Genetic Network Identification Using Convex Programming

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

* Dept. Electrical and Systems Engineering, University of Pennsylvania.

† Dept. Electrical Engineering, Stanford University.

January 14, 2009

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 *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. 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 an artificial network for measuring the performance of the method.

Keywords: Genetic networks, identification, convex optimization.

1 Introduction

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 drug 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 [1, 2, 3, 4, 5] 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.

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 [5]). 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 that model the network as the network is modelled as a set of differential equations [6, 7, 8, 9]. 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 exploration of all possible network topologies. Such a problem has been shown to have NP-Hard complexity [10]. Pioneering works by Collins *et al* (cf. [11, 6, 7]) provided an important step towards addressing this problem. In [11], the authors propose a method for noiseless measurements, where the optimization is relaxed as a nonrecursive ℓ_1 optimization problem. In [6, 7], 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 [12]. 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 [13].

In solving this problem, we take a 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* ([14]) after relaxing it as a *recursive* ℓ_1 optimization problem ([15, 16]). The same technique has been applied successfully in various fields where sparsity optimization is needed such as, portfolio optimization in finance ([17]) and controller design in engineering ([18]). Different techniques of convex ℓ_1 relaxation have been used in other works in reverse engineering of gene networks, such as [19, 20, 21, 22, 23]. 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.

Solving the problem with convex ℓ_1 relaxation has an advantage of being able to handle noisy data,

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* ([6]) and the segmentation polarity network in *Drosophila melanogaster* ([7]). We also apply our method to an artificial network to assess its performance, primarily in relation to other convex ℓ_1 relaxation methods.

2 Gene Network Modeling and Identification

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.

A genetic regulatory network consisting of n genes in a genetic perturbation experiment can be modeled as a dynamical system ([6, 7]). In general, such a model assumes the following form:

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

$$\frac{d\hat{y}}{dt} = G(\hat{x}, \hat{y}), \tag{2}$$

where $\hat{x}_i \in \mathbb{R}$ denotes the transcription activity (typically measured as mRNA transcript concentration) of gene i in the network, $\hat{y}_i$ denotes the protein concentration of protein i, and $\hat{u}_i$ 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 $p < n$.

The functions F and G summarize the dynamics of transcription and translation, as well as factors such as the degradation and dilution of transcripts and proteins. 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}, y_{eq}) can be approximated by the set of linear differential equations,

$$\frac{dx}{dt} = \mathfrak{A}_{11}x + \mathfrak{A}_{12}y + u, \tag{3}$$

$$\frac{dy}{dt} = \mathfrak{A}_{21}x + \mathfrak{A}_{22}y, \tag{4}$$

where $x := \hat{x} - x_{eq}$ and $y := \hat{y} - y_{eq}$ (cf. [8, 13]). The matrices $\mathfrak{A}_{ij}, i, j = 1, 2$ are the linearization of the dynamics near the equilibrium, while the vector $u \in \mathbb{R}^n$ represents the effect of the perturbation inputs in the linear model. In the case that not all genes can be perturbed, u is constrained in a subspace of $\mathbb{R}^n$. Given that the system is stable around the equilibrium $(x, y) = (0, 0)$, if u is small enough, the system will move to a new equilibrium (x, y), for which

$$\mathfrak{A}_{11}x + \mathfrak{A}_{12}y + u = 0, \tag{5}$$

$$\mathfrak{A}_{21}x + \mathfrak{A}_{22}y = 0. \tag{6}$$

We formulate a theory for the case when only the transcript concentrations are measured. In this case, we can flatten the transcription and translation layer of the regulatory system into an effective gene-gene regulatory network, by eliminating the protein concentration y. We therefore obtain

$$y = -\mathfrak{A}_{22}^{-1}\mathfrak{A}_{21}x, \tag{7}$$

$$\left(\mathfrak{A}_{11} - \mathfrak{A}_{12}\mathfrak{A}_{22}^{-1}\mathfrak{A}_{21}\right)x + u = 0. \tag{8}$$

We then define the matrix

$$A := \left(\mathfrak{A}_{11} - \mathfrak{A}_{12}\mathfrak{A}_{22}^{-1}\mathfrak{A}_{21}\right). \tag{9}$$

The matrix $A \in \mathbb{R}^{n\times n}$ encodes the effective pairwise interactions between the individual genes in the

network. In this representation, we have the relation

$$Ax + u = 0. \tag{10}$$

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 ([2]). That is, we assume $m < n$.

By collecting all m experiments at steady state, the equilibrium conditions (10) can be written as

$$AX + U = 0. \tag{11}$$

In Equation (11), the matrices X and U are known, since they are measured (possibly with noise). 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. *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. Furthermore, we would like to develop a model that constitutes a minimal network.

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 in a strict mathematical sense, 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.

3 Identification algorithm

Consider Equation (11). If we assume that the measurements are noise free and if we have a sufficient number $m = n$ of independent experiments, X is an invertible matrix, and we could obtain 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 (11) is zero. Instead, we define identification error as

$$\eta := AX + U, \tag{12}$$

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.

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

$$\text{Err} = \sum_{j=1,\dots,m} \sum_{i=1,\dots,n} \eta_{ij}^2. \tag{13}$$

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

$$\text{Err} = \sum_{j=1,\dots,m} \sum_{i=1,\dots,n} \sum_{k=1,\dots,n} \eta_{ij}\eta_{kj}R_{ik}^j = \sum_{j=1,\dots,m} \eta_j^T R^j \eta_j, \tag{14}$$

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

$$\begin{bmatrix} S_{ij} = + \\ S_{ij} = - \\ S_{ij} = 0 \\ S_{ij} = ? \end{bmatrix} \Leftrightarrow \begin{bmatrix} A_{ij} \geq \varepsilon \\ A_{ij} \leq -\varepsilon \\ -\frac{\varepsilon}{2} \leq A_{ij} \leq \frac{\varepsilon}{2} \\ A_{ij} \in \mathbb{R} \end{bmatrix}. \tag{15}$$

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 in the network. For example, a matrix consisting of only (?) indicates no *a priori* knowledge about the network. Hereafter, we shall denote the set of all matrices A that satisfy a given *a priori* knowledge constraint S as $\mathcal{S}$. A critical property of the set $\mathcal{S}$ that can be easily shown is that it is convex (cf. [14]).

Bias due to mRNA decay. As defined in (9), the effective gene-gene network model A is comprised of two parts. The direct transcript-transcript factor $\mathfrak{A}_{11}$ and the regulation through protein factor $-\mathfrak{A}_{12}\mathfrak{A}_{22}^{-1}\mathfrak{A}_{21}$.

We can typically assume that the dynamics of the concentration of protein i, y_i, is determined solely by its own transcript concentration x_i and its decay and/or dilution process. In this case, both $\mathfrak{A}_{21}$ and $\mathfrak{A}_{22}$ in (7) are diagonal matrices with positive and negative entries, respectively. We can therefore write

$$-\mathfrak{A}_{22}^{-1}\mathfrak{A}_{21} =: \Lambda_1, \tag{16}$$

where Λ_1 is a a diagonal matrix with positive entries. Similarly, if we assume that the dynamics of

the transcript concentration, x_i, is due to decay, we can write

$$\mathfrak{A}_{11} =: -\Lambda_2, \tag{17}$$

where Λ_2 is another diagonal matrix with positive entries. We then have

$$A = \Lambda_1 \mathfrak{A}_{12} - \Lambda_2, \tag{18}$$

where $\mathfrak{A}_{12}$ represents transcription regulation by proteins. Consequently, any network model that results from identification using genetic perturbation data will have a negative bias on the diagonal terms. When the decay rates Λ_2 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., [14].

Convex optimization solver. The convex optimization problems that we pose in this paper are solved using MATLAB with the toolbox `cvx` (see [24]) 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 allowing us in the future to handle genome scale problems.

The method that we use in this paper can be explained 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

$$\begin{array}{ll} \text{minimize} & \sum_{j=1,\dots,m} \sum_{i=1,\dots,n} \eta_{ij}^2 \\ \text{subject to} & \eta = AX + U, \qquad A \in \mathcal{S}, \end{array} \tag{19}$$

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

$$\text{Var}\left[\eta_j\right] = A\text{Var}\left[X_j\right] A^T + \text{Var}[U_j]. \tag{20}$$

As discussed above, the error criterion that we use is given in (14), where

$$R^j = \left(\text{Var}\left[\eta_j\right]\right)^{-1}. \tag{21}$$

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

$$\begin{array}{ll} \text{minimize} & \sum_{j=1,\dots,m} \eta_j^T R^j \eta_j \\ \text{subject to} & \eta = AX + U, \qquad A \in \mathcal{S}, \\ & R^j = \left(A\text{Var}\left[X_j\right] A^T + \text{Var}[U_j]\right)^{-1}, \end{array} \tag{22}$$

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 (19). Denote this model as $\tilde{A}$.

The weight matrices R^j are then given by

$$R^j = \left(\tilde{A}\mathrm{Var}\left[X_j\right]\tilde{A}^T + \mathrm{Var}[U_j]\right)^{-1}. \tag{23}$$

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

$$\begin{array}{ll} \text{minimize} & \sum_{j=1,\ldots,m} \eta_j^T R^j \eta_j \\ \text{subject to} & \eta = AX + U, \qquad A \in \mathcal{S}, \end{array} \tag{24}$$

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. 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

$$\begin{array}{ll} \text{minimize} & \|A\|_0 \\ \text{subject to} & \eta = AX + U, \qquad A \in \mathcal{S}, \\ & \sum_{j=1,\ldots,m} \eta_j^T R^j \eta_j \leq \beta E_{\mathrm{bs}}, \end{array} \tag{25}$$

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 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 recursive weighted ℓ_1 minimization.

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

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

$$\begin{array}{ll} \text{minimize} & \sum_{i=1,\ldots,n} \sum_{j=1,\ldots,n} W_{ij} \, |A_{ij}| \\ \text{subject to} & \eta = AX + U, \qquad A \in \mathcal{S}, \\ & \sum_{j=1,\ldots,m} \eta_j^T R^j \eta_j \leq \beta E_{\text{bs}}, \end{array} \tag{26}$$

where A is the variable.

Step 2.3: Update $W_{ij} = f(A_{\text{update},ij})$. Here $f(\cdot)$ 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.

$$A_{\min} = A_{\text{update}}. \tag{27}$$

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

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 ([15]). This is because we are interested in maximizing the number of zero entries in A. We thus emphasize more on reducing the entries that are already small. The generic form of $f(A_{ij})$ that we use in this paper is as follows.

$$f(A_{ij}) = \frac{\delta^p}{\delta^p + |A_{ij}|^p}, \tag{28}$$

where δ is a small number that acts a treshold, below which a number is considered "small". The parameter δ thus determines the 'boundary' between small values and large values. The term 'boundary' is to be interpreted loosely, as the transition is smooth. The exponent p determines the shape

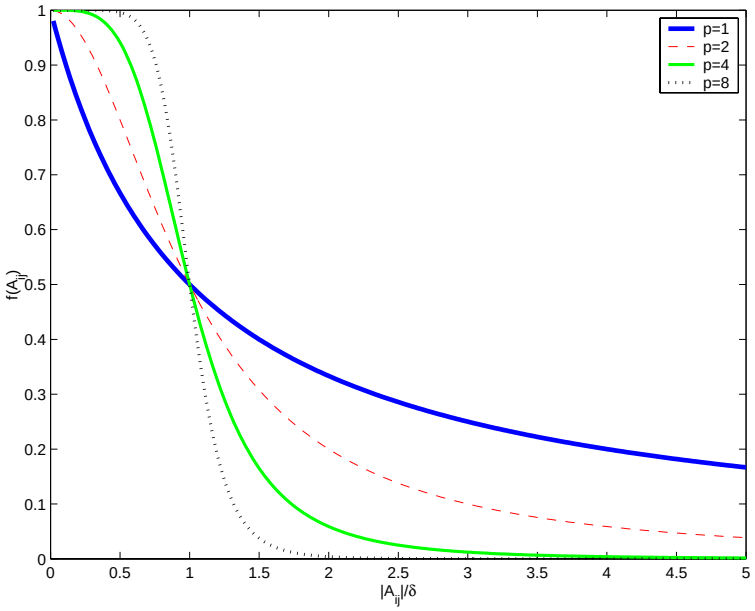

Figure 1: The plots of the weight function $f(A_{ij})$ vs $\frac{|A_{ij}|}{\delta}$ for various shape parameters p. Notice that scaling the factor δ amounts to shifting the transition between high weight and the low weight.

of the function, and how abrupt the weight transitions from 1 to 0. The plot of the function $f(A_{ij})$ can be seen in Figure 1. Throughout this paper, we use $\delta = 10^{-2}$ and $p = 1$.

The overall identification procedure can be summarized in the flowchart in Figure 2.

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 [7]. The network model captures the interaction between five genes and five transcription factors (see Figure 3).

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

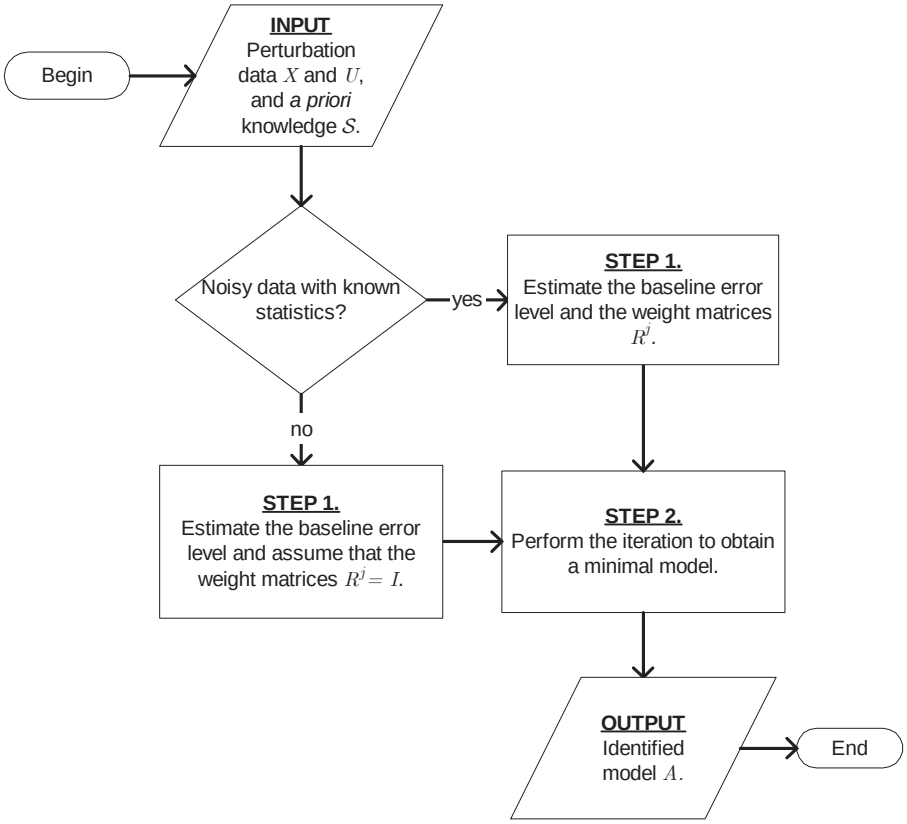

Figure 2: The flowchart summarizing the method proposed in this paper. The inputs to the algorithm are the perturbation data represented by the matrices X and U, as well as potential *a priori* structural knowledge about the network, which is represented by $\mathcal{S}$. The output of the algorithm is the identified model, represented by the matrix A.

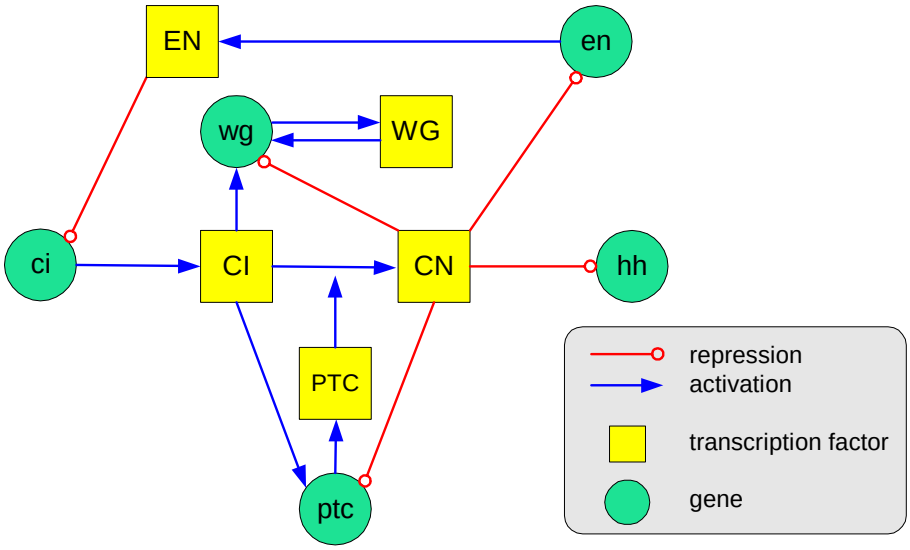

Figure 3: The segmentation polarity network of *Drosophila melanogaster* ([7]).

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 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 [7]. 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 (19). We do not estimate any error covariance matrices, and the performance measure is thus given by (13). 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_2 (see (18)).

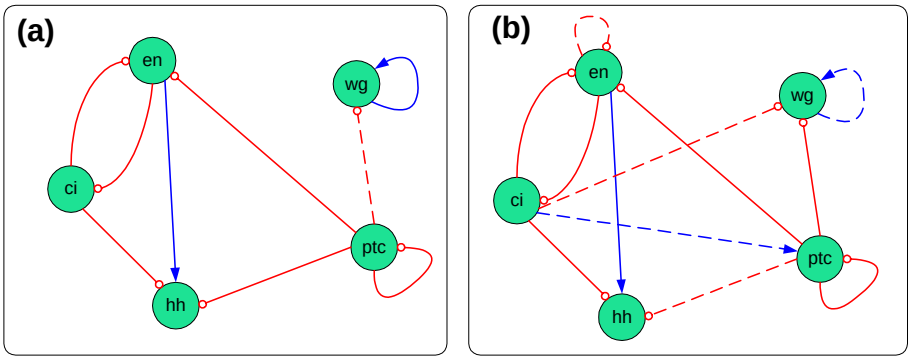

Figure 4: Identified network using *full* numerical data from the model of the segmentation polarity network of *Drosophila melanogaster*. Solid lines indicate strong interaction, while broken lines indicate weak interaction. **(a)** The network identified using the method proposed in this paper. **(b)** The network identified in [7].

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

$$\mu := \max_{i \neq j} \|A_{ij}\|. \tag{29}$$

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

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 4 shows the result of our network identification method and the network identified in [7]. 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 description of the network earlier in this section. The network in [7] 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*. These connections are 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 5b. Observe that the connections from *ptc* to other genes disappear as expected, since there is no data that dictates

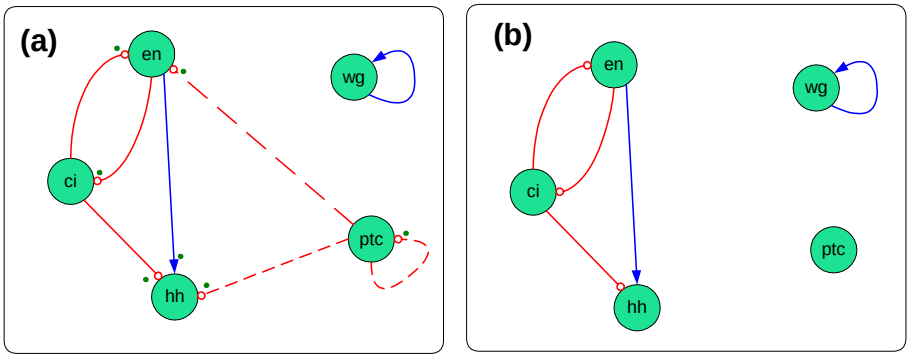

Figure 5: Identified network using *partial* numerical data from the model of the segmentation polarity network of *Drosophila melanogaster*. **(a)** The network identified when *a priori* knowledge about the sign pattern is incorporated. Arrows with green dots indicate interactions that are included in the *a priori* knowledge. **(b)** The network identified without incorporating any *a priori* knowledge.

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 5a. 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.

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 [6]. The subnetwork that we consider consists of nine genes and several transcription factors and metabolites (see Figure 2 in the Supplementary Material).

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 (*recA*, *ssb*, *dinI*, *umuDC*, and *rpoD*). 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 [6]. Since there is no explicit mentioning of U, we assume

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 1: A summary of *a priori* knowledge used in the network identification. The values in brackets represent known connections based on [6]. A + sign indicates known activation, − indicates known inhibition, 0 indicates the absence of connection, and ? indicates unknown connection.

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 model.

The covariance matrix of the measurement in X is obtained by processing the standard error matrix provided in [6]. Assuming that the measurement errors of different genes are uncorrelated, we can obtain

$$(\mathrm{Var}[X_j])_{ik} = \begin{cases} \sigma_{ij}^2, & i = k, \\ 0, & i \neq k, \end{cases} \tag{30}$$

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.

We compile the connections that are included in the *a priori* knowledge in Table 1. This list is compiled based on the diagram in Figure 6. 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 (14). 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 (14) as our error criterion. Instead, we use identity weight matrices. This approach turns out to be inferior to the one with estimated error covariance.

The result of our method is shown in Figure 7. In panel (a), we see the network identified using our method with estimated error covariance and $\beta = 1.75$. The network identified in [6] is shown

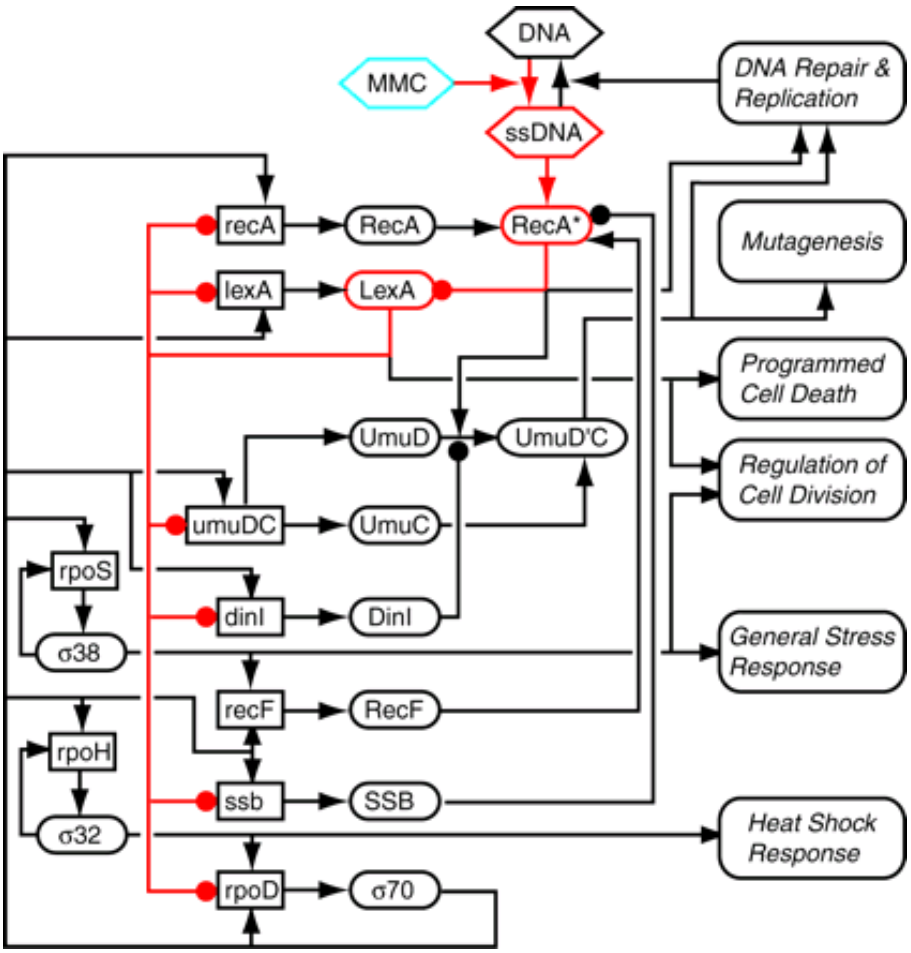

Figure 6: (Taken from [6]) Diagram of interactions in the SOS network. DNA lesions caused by mitomycin C (MMC) (blue hexagon) are converted to single-stranded DNA during chromosomal replication. Upon binding to ssDNA, the RecA protein is activated (RecA*) and serves as a coprotease for the LexA protein. 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.

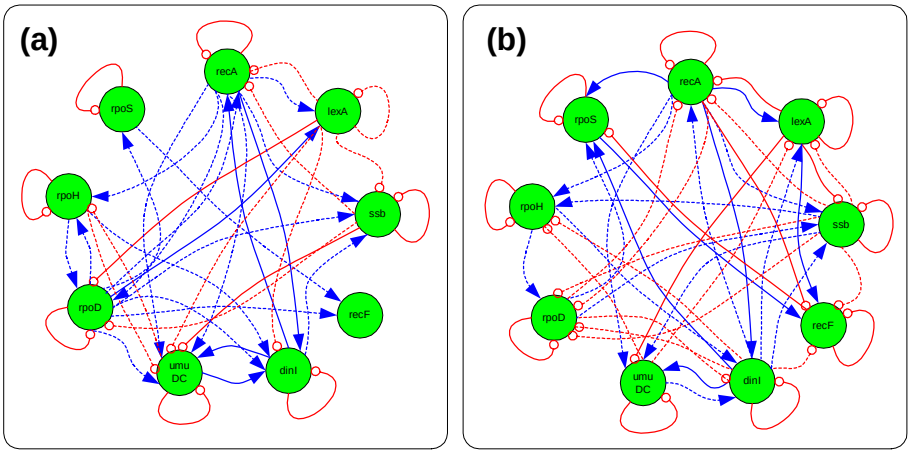

Figure 7: Identification results of the *Escherichia coli* SOS network. Lines with arrows indicate activation, while circles indicate repression. Solid lines denote strong interaction, while dotted lines denote weak interaction. The distinction between strong and weak connections follows the same convention as in the segmentation polarity network. **(a)** The result of our method using estimated error covariance and $\beta = 1.75$. **(b)** The network identified in [6].

in Figure 7 panel (b). Comparing it to our result in panel (a), we can see that the network in (b) misidentifies several known one-hop interconnections, such as the mutual repression between *recA* and *rpoD*.

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*, the activated form of recA (cf. [25]). Thereby, it effectively promotes the degradation of 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 other known interactions in the literature has been compiled by [6] and shown as the values between brackets in Table 1. We use this list as the "ground truth" and compare the results of the network identification methods with it.

The plot in Figure 8 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

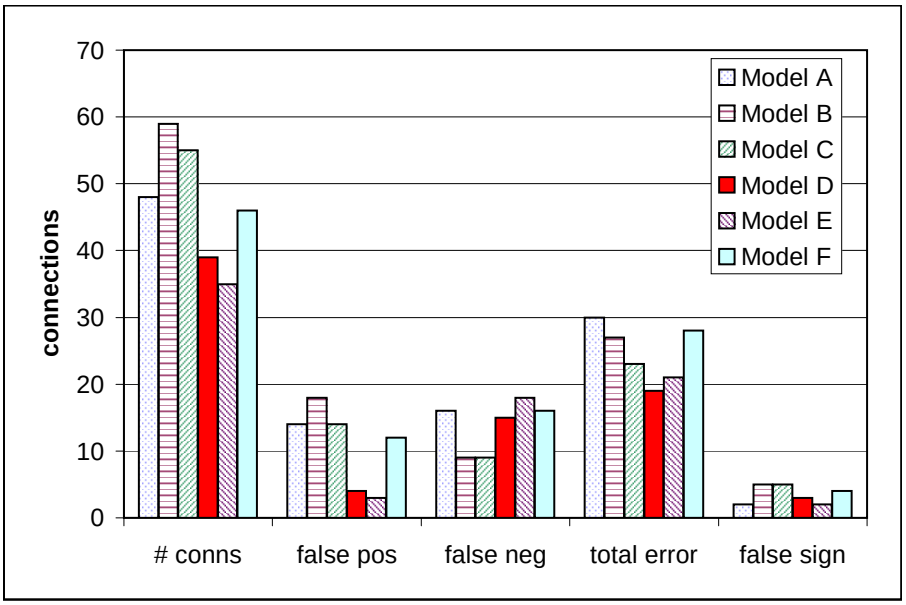

Figure 8: Performance comparison of various identified models of the *Escherichia coli* SOS network. Model A is the result of our method without using error covariance estimation and $\beta = 1.75$. Model B, C, D and E are the results of our method using error covariance estimation, with $\beta = 1.1$, 1.25, 1.75, and 2, respectively. Model D is shown in Figure 7 panel (a). Model F is the network identified in [6], as also shown in Figure 7 panel (b).

of the method. Comparing Model A and D, we can see that using the estimated error covariance gives us 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 7 panel (a).

3. The models identified using our method results in fewer errors compared to that in [6]. 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 Heterozigous knockdown of an *in silico* network

We generate an artificial network with 20 genes and study the performance of our method for various β parameters. As the measure of performance, we use the Receiver Operating Characteristic (ROC) curve. The ROC curve plots the sensitivity of the prediction results versus (1 - specificity). These quantities are given by the following formula (cf. [26]).

$$\text{Sensitivity} = \frac{\text{TP}}{\text{TP} + \text{FN}}, \ \text{Specificity} = \frac{\text{TN}}{\text{TN} + \text{FP}}, \tag{31}$$

where T='True', F='False', P='Positives', and N='Negatives'.

As the true system, we use the network shown in Figure 9. This network has a fan-like topology with a few regulator genes and a larger number of genes that are regulated directly or indirectly. The center of the network is formed by Gene 1 and Gene 2, that are interconnected in a mutual inhibition. As such, these two genes form a toggle switch [27], in which they can only be active complementarily. Gene 1 acts as an inhibitor to another pair of genes (Gene 3 and 4) that also form a toggle switch. This group of four genes thus form a staged toggle switch than can hold one of the following three states: (on,off,off,off), (off,on,off,on), and (off,on,on,off). The remaining sixteen genes in the network are regulated by these master genes directly or indirectly, as shown in Figure

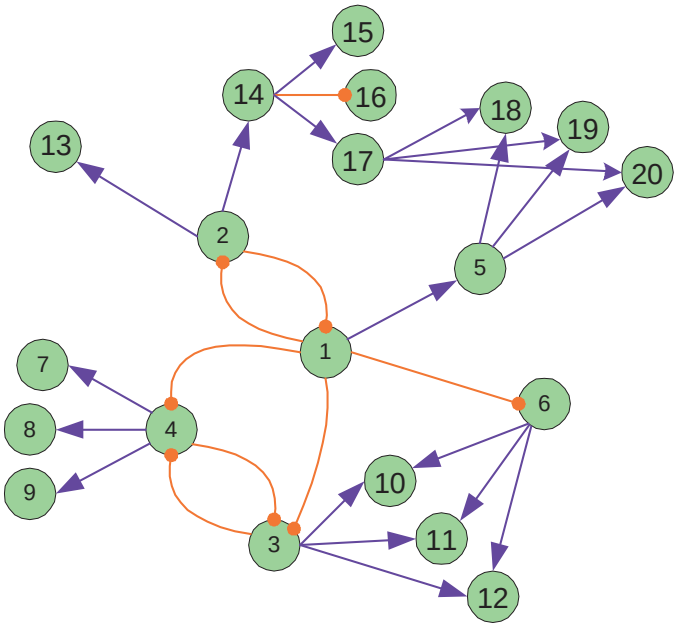

Figure 9: The *in silico* network used in Section 4.3.

9.

We assume that in the wild type specimen, each gene has two copies. Perturbation of the network is then performed by removing on the copies (heterozygous knockdown).

The dynamics of gene transcription and translation are modeled as a nonlinear system as follows:

$$\frac{dx_i}{dt} = \left(\Gamma_0 + \prod_{j\in \mathrm{Inh}_i} \frac{K^n}{K^n + y_j^n} \prod_{j\in \mathrm{Act}_i} \frac{y_j^n}{K^n + y_j^n}\right)\Gamma_i - \lambda_i x_i \tag{32}$$

$$\frac{dy_i}{dt} = x_i - \kappa_i y_i, \quad i \in \{1,\ldots,20\}, \tag{33}$$

where x_i is the transcript concentration of gene i, y_i is the concentration of protein i, Inh_i and Act_i are the set of genes that inhibit and activate gene i respectively. The variable Γ_i represent the availability of gene i. In the wild type, $\Gamma_{i,i\in\{1,\ldots,20\}} = 1$. When gene i is knocked down, $\Gamma_i = \frac{1}{2}$. The parameters of the model are Γ_0 (basal transcription rate), K (inhibition and activation treshold), n (Hill coefficient), λ_i (transcript decay rate), and κ_i (protein decay rate).

We generate the data that we use in the identification by computing the steady state values of the

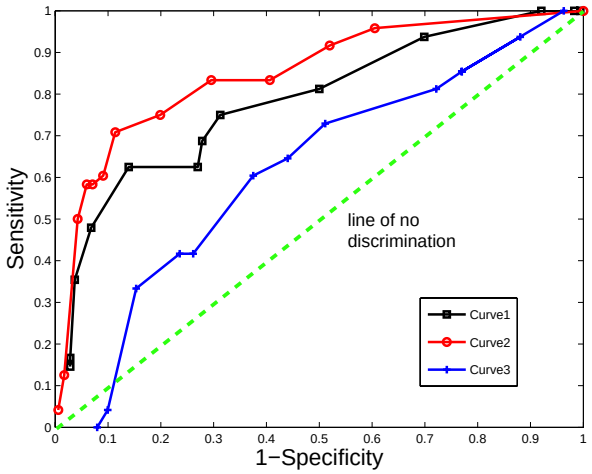

Figure 10: The ROC curves of various convex ℓ_1 relaxation techniques. Curve 1 is the result of non-weighted recursive algorithm, Curve 2 is the result of our algorithm, and Curve 3 is the result of the non-recursive regularization algorithm.

transcript concentrations (x) in (33), for the wild type case and for each of the perturbations. The perturbation input U is taken to be a negative identity matrix of size 20. To simulate noisy measurement, we add some zero-mean Gaussian noise with uniform standard deviation to the computed transcript concentrations.

As we have seen in the previous subsections, tuning the parameter β affects the performance of our algorithm. Since β represents a trade-off between sparsity and model completeness that cannot be known *a priori*, a fair assessment of the performance of the algorithm should be done by testing the algorithm for a wide range of choice of β. This is plotted in Figure 10.

In analyzing the performance of our algorithm we also compare our convex ℓ_1 relaxation technique with others that have been proposed in the literature.

Non-weighted recursive algorithm. This algorithm is based on the approach taken in [22, 23]. This algorithm does not use any weighting scheme in the ℓ_1 optimization. Instead, in each iteration of the sparsity optimization, entries that are small (less than δ) are constrained to be zero in the subsequent iterations. The recursion is performed until it converges. The value of δ can be tuned to adjust the sparsity of the result.

Non-recursive regularization algorithm. This algorithm based on the approach taken in, e.g. [19, 20, 21]. As suggested by the name, this algorithm does not involve any iteration. To obtain a sparse model, a term with the ℓ_1 norm of the identified model is added to the cost function. Therefore, instead of minimizing $\sum_{j=1,\ldots,m} \eta_j^T R^j \eta_j$ in (24), we minimize $\sum_{j=1,\ldots,m} \eta_j^T R^j \eta_j + w \cdot \sum_{i,j} |A_{i,j}|$. The weight w can be tuned to adjust the sparsity of the result. Specifically, larger w means higher priority on the sparsity of the model, and thus results in sparser model.

The performance comparison between our method and these two methods are plotted in Figure 10. We can see that for this example, our algorithm performs better than the other two relaxation techniques.

5 Conclusion

We propose a method for identifying genetic regulatory networks using expression profiles from genetic perturbation experiments. Some of the features of our method 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 *a priori* information about the structure of the network. Third, we take into account the statistics of the measurement noise in formulating the cost function of our identification optimization.

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] Bonneau R, Reiss DJ, Shannon P, Facciotti M, Hood L, Baliga NS, et al. The inferelator: an algorithm for learning parsimonious regulatory networks from systems-biology data sets *de novo*. Genome Biology. 2006;7:R36.

[2] Hayete B, Gardner TS, Collins JJ. Size matters: network inference tackles the genome scale. Molecular Systems Biology. 2007;3(10.1038/msb4100118).

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

[4] Faith JJ, Hayete B, Thaden JT, Mogno I, Wierzbowski J, Cottarel G, et al. Large-scale mapping and validation of *Escherichia coli* transcriptional regulation from a compendium of expression profiles. PLoS Biology. 2007;5(1):e8.

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

[6] Gardner TS, di Bernardo D, Lorenz D, Collins JJ. Inferring genetic networks and identifying compound mode of action via expression profiling. Science. 2003;301:102–105.

[7] Tegner J, Yeung MKS, Hasty J, Collins JJ. Reverse engineering gene networks: integrating genetic perturbations with dynamical modeling. Proc of the National Academy of Science. 2003;100(10):5944–5949.

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

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

[10] Hoffman AJ, McCormick ST. A fast algorithm that makes matrices optimally sparse. In: Pulleybank WR, editor. Progress in Combinatorial Optimization. Academic Press; 1984. p. 185–196.

[11] Yeung MKS, Tegner J, Collins JJ. Reverse engineering gene networks using singular value decomposition and robust regression. Proc of the National Academy of Science. 2002;99(9):6163–6168.

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

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

[14] Boyd S, Vandenberghe L. Convex Optimization. Cambridge University Press; 2004. Available online at www.stanford.edu/ boyd/cvxbook/.

[15] Boyd S. ℓ_1-norm norm methods for convex cardinality problems; 2007. Available online at www.stanford.edu/class/ee364b/. Lecture Notes for EE364b, Stanford University.

[16] Candes EJ, Wakin MB, Boyd S. Enhancing sparsity by reweighted ℓ_1 minimization. Journal of Fourier Analysis and Applications. 2008;14(5):877–905.

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

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

[19] Li F, Yang Y. Recovering genetic regulatory networks from micro-array data and location analysis data. Genome Informatics. 2004;15(2):131–140.

[20] Han S, Yoon Y, Cho KH. Inferring biomolecular interaction networks based on convex optimization. Computational Biology and Chemistry. 2007;31(5-6):347–354.

[21] Guo Y, Schuurmans D. Learning gene regulatory networks via globally regularized risk minimization. In: Comparative Genomics. Berlin: Springer Verlag; 2007. p. 83–95.

[22] Papachristodoulou A, Recht B. Determining Interconnections in Chemical Reaction Networks. In: Proc. American Control Conference. New York, USA; 2007. p. 4872 – 4877.

[23] Cosentino C, Curatola W, Montefusco F, Bansal M, di Bernardo D, Amato F. Linear matrix inequalities approach to reconstruction of biological networks. IET Systems Biology. 2007;1(3):164–173.

[24] Boyd S, Grant MC. cvx – MATLAB software for disciplined convex programming; 2005. Http://www.stanford.edu/∼boyd/cvx/.

[25] Lusetti SL, Voloshin ON, Inman RB, Camerini-Otero RD, Cox MM. The DinI Protein Stabilizes RecA Protein Filaments. Journal of Biological Chemistry. 2004;279(29):30037–30046.

[26] De Muth JE. Basic Statistics and Pharmaceutical Statistical Applications. 2nd ed. Chapman & Hall/CRC; 2006.

[27] Gardner TS, Cantor CR, Collins JJ. Construction of a genetic toggle switch in Escherichia coli. Nature. 2000;403:339–342.