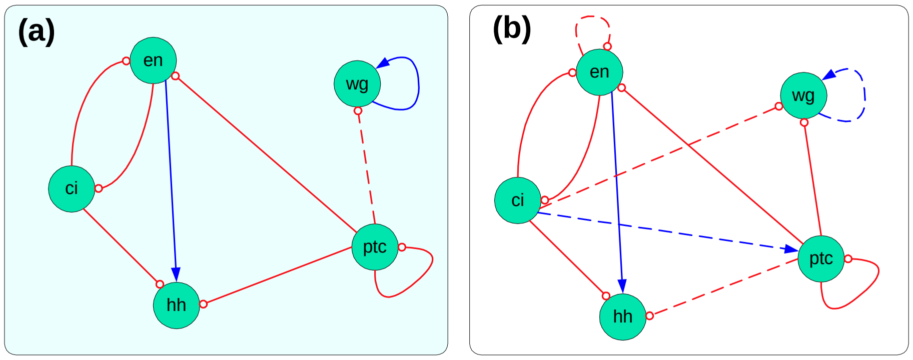

Figure 3: Identified network using *full* numerical data from the model of the segmentation polarity network of *Drosophila melanogaster*. Solid lines indicate strong interaction, while broken lines indicate weak interaction. **(a)** The network identified using the method proposed in this paper. **(b)** The network identified in [22].

$\|A_{ij}\| \leq 0.1 \cdot \mu$.

We execute our method to obtain a network model. The execution takes about 6 seconds on the platform that is detailed in the previous section. Figure 3 shows the result of our network identification method and the network identified in [22]. Our model uses the full set of numerical data from the model, in which all five genes are perturbed separately. We can see that our method produces a smaller model, and that all the identified connections can be accounted for based on the explanation above. The network in [22] contains a self repression loop in *en*, which is a false positive as it is not included in the real model. The connections from *ci* to *wg* and *ptc* are not present in our model. As explained above, *ci* produces both an activator and repressor for *wg* and *ptc*. The resulting connection is thus very weak and not included in our model.

We also apply the method on a partial data set. In this set, we do not include the data from the perturbation of *ptc*. The network identified using this data set is shown in Figure 4b. Observe that the connections from *ptc* to other genes disappear as expected, since there is no data that dictates their existence. This lack of data can be supplemented by *a priori* knowledge. We then include a sign pattern matrix as a constraint. The result is shown in Figure 4a. This network includes weak connections from *ptc* to itself, *en*, and *hh* as required by the constraint. These connections are weak because their existence is required by the *a priori* knowledge without any supporting data.

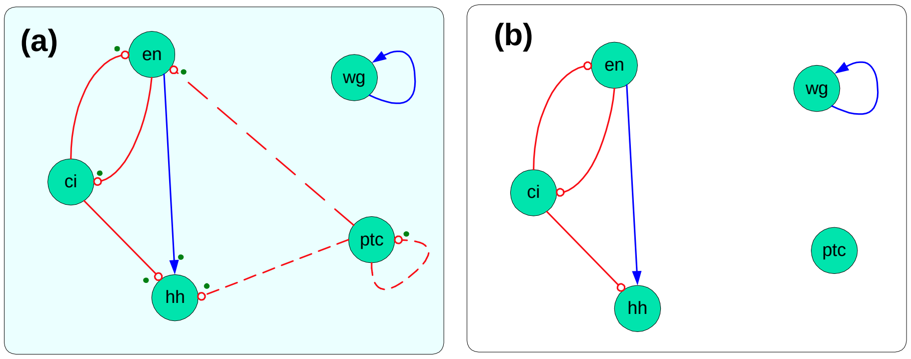

Figure 4: Identified network using *partial* numerical data from the model of the segmentation polarity network of *Drosophila melanogaster*. Solid lines indicate strong interaction, while broken lines indicate weak interaction. **(a)** The network identified when *a priori* knowledge about the sign pattern is incorporated. Arrows with green dots indicate interactions that are included in the *a priori* knowledge. **(b)** The network identified without incorporating any *a priori* knowledge.

4.2 SOS pathway in *Escherichia coli*

We also apply our proposed method on a subnetwork of the SOS pathway in *Escherichia coli*, using the genetic perturbation experimental data set provided in [13]. The subnetwork that we consider consists of nine genes and several transcription factors and metabolites, as shown in Figure 5 (taken from [13]).

The main pathway featured in this network is the pathway between the single-stranded DNA (ssDNA) and the protein LexA that acts as a repressor to several other genes. The protein RecA, which is activated by the single-stranded DNA, cleaves LexA and thus upregulates the above mentioned genes. Other key regulators in the network are the sigma factors $\sigma 70$, $\sigma 32$, and $\sigma 38$. These sigma factors play an important role in initiating transcription in heat shock and starvation responses.

We obtain the perturbation data X from [13]. Since there is no explicit mentioning of U, we assume that it is an identity matrix. Notice that this is a justifiable assumption, as a different value of U would just result in a scaling of the identified model.

The covariance matrix of the measurement in X is obtained by processing the standard error matrix provided in [13]. Assuming that the measurement errors of different genes are uncorrelated, we can obtain

$$(\mathrm{Var}[X_j])_{ik} = \begin{cases} \sigma_{ij}^2, & i = k, \\ 0, & i \neq k, \end{cases} \tag{23}$$

Genes	*recA*	*lexA*	*ssb*	*recF*	*dinI*	*umuDC*	*rpoD*	*rpoH*	*rpoS*
recA	?	$-$	?	?	?	?	$+$	?	?
lexA	$+$	$-$	?	?	?	?	$+$	?	?
ssb	$+$	$-$	?	?	?	?	$+$	?	?
recF	?	?	?	?	?	?	$+$	?	$+$
dinI	$+$	$-$	?	?	?	?	$+$	?	?
umuDC	$+$	$-$	?	?	?	?	$+$	?	?
rpoD	$+$	$-$	?	?	?	?	?	$+$	?
rpoH	?	?	?	?	?	?	$+$	?	?
rpoS	?	?	?	?	?	?	$+$	?	?

Table 1: A summary of known one-hop connections based on Figure 5. A $+$ sign indicates known activation, $-$ indicates known inhibition, and ? indicates unknown connection.

where σ_{ij} is the standard error of the measurement of gene i in the jth experiment. Since there is no information about the measurement error for U, we assume it is zero.

Based on Figure 5, we can identify known one-hop connections as summarized in Table 1. We use this table as *a priori* knowledge, when we perform our network identification method. We begin with Step 1 of the method to obtain a baseline error level and estimated error covariance. We use the estimated error covariance to obtain the weight matrices R^j that are used in the error criterion (9). We then proceed to step 2, and use different β values obtain different models and analyze them (as detailed below).

As comparison, we also perform the identification without estimating the error covariance and using the weighted sum (9) as our error criterion. Instead, we use identity weight matrices. As discussed in the previous section, this approach turns out to be inferior to the one with estimated error covariance.

The distinction between strong and weak connections follows the same convention as in the segmentation polarity network. An interaction represented by A_{ij} is considered strong if $\|A_{ij}\| > 0.1 \cdot \mu$, and weak if $10^{-3} \leq \|A_{ij}\| \leq 0.1 \cdot \mu$, where μ is the strongest intergenes interaction.

The results of our method are shown in Figure 6. In panel (a), we see the network identified using our method with estimated error covariance and $\beta = 1.75$. Recall that the parameter β quantifies the trade-off between model minimality and model accuracy. The higher β is, the more willing we are to trade accuracy with model minimality. Thus, higher β is expected to lead to smaller model (fewer connections).

The network identified in [13] is shown in Figure 6 panel (b). Comparing it to our result in panel (a), we can see that the network in (b) misidentifies several known one-hop interconnections,

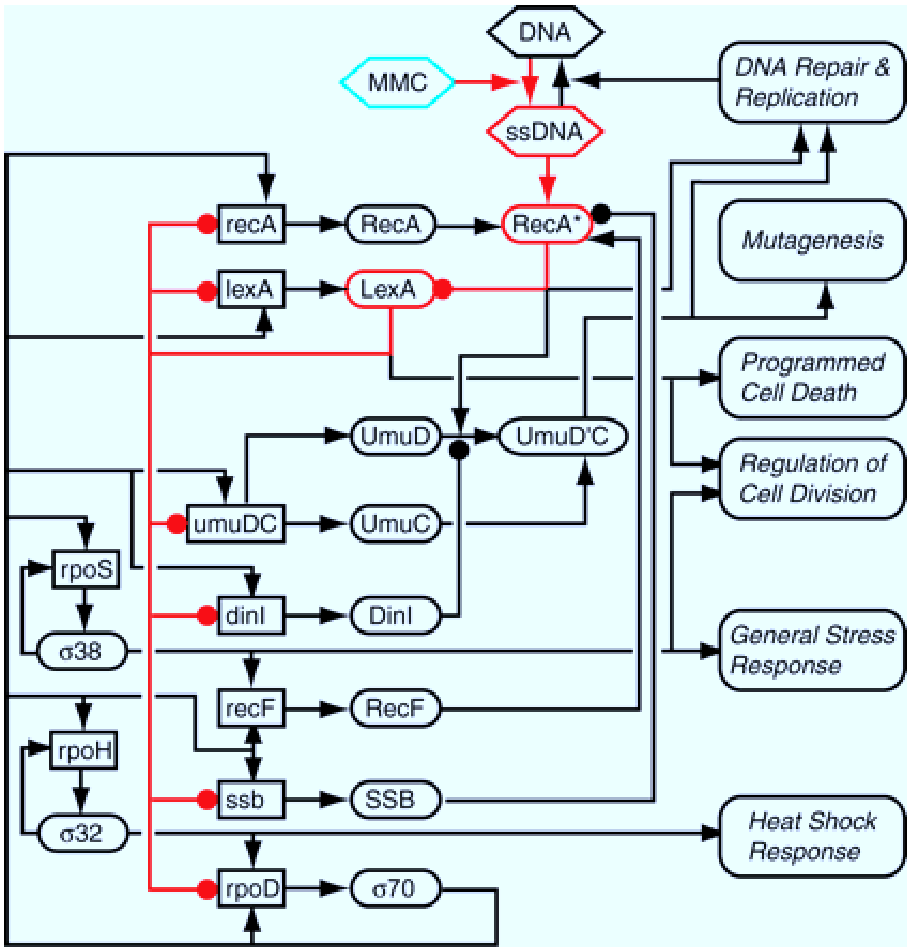

Figure 5: (Taken from [13]) Diagram of interactions in the SOS network. DNA lesions caused by mitomycin C (MMC) (blue hexagon) are converted to single-stranded DNA during chromosomal replication. Upon binding to ssDNA, the RecA protein is activated (RecA*) and serves as a coprotease for the LexA protein. The LexA protein is cleaved, thereby diminishing the repression of genes that mediate multiple protective responses. Boxes denote genes, ellipses denote proteins, hexagons indicate metabolites, arrows denote positive regulation, filled circles denote negative regulation. Red emphasis denotes the primary pathway by which the network is activated after DNA damage.

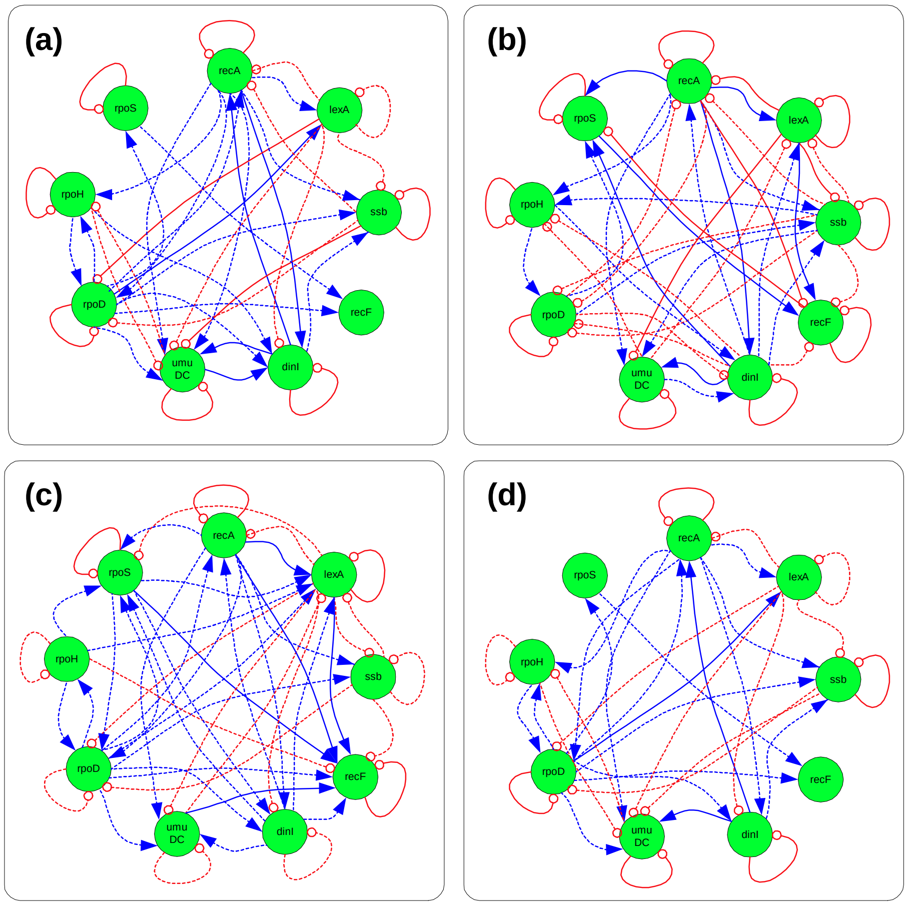

Figure 6: Identification results of the *Escherichia coli* SOS network. Lines with arrows indicate positive interaction (activation), while circles indicate negative interaction (repression). Solid lines denote strong interaction, while dotted lines denote weak interaction. **(a)** The result of our method using estimated error covariance and $\beta = 1.75$. **(b)** The network identified in [13]. **(c)** The result of our method without error covariance estimation and $\beta = 1.75$. **(d)** The result of our method using estimated error covariance and $\beta = 2$.

Genes	*recA*	*lexA*	*ssb*	*recF*	*dinI*	*umuDC*	*rpoD*	*rpoH*	*rpoS*
recA	?	−	−	+	+	−	+	0	0
lexA	+	−	−	+	+	−	+	0	0
ssb	+	−	−	+	+	−	+	0	0
recF	0	0	0	−	0	0	+	0	+
dinI	+	−	−	+	?	−	+	0	0
umuDC	+	−	−	+	+	−	+	0	0
rpoD	+	−	−	+	+	−	?	+	0
rpoH	0	0	0	0	0	0	+	?	0
rpoS	0	0	0	0	0	0	+	0	?

Table 2: A summary of known interactions from the literature (taken from [13]). A + sign indicates known activation, − indicates known inhibition, ? indicates unknown connections, and 0 indicates no known connection. The unknown connections come from known auto-activation, which when combined with mRNA decay results in unknown interaction.

such as the mutual repression between *recA* and *rpoD*.

The network in panel (c) is the result of our method without using error covariance estimation. This leads to worse error assessment in the model. As the result, although the known interactions are incorporated correctly in the model, we see that there are a number of false positives, particularly the interactions leading to *recF* and the genes that regulate the sigma factors *rpoD*, *rpoH*, and *rpoS*.

The network in panel (d) is the result of the same method that we use in panel (a), with higher β. We can indeed see that this change in the parameter β results in a smaller model.

The network that we identify in panel (a) includes a number of interactions that are not included in the *a priori* knowledge. Upon cross validation with the literature about the SOS network, we found that some of these new interactions are valid. For example, the protein dinI is known to stabilize recA* (cf. [19, 17]). Thereby, it effectively promotes the degradation of the repressor protein LexA, and thus activates *recA*, *lexA*, *ssb*, *dinI*, *umuDC*, and *rpoD*. Our result correctly predicts the positive interaction with *recA*, *ssb*, and *umuDC*.

A summary of known interactions in the literature has been compiled by [13] and shown in Table 2. We use this list as the "ground truth" and compare the results of the network identification methods with it.

The plot in Figure 7 shows a performance comparison of various identified models. Several observations that can be made from the comparison are:

1. Incorporating the estimated error covariance in the error assessment improves the performance of the method. Comparing Model A and Model D, we can see that using the estimated

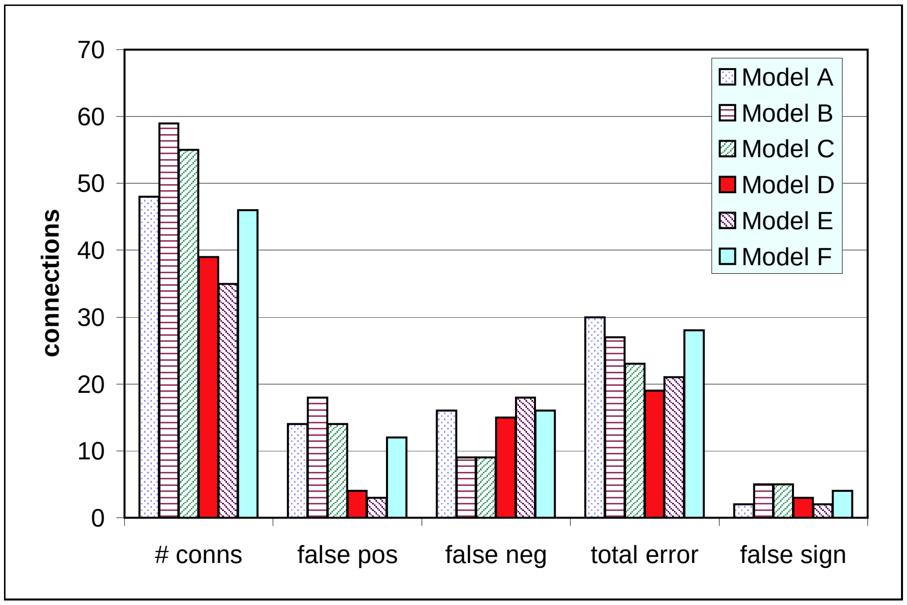

Figure 7: Performance comparison of various identified models of the *Escherichia coli* SOS network. Model A is the result of our method without using error covariance estimation and $\beta = 1.75$. This network is shown in Figure 6 panel (c). Model B, C, D and E are the results of our method using error covariance estimation, with $\beta = 1.1$, 1.25, 1.75, and 2, respectively. Model D and E are shown in Figure 6 panel (a) and (d). Model F is the network identified in [13], as also shown in Figure 6 panel (b).

error covariance, we get a smaller model with fewer false positives and negatives.

2. By increasing the value of β, we emphasize more on obtaining a smaller model (fewer connections) and less on the accuracy. Observing the results from Models B, C, D, and E, we can see that it leads to fewer false positives and more false negatives. The model is the fewest total error (Model D) is shown in Figure 6 panel (a).

3. The models identified using our method results in fewer errors compared to that in [13]. In particular, by tuning the value of β carefully, we can obtain models with very low number of false positives (less than 5%).

4.3 Larger artificial networks

We generate an artificial network with 20 genes and study the performance of our method under various noise levels and β parameter. As the measures of performance, we use the standard notions of *sensitivity*, *specificity*, *positive predictive value* (PPV), and *negative predictive value* (NPV) (cf. [9]). These parameters are formulated as

$$\begin{aligned}
\text{Sensitivity} &= \frac{\text{True Positive}}{\text{True Positive} + \text{False Negative}} && \text{(24a)}\\
\text{Specificity} &= \frac{\text{True Negative}}{\text{True Negative} + \text{False Positive}} && \text{(24b)}\\
\text{PPV} &= \frac{\text{True Positive}}{\text{True Positive} + \text{False Positive}} && \text{(24c)}\\
\text{NPV} &= \frac{\text{True Negative}}{\text{True Negative} + \text{False Negative}} && \text{(24d)}
\end{aligned}$$

The sensitivity of the result tells us the fraction of existing connections in the real system that are correctly identified as existent. The specificity of the result tells us the fraction of nonexisting connections that are correctly identified as nonexistent. The positive predictive value is a measure of how reliable a positive assertion ('Gene A regulates gene B') made by the result of test is. Similarly, the negative predictive value measures the reliability of a negative assertion ('Gene A does not regulate gene B').

As the true system, A_{true}, we generate a random network consisting of 20 genes. The network has 206 connections (out of 400 possible connections). We assume that 40% of these connections are known *a priori* and incorporate this knowledge into our method.

The perturbation input U is taken to be the identity matrix of size 20. From (3), we know that the true perturbation of the equilibria is given by $X_{\text{true}} = -A_{\text{true}}^{-1}U$.

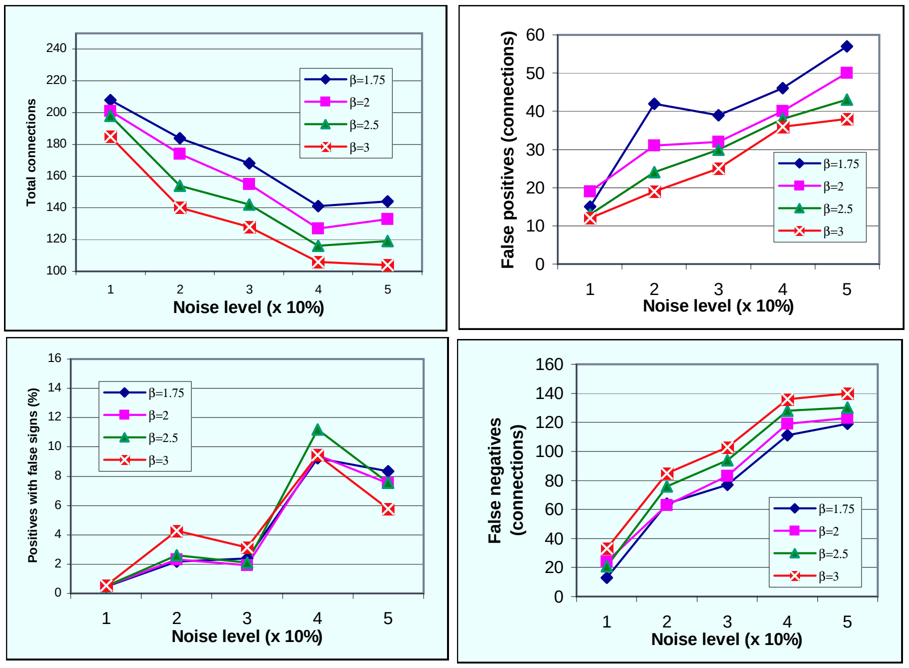

Figure 8: The performance of our method on an artificial network. The noise levels (1, 2, 3, 4, and 5) represent a noise standard deviation of 10%, 20%, 30%, 40%, and 50% the magnitude of the measurement in X respectively. Different β values are used as indicated in the plots. **(Top-left)** The number of connections in the identified models. **(Top-right)** Number of false positives in the identified models. **(Bottom-left)** The fraction of positives with false signs. This is computed by dividing the number of connection with false signs by the total number of connections (Top-left panel). **(Bottom-right)** Number of false negatives in the identified models.

To simulate noisy measurement, the perturbation data that we use in the network identification is obtained by adding some Gaussian noise to X_{true} according to the following formula.

$$X = X_{\text{true}} + \nu, \tag{25}$$

where ν is a matrix of zero-mean Gaussian random variables with the following standard deviation

$$\sigma(\nu_{ij}) = \text{NoiseLevel} \cdot \max(X_{\text{true},ij}, \delta_\nu), \tag{26}$$

and δ_ν is a small number that we introduce to prevent singularity of the covariance matrices. In the simulation, we use $\delta_\nu = 10^{-2}$. We generate five data sets, corresponding the noise levels of 10%, 20%, 30%, 40%, and 50%.

The performance of our method, applied to a random artificial network with 20 genes is shown

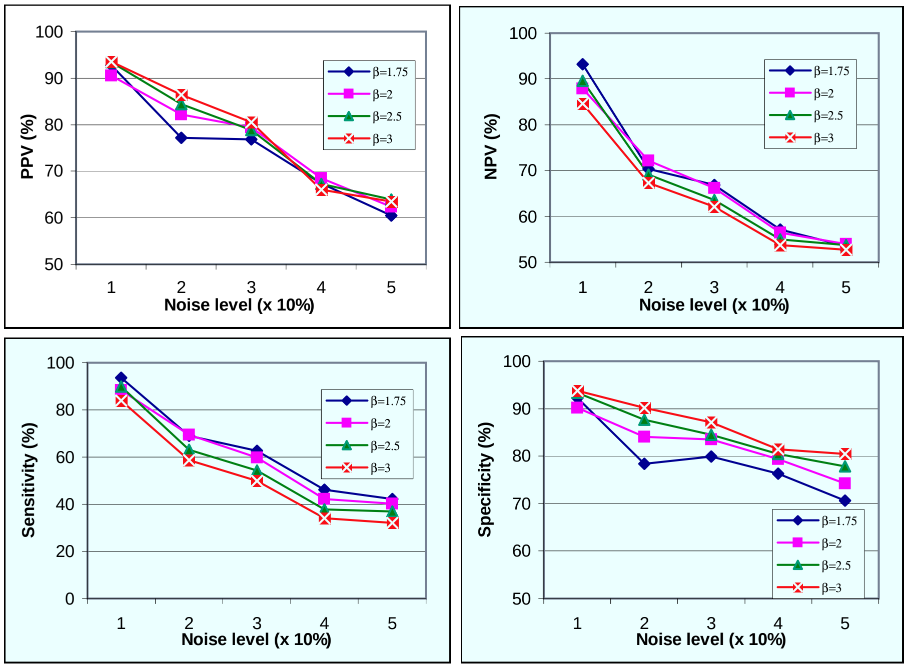

Figure 9: Performance measures of our method on an artificial network. These measures are computed according to (24). **(Top-left)** The positive predictive values of the results. **(Top-right)** The negative predictive values of the results. **(Bottom-left)** The sensitivity of the results. **(Bottom-right)** The specificity of the results.

in Figure 8 and Figure 9. Each data point represents an identified network obtained using the method on each of the five data sets. Each network identification takes about 9 minutes to complete (see the previous section for details on the software and hardware that are used). In applying our method, we assume that we have *a priori* knowledge about 40% of connections and use that in the method. Generally the performance of the method drops as the noise level increases, as expected.

As discussed earlier, the parameter β reflects how much we are willing to trade identification error with sparsity of the identified model. A higher value of β means more emphasis put on obtaining a sparser model. This also means that we are less likely to assign a connection in the identified model, as shown in the top-right panel of Figure 8. As the result, we can expect that higher β values generally lead to less false positive and more false negative (see the top-right and bottom-right panels of Figure 8). Consequently, it also leads to higher specificity and lower sensitivity of the identification results. This theory is confirmed by the results shown in Figure 9.

At 10% noise level, our result demonstrates very high sensitivity, selectivity, and predictive values (above 90% for certain β values). This is clearly better than the benchmark results reported in the survey paper by Bansal *et al* [2] for other existing algorithms with the same noise level.

From Figure 9 we can observe that although the performance of the method drops with increasing noise level, the specificity of the results can be maintained at a high level (above 70%). High level of specificity correlates with the sparsity of the model. Our method explicitly aims to obtain the smallest model, and as the result, most of the non-existing connections can be correctly identified.

5 Conclusion

We propose a method for identifying genetic regulatory networks using expression profiles from genetic perturbation experiments. Two features that distinguish our method from preexisting methods are, first, we aim at deriving a minimal model (characterized by the least number of connections) that explains the experimental data. Second, we can incorporate any *a priori* information about the structure of the network. Upon comparison with preexisting methods, we demonstrate that our method performs better.

Our method is based on convex programming relaxation, that approaches the combinatorially hard problem of finding a minimal model with efficient computational scheme. In this paper, we test our method in a prototypical implementation that handles module size networks. However, as efficient customized implementation of convex optimization algorithms are known to handle problems with thousands of variables, our method has a potential of solving the identification problem on a much larger scale.

Acknowledgement

The authors would like to thank Adam Halász, Marcin Imielinski, and Harvey Rubin for valuable discussion during the preparation of this paper. This work is partially funded by the ARO MURI SWARMS grant W911NF-05-1-0219, NSF award ECS-0423905, NSF award 0529426, and NASA award NNX07AEIIA.

References

[1] M. Andrec, B. N. Kholodenko, R. M. Levy, and E. Sontag. Inference of signaling and gene regulatory networks by steady-state perturbation experiments: structure and accuracy. *Journal of Theoretical Biology*, 232(3):427–441, 2005.

[2] M. Bansal, V. Belcastro, A. Ambesi-Impiombato, and D. di Bernardo. How to infer gene networks from expression profiles. *Molecular Systems Biology*, 3(10.1038/msb4100120), 2007.

[3] M. Bansal, G. Della Gatta, and D. di Bernardo. Inference of gene regulatory networks and compound mode of action from time course gene expression profiles. *Bioinformatics*, 22(7):815–822, 2006.

[4] R. Bonneau, D. J. Reiss, P. Shannon, M. Facciotti, L. Hood, N.S. Baliga, and V. Thorsson. The inferelator: an algorithm for learning parsimonious regulatory networks from systems-biology data sets *de novo*. *Genome Biology*, 7:R36, 2006.

[5] S. Boyd. cvx – MATLAB software for disciplined convex programming, 2005. http://www.stanford.edu/∼boyd/cvx/.

[6] S. Boyd. ℓ_1-norm norm methods for convex cardinality problems. Lecture Notes for EE364b, Stanford University, 2007. Available at www.stanford.edu/class/e364b/.

[7] S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004.

[8] S. Boyd, L. Vandenberghe, A. El Gamal, and S. Yun. Design of robust global power and ground networks. In *Proc. ACM/SIGDA Int. Symposium on Physical Design (ISPD)*, pages 60–65, April 2001.

[9] J. E. De Muth. *Basic Statistics and Pharmaceutical Statistical Applications*. Chapman & Hall/CRC, 2nd edition, 2006.

[10] M. B. Elowitz and S. Leibler. A synthetic oscillatory network of transcriptional regulators. *Nature*, 403(6767):335–338, 2000.

[11] J. J. Faith, B. Hayete, J. T. Thaden, I. Mogno, J. Wierzbowski, G. Cottarel, S. Kasif, J. J. Collins, and T. S. Gardner. Large-scale mapping and validation of *escherichia coli* transcriptional regulation from a compendium of expression profiles. *PLoS Biology*, 5(1):e8, 2007.

[12] T. S. Gardner, C. R. Cantor, and J. J. Collins. Construction of a genetic toggle switch in Escherichia coli. *Nature*, 403:339–342, 2000.

[13] T. S. Gardner, D. di Bernardo, D. Lorenz, and J. J. Collins. Inferring genetic networks and identifying compound mode of action via expression profiling. *Science*, 301:102–105, 2003.

[14] F. Geier, J. Timmer, and C. Fleck. Reconstructing gene-regulatory networks from time-series, knock-out data and prior knowledge. *BMC Systems Biology*, 1(11), February 2007. online publication.

[15] A. Hassibi, J. How, and S. Boyd. Low-authority controller design via convex optimization. *AIAA Journal of Guidance, Control, and Dynamics*, 22(6):862–872, November-December 1999.

[16] B. Hayete, T. S. Gardner, and J. J. Collins. Size matters: network inference tackles the genome scale. *Molecular Systems Biology*, 3(10.1038/msb4100118), 2007.

[17] S. Krishna, S. Maslov, and K. Sneppen. UV-induced mutagenesis in Escherichia coli SOS response: A quantitative model. *PLoS Computational Biology*, 3(3):e41 doi:10.1371/journal.pcbi.0030041, 2007.

[18] M. S. Lobo, M. Fazel, and S. Boyd. Portfolio optimization with linear and fixed transaction costs. *Annals of Operations Research*, 152(1):376–394, July 2007.

[19] S. L. Lusetti, O. N. Voloshin, R. B. Inman, R. D. Camerini-Otero, and M. M. Cox. The DinI protein stabilizes RecA protein filaments. *Journal of Biological Chemistry*, 279(29):30037–30046, 2004.

[20] E. M. Ozbudak, M. Thattal, H. N. Lim, B. I. Shraiman, and A. van Oudenaarden. Multistability in the lactose utilization network of *Escherichia coli*. *Nature*, 427:737 – 740, February 2004.

[21] E. Sontag, A. Kiyatkin, and B. N. Kholodenko. Inferring dynamic architecture of cellular networks using time series of gene expression, protein and metabolite data. *Bioinformatics*, 20(12):1877–1886, 2004.

[22] J. Tegner, M. K. S. Yeung, J. Hasty, and J. J. Collins. Reverse engineering gene networks: integrating genetic perturbations with dynamical modeling. *Proc. of the National Academy of Science*, 100(10):5944–5949, 2003.

[23] R. Thomas, S. Mehrotra, E. T. Papoutsakis, and V. Hatzimanikatis. A model-based optimization framework for the inference on gene regulatory networks from DNA array data. *Bioinformatics*, 20(17):3221–35, 2004.

[24] G. von Dassow, E. Meir, E. M. Munro, and G. M. Odell. The segment polarity network is a robust developmental module. *Nature*, 406:188–192, 2000.