Published in IET Systems Biology
Received on 1st September 2007
doi: 10.1049/iet-syb.2008.0130

ISSN 1751-8849

Genetic network identification using convex programming

A. Julius[1] M. Zavlanos[1] S. Boyd[2] G.J. Pappas[1]

[1]Department of Electrical and Systems Engineering, University of Pennsylvania, PA, USA
[2]Department of Electrical Engineering, Stanford University, CA, USA
E-mail: agung@seas.upenn.edu

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. The authors 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. The method is based on a convex programming relaxation of the combinatorially hard problem of L_0 minimisation. The authors 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.

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 [1–5] and the references therein. On the basis of 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 (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 [6–9]. The method that we propose in this paper belongs to this class.

On the basis of 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 equilibium. 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 and coworkers [6, 7, 11] provided an important step towards addressing this problem. In [11], the authors propose a method for noiseless measurements, where the optimisation is relaxed as a non-recursive ℓ_1 optimisation 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 minimisation 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 optimisation [14] after relaxing it as a recursive ℓ_1 optimisation problem [15]. The same technique has been applied successfully in various fields where sparsity optimisation is needed such as, portfolio optimisation in finance [16] and controller design in engineering [17]. Different techniques of convex ℓ_1 relaxation have been used in other works in reverse engineering of gene networks, such as [18–22]. Posing the problem as a convex optimisation problem is actually very advantageous from the complexity point of view, as convex optimisation 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* [16] 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 modelling and identification

2.1 Notations

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

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

$$\frac{\mathrm{d}\hat{\boldsymbol{x}}}{\mathrm{d}t} = F(\hat{\boldsymbol{x}}, \hat{\boldsymbol{y}}, \hat{\boldsymbol{u}}),\ \hat{\boldsymbol{x}} \in \mathbb{R}^n,\ \hat{\boldsymbol{y}} \in \mathbb{R}^n,\ \hat{\boldsymbol{u}} \in \mathbb{R}^p \qquad (1)$$

$$\frac{\mathrm{d}\hat{\boldsymbol{y}}}{\mathrm{d}t} = G(\hat{\boldsymbol{x}}, \hat{\boldsymbol{y}}) \qquad (2)$$

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 summarise 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 $(\boldsymbol{x}_{\mathrm{eq}}, \boldsymbol{y}_{\mathrm{eq}})$ can be approximated by the set of linear differential equations

$$\frac{\mathrm{d}\boldsymbol{x}}{\mathrm{d}t} = \mathfrak{A}_{11}\boldsymbol{x} + \mathfrak{A}_{12}\boldsymbol{y} + \boldsymbol{u} \qquad (3)$$

$$\frac{\mathrm{d}\boldsymbol{y}}{\mathrm{d}t} = \mathfrak{A}_{21}\boldsymbol{x} + \mathfrak{A}_{22}\boldsymbol{y} \qquad (4)$$

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

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

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

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 $\boldsymbol{y}$. We therefore obtain

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

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

We then define the matrix

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

The matrix $\boldsymbol{A} \in \mathbb{R}^{n\times n}$ encodes the effective pairwise interactions between the individual genes in the network. In this representation, we have the relation

$$\boldsymbol{A}\boldsymbol{x} + \boldsymbol{u} = 0 \tag{10}$$

Let $\boldsymbol{U} = [U_1 \ \dots \ U_m] \in \mathbb{R}^{p\times m}$ denote the stack matrix of the transcription perturbations for different m experiments and $\boldsymbol{X} = [X_1 \ \dots \ X_m] \in \mathbb{R}^{n\times m}$ 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

$$\boldsymbol{A}\boldsymbol{X} + \boldsymbol{U} = 0 \tag{11}$$

In (11), the matrices $\boldsymbol{X}$ and $\boldsymbol{U}$ are known, since they are measured (possibly with noise). The goal of the method we propose in this paper is to find unknown matrix $\boldsymbol{A}$, 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. Here 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 $\boldsymbol{A}$. 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, that is, the number of non-zeros entries in the matrix $\boldsymbol{A}$. This is known as the L_0 norm of the matrix $\boldsymbol{A}$. (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 $\boldsymbol{A}$.

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 (11), if we assume that the measurements are noise free and if we have a sufficient number $m = n$ of independent experiments, $\boldsymbol{X}$ is an invertible matrix, and we could obtain $\boldsymbol{A}$ using $\boldsymbol{A} = -\boldsymbol{U}\boldsymbol{X}^{-1}$. 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

$$\eta := \boldsymbol{A}\boldsymbol{X} + \boldsymbol{U} \tag{12}$$

and try to minimise η (as a function of $\boldsymbol{A}$) with respect to some metric, while obtaining a minimal model for $\boldsymbol{A}$ and satisfying the a priori constraints that might be imposed on $\boldsymbol{A}$.

3.1 Error criterion

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

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

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

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

where $\boldsymbol{R}^j$ is the inverse of the error covariance matrix in the jth 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.

3.2 Model minimality criterion

We define the size of a model given by the matrix $\boldsymbol{A}$ as the number of non-zero entries in $\boldsymbol{A}$, which is denoted by $\|\boldsymbol{A}\|_0$. This is the number of connections in the model.

3.3 Priori knowledge constraint

Such knowledge typically has the form of (partial) sign pattern of the matrix $\boldsymbol{A}$. We encode this by a matrix $\boldsymbol{S} \in \{0, +, -, ?\}^{n\times n}$, 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 $\boldsymbol{A}$ that satisfy a given a priori knowledge constraint $\boldsymbol{S}$ as $\mathcal{S}$. A critical property of the set $\mathcal{S}$ that can be easily shown is that it is convex [14].

3.4 Bias due to mRNA decay

As defined in (9), the effective gene–gene network model $\boldsymbol{A}$ 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

$$-\mathfrak{A}_{22}^{-1}\mathfrak{A}_{21} =: \boldsymbol{\Lambda}_1 \tag{16}$$

where $\boldsymbol{\Lambda}_1$ 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

$$\mathfrak{A}_{11} =: -\boldsymbol{\Lambda}_2 \tag{17}$$

where $\boldsymbol{\Lambda}_2$ is another diagonal matrix with positive entries. We then have

$$\boldsymbol{A} = \boldsymbol{\Lambda}_1\mathfrak{A}_{12} - \boldsymbol{\Lambda}_2 \tag{18}$$

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 $\boldsymbol{\Lambda}_2$ 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 autoregulation of genes.

3.5 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 minimisation 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, example, [14].

3.6 Convex optimisation solver

The convex optimisation problems that we pose in this paper are solved using MATLAB with the toolbox cvx [23] running on an Intel Xeon 2.8 Ghz processor with 4 GB RAM. Here cvx makes forming and solving the problem easy, but at the cost of efficiency. However, custom-made implementations of convex optimisation 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.

3.6.1 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

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

with optimisation variable $\boldsymbol{A}$.

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

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

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

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

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

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

where $\boldsymbol{A}$ 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 minimises the error criterion (19). Denote this model as $\tilde{\boldsymbol{A}}$. The weight matrices $\boldsymbol{R}^j$ are then given by

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

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

$$\begin{array}{ll} \text{minimise} & \sum_{j=1,\dots,m} \eta_j^{\mathrm{T}} \boldsymbol{R}^j \eta_j \\ \text{subject to} & \eta = \boldsymbol{A}\boldsymbol{X} + \boldsymbol{U},\ \boldsymbol{A} \in \mathcal{S} \end{array} \tag{24}$$

with $\boldsymbol{A}$ as the variable.

3.6.2 Step 2 – minimising 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 the β, 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 trade-off between model accuracy and model minimality.

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

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

where $\boldsymbol{A}$ is the variable. We denote the solution of this problem as $\boldsymbol{A}_{\min}$. 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 on a large scale. In order to solve this problem with convex programming, we relax the L_0 minimisation problem as a recursive weighted ℓ_1 minimisation.

Step 2.1: Initiate $\boldsymbol{A}_{\text{old}} = 0$ and $W_{ij} = 1, i = 1, 2, \dots, n$, $j = 1, 2, \dots, n$

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

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

where $\boldsymbol{A}$ 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 $\|\boldsymbol{A}_{\text{old}} - \boldsymbol{A}_{\text{update}}\|_F \geq \varepsilon$, then update $\boldsymbol{A}_{\text{old}} = \boldsymbol{A}_{\text{update}}$ and go to step 2.2, otherwise stop the iteration and return the current solution as the optimal value

$$\boldsymbol{A}_{\min} = \boldsymbol{A}_{\text{update}} \tag{27}$$

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

3.7 Choosing the weight function

The weight function $f(A_{ij})$ is designed such that the entries of $\boldsymbol{A}$ that are small are given larger weight than the larger entries [15]. This is because we are interested in maximising the number of zero entries in $\boldsymbol{A}$. We thus emphasise 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 as 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 of the function, and how abruptly the weight transitions take place from 1 to 0. The plot of the function $f(A_{ij})$ can be seen in Fig. 1. Throughout this paper, we use $\delta = 10^{-2}$ and $p = 1$.

The overall identification procedure can be summarised in the flowchart in Fig. 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

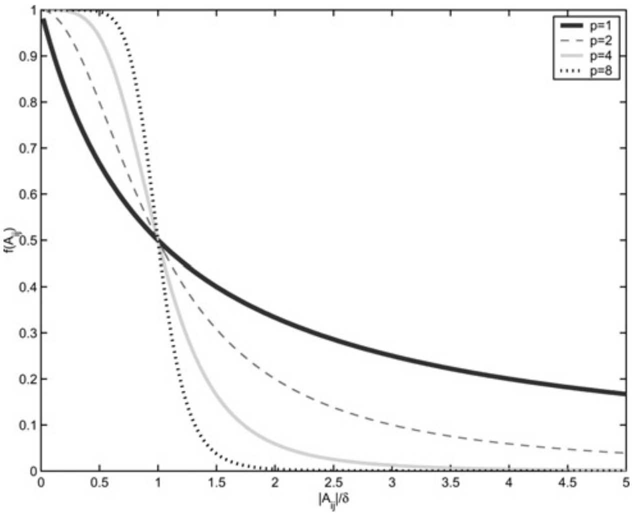

Figure 1 *The plots of the weight function $f\ (A_{ij})$ against $|A_{ij}|/\delta$ for various shape parameters p*

Notice that scaling the factor δ amounts to shifting the transition between high weight and the low weight

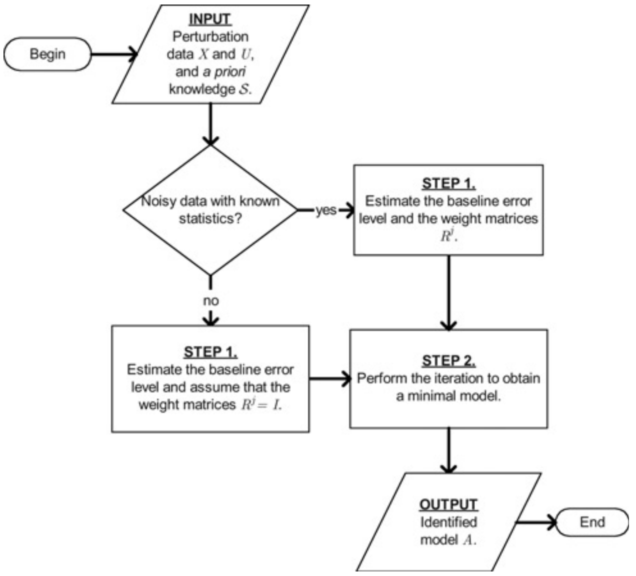
Begin
INPUT
Perturbation data X and U, and a priori knowledge S.
Noisy data with known statistics?
yes
no
STEP 1.
Estimate the baseline error level and the weight matrices R^j.
STEP 1.
Estimate the baseline error level and assume that the weight matrices R^j = I.
STEP 2.
Perform the iteration to obtain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

Figure 2 *The flowchart summarising the method proposed in this paper*

The inputs to the algorithm are the perturbation data represented by the matrices $\boldsymbol{X}$ and $\boldsymbol{U}$, 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 $\boldsymbol{A}$

given in [7]. The network model captures the interaction between five genes and five transcription factors (Fig. 3).

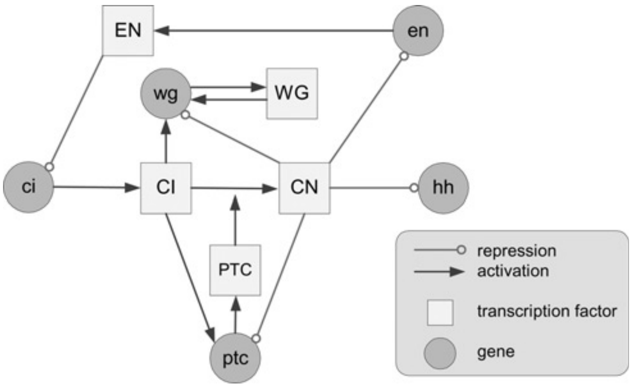

Figure 3 *The segmentation polarity network of Drosophila melanogaster [7]*

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 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 $\boldsymbol{X}$ and $\boldsymbol{U}$ 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, $\boldsymbol{U} = 10^{-3} \cdot \boldsymbol{I}$, where $\boldsymbol{I}$ is an identity matrix. The deviations from the unperturbed equilibrium are recorded in the $\boldsymbol{X}$ 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 $\boldsymbol{\Lambda}_2$ (18).

Given the matrix $\boldsymbol{A}$ 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 s on the platform that is detailed in the previous section. Fig. 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

doi: 10.1049/iet-syb.2008.0130

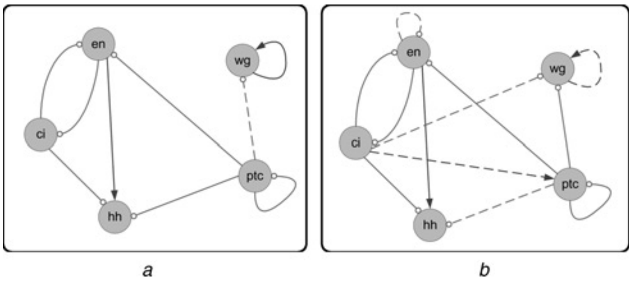

Figure 4 *Identified network using* full *numerical data from the model of the segmentation polarity network of* Drosophila melanogaster

a The network identified using the method proposed in this paper
b The network identified in [7]
Solid lines indicate strong interaction, while broken lines indicate weak interaction

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 Fig. 5*b*. 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 Fig. 5*a*. 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.

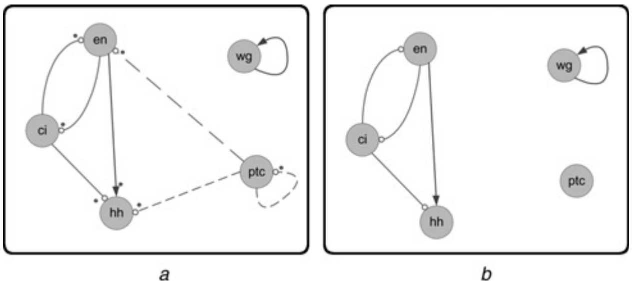

Figure 5 *Identified network using* partial *numerical data from the model of the segmentation polarity network of* Drosophila melanogaster

a The network identified when a priori knowledge about the sign pattern is incorporated
Arrows with black 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 [6]. The subnetwork that we consider consists of nine genes and several transcription factors and metabolites (see Fig. 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 SSDNA, cleaves LexA and thus upregulates the above mentioned genes. Other key regulators in the network are the sigma factors σ70, σ32 and σ38. These sigma factors play an important role in initiating transcription in heat shock and starvation responses.

We obtain the perturbation data $\boldsymbol{X}$ from [6]. Since there is no explicit mentioning of $\boldsymbol{U}$, we assume that it is an identity matrix. Notice that this is a justifiable assumption, as a different value of $\boldsymbol{U}$ would just result in a scaling of the model.

The covariance matrix of the measurement in $\boldsymbol{X}$ 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 $\boldsymbol{U}$, 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 Fig. 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 $\boldsymbol{R}^j$ that are used in the error criterion (14). We then proceed to Step 2, and use different β values obtain different models and analyse them (as detailed below).

As a 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 Fig. 7. In panel (a), we see the network identified using our method with estimated error covariance and $\beta = 1.75$. The network

Table 1 A summary of a priori knowledge used in the network identification

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)	?

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

identified in [6] is shown in Fig. 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*.

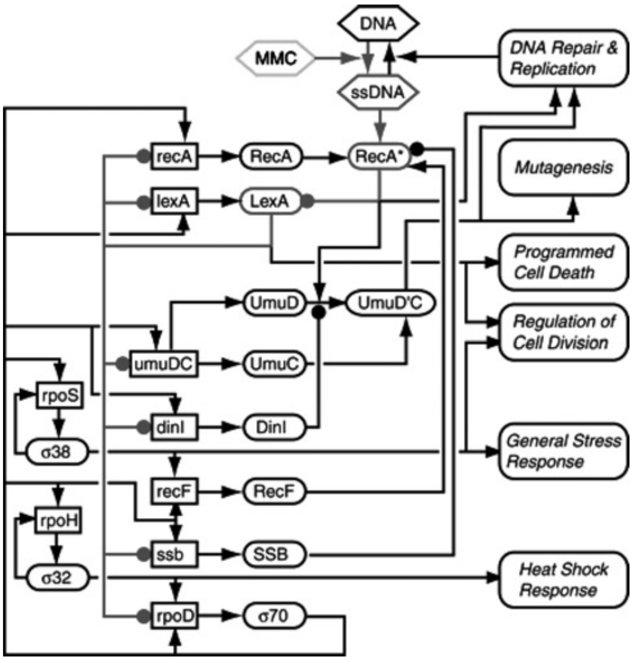

Figure 6 (*Taken from [6]) The diagram of interactions in the SOS network*

DNA lesions caused by mitomycin C (MMC) (light grey 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 and denote proteins, hexagons indicate metabolites, arrows denote positive regulation, filled circles denote negative regulation. Dark grey emphasis and denotes the primary pathway by which the network is activated after DNA damage

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 stabilise recA*, the activated form of *recA* [24]. 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 Fig. 8 shows a performance comparison of various identified models. Several observations that can be made from the comparison are as follows:

1. Incorporating the estimated error covariance in the error assessment improves the performance of the method. Comparing Models 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 emphasise 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 Fig. 7 panel (a).

3. The models identified using our method results in fewer errors compared to that in [6]. In particular, by tuning the

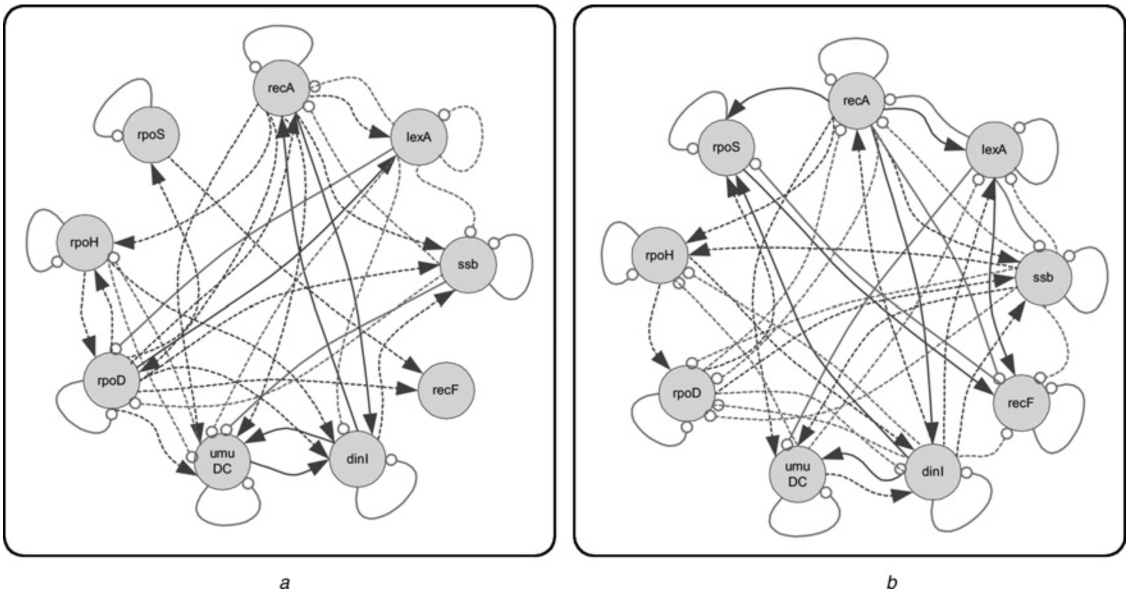

Figure 7 *Identification results of the* Escherichia coli *SOS network*

a The result of our method using estimated error covariance and $\beta = 1.75$
b The network identified in [6]
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

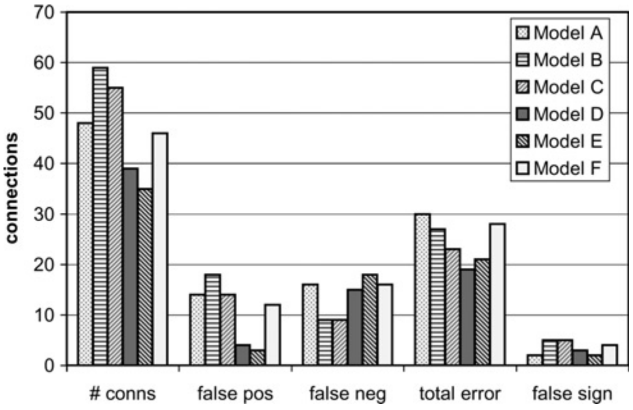

Figure 8 *Performance comparison of various identified models of the* Escherichia coli *SOS network*

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 Fig. 7 panel (a)
Model F is the network identified in [6], as also shown in Fig. 7 panel (b)

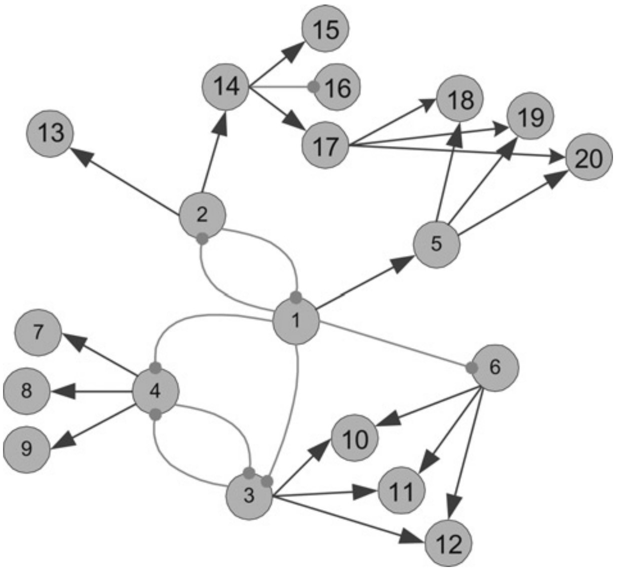

Figure 9 *The in silico network used in Section 4.3*

value of β carefully, we can obtain models with very low number of false positives (less than 5%).

4.3 *Heterozygous 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 against (1 − specificity). These quantities are given by the following formula [25]

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

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

As the true system, we use the network shown in Fig. 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 Genes 1 and 2, that are interconnected in a mutual inhibition. As such, these two genes form a toggle switch [26], in which they can only be active complementarily. Gene 1 acts as an inhibitor to another pair of genes (Genes 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 16 genes in the network are regulated by these master genes directly or indirectly, as shown in Fig. 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 modelled as a nonlinear system as follows

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

$$\frac{\mathrm{d}y_i}{\mathrm{d}t} = x_i - \kappa_i y_i, \quad i \in \{1, \ldots, 20\} \qquad (33)$$

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, $\Gamma_i = 1/2$. 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 transcript concentrations (x) in (33), for the wild-type case and for each of the perturbations. The perturbation input $\boldsymbol{U}$ 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 sections, 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 Fig. 10.

4.3.1 Non-weighted recursive algorithm: This algorithm is based on the approach taken in [21, 22]. This algorithm does not use any weighting scheme in the ℓ_1 optimisation. Instead, in each iteration of the sparsity optimisation, entries that are small ($<\delta$) 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.

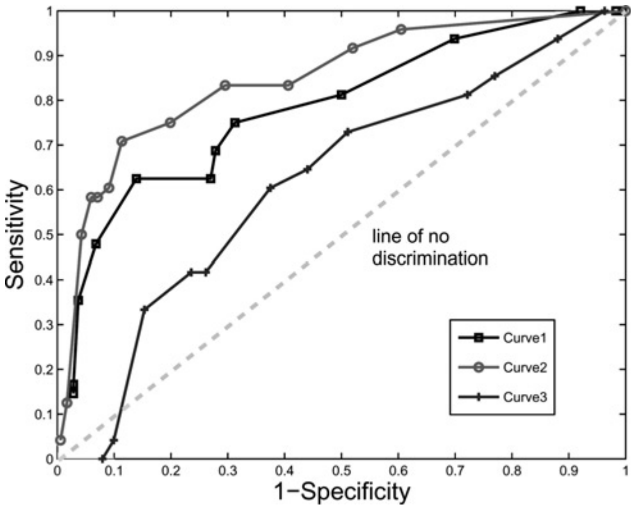

Figure 10 *The ROC curves of various convex ℓ_1 relaxation techniques*

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 regularisation algorithm

4.3.2 Non-recursive regularisation algorithm: This algorithm based on the approach taken in, for example [18–20]. 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 minimising $\sum_{j=1,\ldots,m} \eta_j^{\mathrm{T}} \boldsymbol{R}^j \eta_j$ in (24), we minimise $\sum_{j=1,\ldots,m} \eta_j^{\mathrm{T}} \boldsymbol{R}^j \eta_j + w \cdot \sum_{i,j} |A_{i,j}|$. 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 a sparser model.

The performance comparison between our method and these two methods are plotted in Fig. 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 (characterised 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 optimisation.

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 customised implementation of convex optimisation 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.

6 Acknowledgment

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, the NSF award ECS-0423905, the NSF award 0529426, and the NASA award NNX07AEIIA.

7 References

[1] BONNEAU R., REISS D.J., SHANNON P., *ET AL.*: 'The inferelator: an algorithm for learning parsimonious regulatory networks from systems-biology data sets *de novo*', *Genome Biol.*, 2006, **7**, p. R36

[2] HAYETE B., GARDNER T.S., COLLINS J.J.: 'Size matters: network inference tackles the genome scale', *Mole. Syst. Biol.*, 2007, **3**, article no. 77

[3] GEIER F., TIMMER J., FLECK C.: 'Reconstructing gene-regulatory networks from time-series, knockout data and prior knowledge', *BMC Syst. Biol.*, 2007, **1**, (11), article no. 11

[4] FAITH J.J., HAYETE B., THADEN J.T., *ET AL.*: 'Large-scale mapping and validation of *Escherichia coli* transcriptional regulation from a compendium of expression profiles', *PLoS Biol.*, 2007, **5**, (1), p. e8

[5] BANSAL M., BELCASTRO V., AMBESI-IMPIOMBATO A., DI BERNARDO D.: 'How to infer gene networks from expression profiles', *Mole. Syst. Biol.*, 2007, **3**, article no. 78

[6] GARDNER T.S., DI BERNARDO D., LORENZ D., COLLINS J.J.: 'Inferring genetic networks and identifying compound mode of action via expression profiling', *Science*, 2003, **301**, pp. 102–105

[7] TEGNER J., YEUNG M.K.S., HASTY J., COLLINS J.J.: 'Reverse engineering gene networks: integrating genetic perturbations with dynamical modeling', *Proc. Natl. Acad. Sci.*, 2003, **100**, (10), pp. 5944–5949

[8] SONTAG E., KIYATKIN A., KHOLODENKO B.N.: 'Inferring dynamic architecture of cellular networks using time series of gene expression, protein and metabolite data', *Bioinformatics*, 2004, **20**, (12), pp. 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), pp. 815–822

[10] HOFFMAN A.J., MCCORMICK S.T.: 'A fast algorithm that makes matrices optimally sparse', in PULLEYBANK W.R. (ED.): Progress in combinatorial optimization, (Academic Press, 1984), pp. 185–196

[11] YEUNG M.K.S., TEGNER J., COLLINS J.J.: 'Reverse engineering gene networks using singular value decomposition and robust regression', *Proc. Natl. Acad. Sci.*, 2002, **99**, (9), pp. 6163–6168

[12] THOMAS R., MEHROTRA S., PAPOUTSAKIS E.T., HATZIMANIKATIS V.: 'A model-based optimization frame- work for the inference on gene regulatory networks from DNA array data', *Bioinformatics*, 2004, **20**, (17), pp. 3221–3235

[13] ANDREC M., KHOLODENKO B.N., LEVY R.M., SONTAG E.: 'Inference of signaling and gene regulatory networks by steady-state perturbation experiments: structure and accuracy', *J. Theor. Biol.*, 2005, **232**, (3), pp. 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] LOBO M.S., FAZEL M., BOYD S.: 'Portfolio optimization with linear and fixed transaction costs', *Ann. Oper. Res.*, 2007, **152**, (1), pp. 376–394

[17] HASSIBI A., HOW J., BOYD S.: 'Low-authority controller design via convex optimization', *AIAA J. Guid. Control Dyn.*, 1999, **22**, (6), pp. 862–872

[18] LI F., YANG Y.: 'Recovering genetic regulatory networks from micro-array data and location analysis data', *Genome Inf.*, 2004, **15**, (2), pp. 131–140

[19] HAN S., YOON Y., CHO K.H.: 'Inferring biomolecular interaction networks based on convex optimization', *Comput. Biol. Chem.*, 2007, **31**, (5–6), pp. 347–354

[20] GUO Y., SCHUURMANS D.: 'Learning gene regulatory networks via globally regularized risk minimization', 'Comparative genomics' (Springer Verlag, Berlin, 2007), pp. 83–95

doi: 10.1049/iet-syb.2008.0130

[21] PAPACHRISTODOULOU A., RECHT B.: 'Determining Interconnections in chemical reaction networks'. Proc. American Control Conf., New York, USA, 2007, pp. 4872–4877

[22] COSENTINO C., CURATOLA W., MONTEFUSCO F., BANSAL M., DI BERNARDO D., AMATO F.: 'Linear matrix inequalities approach to reconstruction of biological networks', *IET. Syst. Biol.*, 2007, **1**, (3), pp. 164–173

[23] BOYD S., GRANT M.C.: 'cvx – MATLAB software for disciplined convex programming', 2005, http://www.stanford.edu/∼boyd/cvx/

[24] LUSETTI S.L., VOLOSHIN O.N., INMAN R.B., CAMERINI-OTERO R.D., COX M.M.: 'The DinI Protein Stabilizes RecA Protein Filaments', *J. Biol. Chem.*, 2004, **279**, (29), pp. 30037–30046

[25] DE MUTH J.E.: 'Basic statistics and pharmaceutical statistical applications' (Chapman & Hall/CRC, 2006, 2nd edn.)

[26] GARDNER T.S., CANTOR C.R., COLLINS J.J.: 'Construction of a genetic toggle switch in Escherichia coli', *Nature*, 2000, **403**, pp. 339–342

doi: 10.1049/iet-syb.2008.0130