Chapter 1

Control Engineering Approaches to Reverse Engineering Biomolecular Networks

The last decade has witnessed a tremendous griwth in interdisciplinary research on the application of systems and control engineering techniques to biological problems. A fundamental challenge in this field is the development of appropriate modelling and simulation frameworks for biological networks at the molecular level: gene regulatory, metabolic, signal transduction and protein–protein interaction networks provide a rich field of application for mathematicians, engineers and computer scientists. The reason for this renewed appeal can be largely ascribed to recent breakthroughs in the field of biotechnology, such as cDNA microarrays and oligonucleotide chips [1, 2], which have made high-throughput and quantitative experimental measurements of biological systems much easier and cheaper to make. The availability of such an overwhelming amount of data, however, poses a new challenge: how to reverse engineer the biological systems (especially at the molecular level) starting from the measured response to external perturbations (e.g. drugs, signalling molecules, pathogens) and changes in the environmental conditions (e.g. change in the concentration of nutrients or in the temperature level). In this chapter, we provide an overview of several promising techniques, based on dynamical systems identification theory, for reverse engineering the topology of biomolecular interaction networks from this kind of experimental data.

1.1 Dynamical models for network inference

A standard approach to model the dynamics of biomolecular interaction networks is by means of a system of ordinary differential equations (ODEs) that describes the temporal evolution of the various compounds [3, 4]. Typically, the network is modeled as a system of rate equations in the form

$$\dot{x_i}(t) = f_i(x(t), p(t), u(t)), \tag{1.1}$$

for $i = 1, \ldots, n$ with $x = (x_1, \ldots, x_n)^T \in \mathbb{R}^n$, where the state variables x_i denote the quantities of the different compounds present in the system (e.g. mRNA, proteins, metabolites) at time t, f_i is the function that describes the rate of change of the state variable x_i and its dependence on the other state variables, p is the parameter set and u is the vector of external perturbation signals.

The level of detail and the complexity of these kinetic models can be adjusted, through the choice of the rate functions f_i, by using more or less detailed kinetics, i.e. specific forms of f_i (linear or specific types of nonlinear functions). Moreover, it is possible to adopt a more or less simplified set of entities and reactions, e.g. choosing whether to take into account mRNA and protein degradation, delays for transcription, anf translation or diffusion time [3].

The use of systems of ODEs as a modelling framework for biological networks presents several challenges to control engineering approaches to network reconstruction, which typically stem from classical system identification procedures. The problem is typically tackled in two steps: the first step is to determine the model structure, i.e. the mathematical form of f_i, that is most appropriate to describe the experimental dynamics; the second step aims to compute the values of the parameters that yield the best fit of the experimental data. The two steps are strictly interconnected: the number and type of parameters is defined by the model structure; on the other hand, the interpolation of the experimental data provides important hints about how to modify the model structure to get better results.

When the order of the system increases, nonlinear ODE models quickly become intractable in terms of parametric analysis, numerical simulation and especially for identification purposes. If the nonlinear functions f_i are allowed to take any form, indeed, determination of the network topology becomes impossible. A more sensible approach, therefore, is to use equations composed of as few mathematical terms as possible. Even assuming that the model structure if perfectly known, each equation in the model requires knowledge of one or more parameter values (thermodynamic constants, rate constants), which are difficult to estimate using current data production techniques. This fact, along with the low number of measurements, typically renders the ODE system not uniquely identifiable from the data at hand.

Due to the above issues, although biomolecular networks are characterized by complex nonlinear dynamics, many network inference approaches are based on linear models or are limited to very specific types of nonlinear functions. In what follows, we will illustrate some of the most significant advances that have been achieved in the development of effective network reconstruction methods based on dynamical systems identification. As will become clear, the reverse engineering methods are closely related to the choice of model structure. Therefore, before illustrating the reverse engineering methods, we briefly introduce the most common model structures which may be chosen within this identification framework.

1.1.1 Linear models

The dynamical evolution of a biological network can be described, at least for small excursions of the relevant quantities from the equilibrium point, by means of linear systems, made up of ODEs in the continuous–time case, or difference equations in the discrete–time case (see [5, 6, 7, 8, 9, 10] and references therein).

We consider the continuous–time LTI model

$$\dot{x}(t) = Ax(t) + Bu(t)\,, \tag{1.2}$$

where $x(t) = (x_1(t), \dots, x_n(t))^T \in \mathbb{R}^n$, the state variables $x_i,\ i = 1, \dots, n$, denote the quantities of the different compounds present in the system (e.g. mRNA concentrations for gene expression levels), $A \in \mathbb{R}^{n\times n}$ is the dynamic matrix and $B \in \mathbb{R}^{n\times 1}$ is a vector that determines the direct targets of external perturbations $u(t) \in \mathbb{R}$ (e.g. drugs, overexpression or downregulation of specific genes), which are typically induced during *in vitro* experiments.

Note that the derivative (and therefore the evolution) of x_i at time t is directly influenced by the value $x_j(t)$ iff $A_{ij} \neq 0$. Moreover, the type (i.e. promoting or inhibiting) and extent of this influence can be associated with the sign and magnitude of the element A_{ij}, respectively. Thus, if we consider the state variables as quantities associated with the nodes of a network, the matrix A can be considered as a compact numerical representation of the network topology. Therefore, the topological reverse engineering problem can be recast as the problem of identifying the dynamical system (1.2). A possible criticism of this approach could be raised with respect to the use of a linear model, which is certainly inadequate to capture the complex nonlinear dynamics of certain molecular reactions. However, this criticism would be reasonable only if the aim was to identify an accurate model of large changes in the states of a biological system over time, and this is not the case here. If the goal is simply to describe the qualitative functional relationships between the states of the system when the system is subjected to perturbations then a first–order linear approximation of the dynamics represents a valid choice of model. Indeed, a large number of approaches to network inference and model parameter estimation have recently appeared in the literature which are based on linear dynamical models, e.g. [5, 11, 8, 9, 12]. In addition to their conceptual simplicity, the popularity of such approaches arises in large part due to the existence of many well established and computationally appealing techniques for the analysis and identification of this class of dynamical system.

1.1.2 Nonlinear models

In the following, we introduce several nonlinear models which have also been used for the purposes of network inference. As discussed above, general nonlinear ODE models quickly become intractable for identification purposes and thus, in order to overcome this limitation, alternative modelling approaches have been devised which exploit the particular dynamical characteristics of biological networks.

In general the dynamics of a biomolecular network of m species (x) and

r reactions (v), can be described by a system of ODEs in the form

$$\frac{dx_i}{dt} = \sum_{j=1}^{r} n_{ij} v_j \,,$$

for $i = 1, \ldots, m$, where n_{ij} is the stoichiometric coefficient of the i-th species in the j-th reaction ($n_{ij} > 0$ for products and $n_{ij} < 0$ for reactants) and v_j is the rate of the j-th reaction. For the sake of simplicity, we assume that the changes of concentrations are only due to reactions (i.e. we neglect the effect of convection or diffusion). We can then define the stoichiometric matrix $N = n_{ij}$, for $i = 1, \ldots, m$, and for $j = 1, \ldots, r$, in which columns correspond to reactions and rows to concentration variations. Therefore, the mathematical description of a biomolecular network can be given in matrix form as

$$\frac{dx}{dt} = Nv \,. \tag{1.3}$$

The reaction rate v_j is typically a polynomial (e.g. mass-action kinetics) or rational (e.g. Michaelis-Menten or Hill functions) function of the concentrations of the chemical species taking part in a reaction. A particular case is when these reaction rates are approximated through power-law terms, yielding the so-called S-system models.

Polynomial and rational models

The behavior of a biomolecular network can be described by a system of differential equations obtained from the reaction mechanism by the law of mass action: the rate of an elementary reaction (a reaction that proceeds through only one transition state, that is one mechanistic step) is proportional to the product of the concentrations of the participating molecules. For example, for the following interaction

$$\mathrm{A} + \mathrm{B} \xrightarrow{k} \mathrm{C} \,,$$

the rate of change of protein C with respect to time can be described as a polynomial function

$$\frac{d[C]}{dt} = k[A][B] \,,$$

where $[A]$, $[B]$ and $[C]$ denote the concentrations of the molecules A, B and C, respectively, and k represents the rate constant that depends on reaction conditions such as temperature, pH, solvents, etc.

To occur at significant rates, almost all biological processes in the cell need enzymes, proteins that catalyze chemical reactions. Consider the simplest enzymatic reaction, in which there is a reversible association between an enzyme E and a substrate S, yielding an intermediate enzyme-substrate complex C which irreversibly breaks down to form a product P:

$$\mathrm{S} + \mathrm{E} \underset{k_{-1}}{\overset{k_1}{\rightleftharpoons}} \mathrm{C} \xrightarrow{k_2} \mathrm{P} + \mathrm{E}\,, \tag{1.4}$$

where k_1, k_{-1} and k_2 are relative reaction constants. By applying the law of mass action we obtain

$$\begin{aligned}
\frac{d[S]}{dt} =& - k_1[E][S] + k_{-1}[C] \\
\frac{d[E]}{dt} =& - k_1[E][S] + (k_{-1} + k_2)[C] \\
\frac{d[C]}{dt} =& k_1[E][S] - (k_{-1} + k_2)[C] \\
\frac{d[P]}{dt} =& k_2[C]\,,
\end{aligned}$$

where the $[E]$, $[S]$, $[C]$ and $[P]$ denote the concentrations of the relative proteins, with the initial conditions $([S], [E], [C], [P]) = ([S_0], [E_0], 0, 0)$ at time $t = 0$. Note that, by summing the second and third equation, the total amount of free and bound enzyme is invariant over time

$$\frac{d[E]}{dt} + \frac{d[C]}{dt} = 0 \Rightarrow [E](t) + [C](t) = E_0\,,$$

and thus we obtain the simplified model described by the following two equations:

$$\begin{aligned}
\frac{d[S]}{dt} =& - k_1 E_0[S] + (k_1[S] + k_{-1})[C] \\
\frac{d[C]}{dt} =& k_1 E_0[S] - (k_1[S] + k_{-1} + k_2)[C]\,,
\end{aligned}$$

with the initial conditions $([S], [C]) = ([S_0], 0)$ at time $t = 0$. As the formation of the complex C is very fast it may be considered to be at the equilibrium state ($\frac{d[C]}{dt} = 0$), and we obtain

$$[C] = \frac{E_0[S]}{[S] + K_m} \Rightarrow \frac{d[P]}{dt} = -\frac{d[S]}{dt} = \frac{v_{max}[S]}{[S] + K_m},$$

where $v_{max} = k_2 E_0$ is the maximum reaction velocity and $K_m = \frac{k_{-1}+k_2}{k_1}$ is known as the Michaelis-Menten constant. Clearly this kinetic term is a rational function. This type of rate law exhibits saturation at high substrate concentrations, a well known behavior of enzymatic reactions. Empirically, for many reactions the rate of product formation follows sigmoidal kinetics with the substrate concentration. In 1910, Hill devised an equation to describe the cooperative binding (i.e. the affinity of a protein for its ligand changes with the amount of ligand already bound) of oxygen to haemoglobin, which is of the form for enzyme-substrate reactions:

$$\frac{d[P]}{dt} = -\frac{d[S]}{dt} = \frac{v_{max}[S]^n}{[S]^n + {K_m}^n},$$

where n is the Hill coefficient. From the Hill equation we see that in the absence of cooperativity $n = 1$. $n > 1$ is called positive cooperativity and $n < 1$ negative cooperativity.

The Michaelis-Menten reaction (1.4), in the form of the stoichiometric model (1.3), is given by

$$x = \begin{pmatrix} [E] \\ [S] \\ [ES] \\ [P] \end{pmatrix}, v = \begin{pmatrix} v_1 \\ v_2 \\ v_3 \end{pmatrix}, N = \begin{pmatrix} -1 & 1 & 1 \\ -1 & 1 & 0 \\ 1 & -1 & -1 \\ 0 & 0 & 1 \end{pmatrix},$$

where v_1, v_2 and v_3 represent the reaction rates of the complex $[ES]$ formation, of the $[ES]$ dissociation and of $[P]$ production, respectively. The relation (1.3) hides the underlying chemical network structure that we are trying to identify. Hence, in the following, we introduce the notation used in chemical reaction network theory: N and v are decomposed into the so called *bookkeeping matrix* Y, which maps the space of complexes into the space of species, the concentration vector of the different complexes $\Psi(x)$ and matrix A_k, which defines the network structure. For the Michaelis-Menten reaction

(1.4), the vector of complexes is given by

$$\Psi = \begin{pmatrix} [E][S] \\ [ES] \\ [E][P] \end{pmatrix} .$$

The matrix Y is determined in the following way: the elements of the i-th row tell us in which complexes species i appears and how often; equivalently, the entries to the j-th column tell us of how much of each species make up complex j. Thus, for (1.4),

$$Y = \begin{pmatrix} 1 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} .$$

Matrix K is the transpose of the weighted adjacency matrix of the digraph representing the chemical reaction network; that is, entry K_{ij} is nonnegative and corresponds to the rate constant associated with the reaction from complex j to i. The so called kinetic matrix A_k is given by $A_k = K - diag(K^T e)$, where $e = (1, \ldots, 1)^T \in \mathbb{R}^n$ and n is the number of complexes. For (1.4),

$$K = \begin{pmatrix} 0 & k_{-1} & 0 \\ k_1 & 0 & 0 \\ 0 & k_2 & 0 \end{pmatrix}, K = \begin{pmatrix} -k_1 & k_{-1} & 0 \\ k_1 & -(k_{-1} + k2) & 0 \\ 0 & k_2 & 0 \end{pmatrix} .$$

The set of nonlinear ODEs given by (1.2) can be rewritten as [13]:

$$\frac{dx}{dt} = Y A_k \Psi(x) , \tag{1.5}$$

where $\ln \Psi(x) = Y^T \ln x$. Then, from experimental data (in addition Y is often known), the identification problem corresponds to the reconstruction of the network structure given by A_k.

An example of a mathematical model of a biological pathway, based on the mass-action law for protein interactions and the saturating rate law for transcriptional reactions using a Hill-type function, is presented in [14], where the authors constructed a model of the aryl hydrocarbon receptor (AhR) signal transduction pathway.

Rational terms do not necessarily have to arise from the law of mass action, but can also be used as a phenomenological description of some biological events exhibiting a sigmoidal response. For instance, in [15] the authors modelled a gene network, consisting of genes, mRNA and proteins, by the following ODE system:

$$\begin{aligned}\frac{d[x_i]}{dt} &= m_i \cdot f_i(y) - {\lambda_i}^{RNA} \cdot x_i \\ \frac{d[y_i]}{dt} &= r_i \cdot x_i - {\lambda_i}^{Prot} \cdot y_i \,,\end{aligned}$$

where m_i is the maximum transcription rate, r_i the translation rate, ${\lambda_i}^{RNA}$ and ${\lambda_i}^{Prot}$ are the mRNA and protein degradation rates, respectively. $f_i(\cdot)$ is the so-called input function of gene i, which determines the relative activation of the gene, modulated by the binding of transcription factors (TFs) to cis-regulatory sites, and is approximated using Hill-type terms.

S-systems

Power-law models have been developed as an alternative approach for modelling reactions following non-ideal kinetics in various biological systems [16]. The basic concept underlying power-law models is the approximation of classical ODE models by means of a uniform mathematical structure. Michaelis-Menten kinetics of the form

$$v = v(X) = \frac{v_{max}[X]}{K_m + [X]}$$

can be approximated by power-law functions as

$$v \approx \alpha[X]^g \,.$$

A particular class of power-law models are the S-systems (synergistic-systems), where the rate of change of a state variable is equal to the difference of two products of variables raised to non-integer powers:

$$\frac{d[X_i]}{dt} = \alpha_i \prod_{j=1}^{n} {X_j}^{g_{i,j}} - \beta_i \prod_{j=1}^{n} {X_j}^{h_{i,j}} \,, \tag{1.6}$$

for $i = 1, \ldots, n$, where the first term represents the net production and the second term the net removal rate for the i-th species, α_i and β_i are multiplicative parameters called *rate constants*, and $g_{i,j}$ and $h_{i,j}$ are exponential parameters called *kinetic orders* for the production and degradation term, respectively. By changing to a logarithmic scale, the relation (1.2) becomes a linear system that is much more tractable to analyze than the original nonlinear system. However the generalized aggregation may introduce a loss of accuracy, the model may conceal important structural features of the network, and it is not able to describe many important biochemical effects such as saturation and sigmoidicity [17].

1.2 Reconstruction methods based on linear models

The general problem of reverse engineering a biological interaction network from experimental data may be tackled via methods based on dynamical linear systems identification theory. The basic step of the inference process consists of estimating, from experimental measurements (either steady–state or time–series data), the weighted connectivity matrix A and the exogenous perturbation vector B of the *in silico* network model (1.2). With this objective, the problem may be tackled by means of regression algorithms based on the classical Least Squares Estimator (LSE), extensions of the LSE algorithm, such as the Constrained Total Least Squares (CTLS) technique, or by using efficient convex optimization procedures cast in the form of linear matrix inequalities (LMIs).

1.2.1 Least Squares

In this subsection, approaches based on the classical Least Squares method are presented. The problem tackled consists of identifying the original network topology starting from the experimental measurements of the temporal evolution of each node.

Identification of the connectivity matrix by Least Squares methods

Assuming that $h+1$ experimental observations, $x(k) \in \mathbb{R}^n$, $k = 0, \ldots, h$, are available, we can recast the problem in the discrete–time domain as

$$X := \begin{pmatrix} x(h) & \ldots & x(1) \end{pmatrix} = \Theta\Omega\,, \tag{1.7}$$

where

$$\Theta = \begin{bmatrix} \hat{A} & \hat{B} \end{bmatrix}, \quad \Omega := \begin{pmatrix} x(h-1) & \ldots & x(0) \\ u(h-1) & \ldots & u(0) \end{pmatrix}.$$

Since we are dealing with a linear model, it is possible to separately estimate each row, $\theta_{i,\star}$, of the connectivity matrix Θ to be identified. Let $Z = \Omega^T \in \mathcal{R}^{h\times(n+1)}$, $X_i = (x_i(h), \ldots, x_i(1))^T \in \mathcal{R}^h$ and $\beta = \hat{\theta}^T_{i,\star} \in \mathcal{R}^{n+1}$. The problem to be solved in a standard Least Squares (LS) setting can then be formulated as follows

$$X_i = Z \cdot \beta\,. \tag{1.8}$$

Now, if we assume that the measurements are noisy, relation (1.8) can be written in the following form

$$X_i + \Delta X_i = (Z + \Delta Z) \cdot \beta\,, \tag{1.9}$$

where

$$\Delta Z = \begin{pmatrix} 1_{v_{h-1}} & \cdots & n_{v_{h-1}} & 0 \\ \vdots & \ddots & \vdots & \vdots \\ 1_{v_0} & \cdots & n_{v_0} & 0 \end{pmatrix} \in \mathcal{R}^{h\times(n+1)}\,,$$

$$\Delta X_i = (i_{v_h} \cdots i_{v_1})^T \in \mathcal{R}^h\,,$$

and i_{v_j} is the i-th noise component of v_j, for $i = 1, \ldots, n$ and for $j = 0, \ldots, h$. ΔZ and ΔX_i are unknown terms caused by the noise in the data. Although the exact values of the correction terms, ΔZ and ΔX_i, are not known, the structure, i.e. how the noise appears in each element, is known. If the unknown terms are ignored, then the problem is solved by the standard least squares (LS) method as follows. The optimal estimator for the regression coefficients vector is given by

$$\left(Z^T Z\right) \beta_{LS} = Z^T X_i\,,$$

which admits the unique solution

$$\beta_{LS} = \left(Z^T Z\right)^{-1} Z^T X_i$$

if the number of samples is greater or equal than the number of regression coefficients, that is $h \geq n+1$, and the matrix Z has full rank, $n+1$. When Z does not have full rank, there are infinitely many LSEs of β, in the form

$$\beta_{LS} = \left(Z^T Z\right)^{\dagger} Z^T X_i\,,$$

where $\left(Z^T Z\right)^{\dagger}$ is a generalized inverse of $\left(Z^T Z\right)$ [18].

1.2.2 Methods based on Least Squares

In this section, we introduce some methods for reverse-engineering biomolecular networks based on linear models and least squares regression. In this field, important contributions have been produced by the groups of Gardner and Collins, that devised NIR (Network Identification by multiple Regression, [5]) and MNI (Microarray Network Identification, [19]) algorithms, and by the group of di Bernardo, that devised TSNI (Time-Series Network Identification, [20, 21]). A common motif of these approaches is the use of linear ODE models and multivariate regression to identify the targets of exogenous perturbations in a biomolecular network.

The NIR algorithm has been developed for application with perturbation experiments on gene regulatory networks. The direct targets of the perturbation are assumed to be known and the method uses only the steady-state gene expression. Under the steady-state assumption ($\dot{x}(t) = 0$ in (1.2)) the problem to be solved is

$$\sum_{j=1}^{n} a_{ij} x_j = -b_i u\,, \tag{1.10}$$

The least squares formula is used to compute the network structure, that is the rows $a_{i,\star}$ of the connectivity matrix, from the gene expression profiles ($x_j\,, j = 1, \ldots, n$) following each perturbation experiment; the genes that are directly affected by the perturbation are expressed through a nonzero element in the B vector. NIR is based on a network sparsity assumption: only k (maximum number of incoming edges per gene) out of the n elements on each row are different from zero. For each possible combination of k out of

n weights, the k coefficients for each gene are computed such as to minimize the interpolation error. The maximum number of incoming edges, k, can be varied by the user. An advantage of NIR is that k can be tuned so as to avoid underdetermined problems. Indeed, if one has N_e different (independent) perturbation experiments, the exact solution to the regression problem can be found for $k \leq N_e$, at least in the ideal case of zero noise.

The MNI algorithm, similarly to NIR, uses steady-state data and is based on relation (1.10), but it does not require *a priori* knowledge of the specific target gene for each perturbation. The algorithm employs an iterative procedure: first, it predicts the targets of the treatment using a full network model; subsequently, it translates the predicted targets into constraints on the model structure and repeats the model identification to improve the reconstruction. The procedure is iterated until certain convergence criteria are met.

The TSNI algorithm uses time-series data, instead of steady-state values, of gene expression following a perturbation. It identifies the gene network (A), as well as the direct targets of the perturbations (B), by applying the LS to solve the linear equation (1.2). Note that, to solve (1.2), it is necessary to measure the derivative values, which are never available. Numerical estimation of the derivative is not a suitable option, since it is well known to yield considerable amplification of the measurement noise. The solution implemented by TSNI consists in converting (1.2) to the corresponding discrete-time system

$$x(k+1) = A_d x(k) + B_d u(k)\,,$$

As discussed above (see Section 1.2.1) this problem admits a unique globally optimal solution if $h \geq n + p$, where h is the number of data points, n is the number of state variables and p the number of perturbations. To increase the number of data points, after using a cubic smoothing spline filter, a piecewise cubic spline interpolation is performed. Then a Principal Component Analysis (PCA) is applied to the data-set in order to reduce its dimensionality and the problem is solved in the reduced dimension space. In order to compute the continuous-time system's matrices, A and B, from the corresponding discretized A_d and B_d, respectively, the following bilinear transformation is applied [22]:

$$A = \frac{2A_d - I}{T_s A_d + I}$$

$$B = (A_d + I)AB_d \, ,$$

where $I \in \mathbb{R}^{n \times n}$ is the square identity matrix and T_s is the sampling interval.

Bonneau et al. devised another algorithm, named Inferelator [23], which uses regression and variable selection to infer regulatory influences for genes and/or gene clusters from mRNA and/or protein expression levels.

Finally, the group of K.-H. Cho have developed a number of algorithms based on dynamical linear systems theory and convex optimization to infer biological regulatory networks from time-series measurements. In [9] the identification procedure leads to a convex optimization problem with regularization [24] in order to achieve a sparse network and to take into account any *a priori* information on the network structure. In [8] an optimization method was presented which allows the inference of gene regulatory networks from time-series gene expression data taking into account time-delays and noise in the measurement data.

1.2.3 Dealing with Noise: Constrained Total Least Squares

To write (1.9) in a more compact form, we make the following definitions:

$$C := \begin{pmatrix} Z & X_i \end{pmatrix} \, ,$$

$$\Delta C := \begin{pmatrix} \Delta Z & \Delta X_i \end{pmatrix} \, .$$

Then the relation (1.8) is written as

$$\Big(C + \Delta C \Big) \begin{pmatrix} \beta \\ -1 \end{pmatrix} = 0 \, . \tag{1.11}$$

An extension of the LS algorithm, namely the Total Least Squares (TLS) technique, was developed to solve exactly this problem by finding the correction term ΔC. The TLS problem is then posed as follows [25]:

$$\begin{aligned} & \min_{v,\beta} ||\Delta C||_F^2 \\ \text{s.t.} \quad & \Big(C + \Delta C\Big) \begin{pmatrix} \beta \\ -1 \end{pmatrix} = 0 \, , \end{aligned} \tag{1.12}$$

where $|| \cdot ||_F$ is the Frobenius norm defined by $||A||_F = \sqrt{tr(AA^T)}$ for a matrix A in which $tr(AA^T)$ is the trace of the matrix, i.e. the sum of the diagonal terms. When the smallest singular value of $(Z \quad X_i)$ is not repeated, the solution of the TLS problem is given by:

$$\beta_{TLS} = \left(Z^T Z - \lambda^2 I\right)^{-1} Z^T X_i \, , \tag{1.13}$$

where λ is the smallest singular value of $(Z \quad X_i)$. The TLS solution has a correction term, λ^2, at the inverse of the matrix, compared to the LSE. This reduces the bias in the solution, which is caused by the noise in C. The TLS solution can be also computed using the singular value decomposition as follows ([26], p. 503):

$$C = U \Sigma V^T \, ,$$

where $U \Sigma V$ is the singular value decomposition of the matrix C, $U \in \mathcal{R}^{h \times h}$ and $V \in \mathcal{R}^{(n+2)\times(n+2)}$ are unitary matrices and $\Sigma \in \mathcal{R}^{h \times (n+2)}$ is a diagonal matrix with $k = \min(h, n+2)$ non-negative singular values, σ_i, arranged in descending order along its main diagonal, the other entries are zero. The singular values are the positive square roots of the eigenvalues of $C^T C$. Let $V = [\nu_1 \cdots \quad \nu_n \quad \nu_{n+1} \quad \nu_{n+2}]$, where $\nu_i \in \mathcal{R}^{n+2}$ is the i-th column of the matrix V. Then, the solution is given by

$$\begin{pmatrix} \beta_{TLS} \\ 1 \end{pmatrix} = -\frac{\nu_{n+2}}{\nu} \, , \tag{1.14}$$

where ν is the last element of ν_{n+2}. Numerically, this is a more robust method than computing the inverse of a matrix.

The TLS solution, (1.13) or (1.14), is not optimal when the two noise terms in Z and X_i are correlated, since one of the main assumptions in this method is that the two noise terms are independent of each other. If there is some correlation between them, this knowledge can be used to improve the solution by using the constrained total least squares (CTLS) technique [27]. In the case of the problem in the form (1.9), the two noise terms are obviously correlated because Z is a function of the noise term from the sampling time k equal to 1 to h and X_i is a function of the noise term from k equal to 0 to $h-1$. To use the structural information in ΔC, first the minimal set of noise is defined as follows:

$$v = (v_h^T \cdots v_0^T)^T \in \mathcal{R}^{n(h+1)\times 1}\,.$$

If v is not white random noise, a whitening process using Cholesky factorization is performed [27]. Here, v is assumed to be white noise and this whitening process is not necessary. Consider each column of ΔC, i.e.

$$\Delta C = \begin{pmatrix} \Delta C_1 & \cdots & \Delta C_n & \Delta C_{n+1} & \Delta C_{n+2} \end{pmatrix}\,,$$

where ΔC_i is the i-th column vector of ΔC. More specifically

$$\Delta C_i = (i_{v_{h-1}} \cdots i_{v_0})^T \in \mathcal{R}^h, \quad i = 1, \ldots, n\,,$$

$$\Delta C_{n+1} = 0_{h\times 1}, \quad \Delta C_{n+2} = \Delta X_i = (i_{v_h} \cdots i_{v_1})^T \in \mathcal{R}^h\,. \tag{1.15}$$

Each ΔCi can be written as

$$\Delta Ci = G_i v$$

for $i = 1, \ldots, n, n+1, n+2$. To obtain the explicit form for each G_i, we first define the following column vector of all zero elements, but one, the i-th element, equal to 1:

$$e_i = (0 \cdots 0 \quad 1 \quad 0 \ldots 0)^T \in \mathcal{R}^n \quad i = 1, \ldots, n\,.$$

For i equal to 1,

$$\begin{aligned} \Delta C_1 &= \left(1_{v_{h-1}} \cdots 1_{v_0}\right)^T = \left(v_{h-1}^T e_1 \cdots v_0^T e_1\right) = \\ &= \left[v^T \begin{pmatrix} 0_{n\times h} \\ I_h \otimes e_1 \end{pmatrix} \right]^T = \begin{bmatrix} 0_{h\times n} & (I_h \otimes e_1)^T \end{bmatrix} v\,. \end{aligned}$$

Likewise for the i-th column of ΔC,

$$\Delta C_i = \begin{bmatrix} 0_{h\times n} & (I_h \otimes e_i)^T \end{bmatrix} v$$

and hence

$$G_i = \begin{bmatrix} 0_{h\times n} & (I_h \otimes e_i)^T \end{bmatrix}$$

for $i = 1, \ldots, n$. Also, from (1.15)

$$G_{n+1} = 0_{h\times n(h+1)} ,$$

$$G_{n+2} = \begin{bmatrix} (I_h \otimes e_i)^T & 0_{h\times n} \end{bmatrix} .$$

Since ΔC can be written as

$$\Delta C = \begin{pmatrix} G_1 v & \ldots & G_n v & G_{n+1} v & G_{n+2} v \end{pmatrix} ,$$

then the TLS problem can be recast as follows [27]:

$$\begin{aligned} &\min_{v,\beta} \|v\|^2 \\ &\text{s.t.} \left[C + \begin{pmatrix} G_1 v & \ldots & G_n v & G_{n+1} v & G_{n+2} v \end{pmatrix} \right] \begin{bmatrix} \beta \\ -1 \end{bmatrix} = 0 . \end{aligned} \qquad (1.16)$$

This is called the constrained total least squares (CTLS) problem. With the following definition,

$$H_\beta := \sum_{j=1}^{n} a_{i,j} G_j + b_1 G_{n+1} - G_{n+2} = \sum_{j=1}^{n+1} \beta_j G_j - G_{n+2} , \qquad (1.17)$$

where β_j for $j = 1, \ldots, n$ is the j-th element of the i-th row of A and β_{n+1} is the i-th element of the vector B, (1.16) can be written in the following form:

$$C \begin{bmatrix} \beta \\ -1 \end{bmatrix} H_\beta v = 0 .$$

Solving for v, we get

$$v = -H_\beta^\dagger C \begin{bmatrix} \beta \\ -1 \end{bmatrix} , \qquad (1.18)$$

where $H_\beta^\dagger$ is the pseudoinverse of H_β. Hence, the original constrained minimization problem, (1.16), is transformed into an unconstrained minimization problem as follows:

$$\min_{v,\beta} \|v\|^2 = \min_{\beta} \begin{bmatrix} \beta^T & -1 \end{bmatrix} C^T {H_\beta^\dagger}^T H_\beta^\dagger C \begin{bmatrix} \beta \\ -1 \end{bmatrix} . \qquad (1.19)$$

Now, we introduce two assumptions, which make the formulation simpler.

1. The number of measurements are always strictly greater than the number of unknowns, i.e. we only consider the overdetermined case, explicitly $h+1 > n+2$, that is $h > n+1$.
2. H_β is full rank.

Then the pseudoinverse $H_\beta^\dagger$ is given by

$$H_\beta^\dagger = H_\beta^T \left(H_\beta H_\beta^T\right)^{-1}$$

and the unconstrained minimisation problem can be further simplified as follows:

$$\min_{\beta} \begin{bmatrix} \beta^T & -1 \end{bmatrix} C^T \left(H_\beta H_\beta^T\right)^{-1} C \begin{bmatrix} \beta \\ -1 \end{bmatrix} . \qquad (1.20)$$

The starting guess for β used in the above optimisation problem is simply the value returned by the solution of the standard least squares problem.

The problem to be solved is to find the values of $n(n+1)$ parameters of a linear model that yield the best fit to the observations in the least–squares sense. Hence, as assumed above, if the number of observations are always strictly greater than the number of explanatory variables, that is $h > n+1$, then the problem admits a unique globally optimal solution. In the other case, $h \leq n+1$, the interpolation problem is undetermined, and thus there exist infinitely many values of the optimization variables that equivalently fit the experimental measurements. In this case, several expedients can be adopted: first, it is possible to exploit clustering techniques to reduce the number of nodes and smoothing techniques to increase the number of samples, in order to satisfy the constraint $h > n+1$. Furthermore, adopting a bottom–up reconstruction approach (i.e. starting with a blank network and increasingly adding new edges) may help in overcoming the dimensionality problem: in this case, indeed, the number of edges incident to each node (and therefore the number of explanatory variables) is iteratively increased and can be limited to satisfy the above constraint. Finally, the introduction of sign constraints on the optimization variables, derived from qualitative prior knowledge of the network topology (as described below), results in a significant reduction of the solution space.

Note that the methods illustrated above can be extended if p experiments are performed around the same equilibrium point. Then the terms in the relation (1.9) are constructed as follows:

$$Z = \begin{pmatrix} \zeta^1 \cdots \zeta^p \\ I_p \otimes \mathbf{1}_{1\times h} \end{pmatrix}^T , \quad X_i = \begin{pmatrix} {\chi_i}^1 \cdots {\chi_i}^p \end{pmatrix}^T ,$$

where

$$\zeta^k = \begin{pmatrix} {x_1}^k(h-1) & \cdots & {x_1}^k(0) \\ \vdots & \ddots & \vdots \\ {x_n}^k(h-1) & \cdots & {x_n}^k(0) \end{pmatrix} \in \mathcal{R}^{n\times h} ,$$

is the k-th set of experimental data, ${\chi_i}^k = (x_i^k(h), \ldots, x_i^k(1)) \in \mathcal{R}^h$, for $k = 1, \ldots ,p$, $I_p \in \mathcal{R}^{p\times p}$ is the identity matrix, $\mathbf{1} \in \mathcal{R}^{1\times h}$ is a vector of ones and $\otimes$ is the Kronecker product. The unknown β is given by

$$\beta = \begin{pmatrix} \hat{a}_{i,\star} & \hat{b_i}^1 \cdots \hat{b_i}^p \end{pmatrix}^T \in \mathcal{R}^{n+p}$$

The correction terms, ΔZ and ΔX_i, are given by

$$\Delta Z = \begin{pmatrix} \Delta\zeta^1 \cdots \Delta\zeta^p \\ \mathbf{0}_{p\times ph} \end{pmatrix} , \quad \Delta X_i = \begin{pmatrix} \Delta{\chi_i}^1 \cdots \Delta{\chi_i}^p \end{pmatrix} ,$$

where

$$\Delta\zeta^k = \begin{pmatrix} 1^k_{v_{h-1}} & \cdots & 1^k_{v_0} \\ \vdots & \ddots & \vdots \\ n^k_{v_{h-1}} & \cdots & n^k_{v_0} \end{pmatrix} \in \mathcal{R}^{n\times h} ,$$

$$\Delta{\chi_i}^k = (i^k_{v_h} \cdots i^k_{v_1})^T \in \mathcal{R}^h ,$$

for $k = 1, \ldots ,p$ and $\mathbf{0} \in \mathcal{R}^{p\times ph}$ is a matrix of zeros .

PACTLS algorithm

In this subsection we describe the PACTLS algorithm, a method devised for the reverse engineering of partially-known networks from noisy data. PACTLS uses the CTLS technique to optimally reduce the effects of measurement noise in the data on the reliability of the inference results, while exploiting qualitative prior knowledge about the network interactions with an edge selection heuristic based on mechanisms underpinning scale–free network generation, i.e. network growth and preferential attachment (PA).

The algorithm allows prior knowledge about the network topology to be taken into account within the CTLS optimization procedure. Since each element of A can be interpreted as the weight of the edge between two nodes of the network, this goal can be achieved by constraining some of the optimization variables to be zero and others to be strictly positive (or negative), and using a constrained optimization problem solver, e.g. the nonlinear optimisation function *fmincon* from the MATLAB Optimization Toolbox, to solve (1.20). Similarly, we can impose a sign constraint on the i-th element of the input vector, b_i, if we *a priori* know the qualitative (i.e. promoting or repressing) effect of the perturbation on the i-th node. Alternatively, an edge can be easily pruned from the network by setting to zero the corresponding entry in the minimization problem.

Note that, since the system evolution is sampled, $\hat{A}$ and $\hat{B}$ are not actually the estimates of A and B in (1.2), but rather of the corresponding matrices of the discrete–time system obtained through the Zero-Order-Hold (ZOH) discretization method ([28], p. 676) with sampling time T_s from system (1.2), that is

$$x(k+1) = A_d x(k) + B_d u(k)\,, \tag{1.21}$$

where $x(k+1)$ is a shorthand notation for $x(kT_s+T_s)$, $x(k)$ for $x(kT_s)$, $u(k)$ for $u(kT_s)$, and

$$A_d = e^{AT_s}\,, \qquad B_d = \left(\int_0^{T_s} e^{A\tau} d\tau\right) B\,.$$

In general, the sparsity patterns of A_d and B_d differ from those of A and B. However, if the sampling time is suitably small, $(A)_{ij} = 0$ implies that $(A_d)_{ij}$ exhibits a very low value, compared to the other elements on the same row and column, and the same applies for B_d and B (see Section 1.2.5 for a detailed discussion). Therefore, in order to reconstruct the original sparsity pattern of the continuous–time system's matrices, one can set to zero the elements of the estimated matrices whose values are below a certain threshold; this is the basic principle underpinning the edges selection strategy, as described next.

So far we have described a method to add/remove edges and to introduce constraints on the sign of the associated weights in the optimization problem. The problem remains of how to devise an effective strategy to select the nonzero entries of the connectivity matrix.

The initialization network for the devised algorithm has only self–loops on every node, which means that the evolution of the i-th state variable is always influenced by its current value. This yields a diagonal initialization matrix, $\hat{A}^{(0)}$. Subsequently, new edges are added step-by-step to the network according to the following iterative procedure:

P1) A first matrix, $\bar{A}$, is computed by solving (1.20) for each row, without setting any optimization variable to zero. The available prior information is taken into account at this point by adding the proper sign constraints on the corresponding entries of A before solving the optimization problem, as explained in the previous subsection. Since it typically exhibits all nonzero entries, matrix $\bar{A}$ is not representative of the network topology, but is rather used to weight the relative influence of each entry on the system's dynamics. This information will be used to select the edges to be added to the network at each step. Each element of $\bar{A}$ is normalized with respect to the values of the other elements in the same row and column, which yields the matrix $\tilde{A}$, whose elements are defined as

$$\tilde{A}_{ij} = \frac{\bar{A}_{ij}}{\left(\|\bar{A}_{\star,j}\| \cdot \|\bar{A}_{i,\star}\|\right)^{1/2}} \, .$$

P2) At the k-th iteration, the edges ranking matrix $\tilde{G}^{(k)}$ is computed,

$$\tilde{G}^{(k)}_{ij} = \frac{|\tilde{A}_{ij}| p_j^{(k)}}{\sum\limits_{l=1}^{n} p_l^{(k)} |\tilde{A}_{il}|} \, , \tag{1.22}$$

where

$$p_j^{(k)} = \frac{K_j^{(k)}}{\sum\limits_{l=1}^{n} K_l^{(k)}} \tag{1.23}$$

is the probability of inserting a new edge starting from node j and $K_l^{(k)}$ is the number of outgoing connections from the l-th node at the k-th iteration. The $\mu(k)$ edges with the largest scores in $\tilde{G}^{(k)}$ are selected and added to the network; $\mu(\cdot)$ is chosen as a decreasing function of k, that is $\mu(k) = \lceil n/k \rceil$. Thus, the network grows rapidly at the beginning and is subsequently refined by adding smaller numbers of nodes at each iteration. The form of the function $p(\cdot)$ stems from the so–called *preferential attachment* (PA) mechanism, which states that in a growing network new edges preferentially start from *popular* nodes (those with the highest connectivity degree, i.e. the hubs). By exploiting the mechanisms of network growth and PA, we are able to guide the network reconstruction algorithm to increase the probability of producing a network with a small number of hubs and many poorly connected nodes. Note also that, for each edge, the probability of incidence is blended with the edge's weight estimated at point P1); therefore, the edges with larger estimated weights have a higher chance to be selected. This ensures that the interactions exerting greater influence on the network dynamics have a higher probability of being selected.

P3) The structure of nonzero elements of $\hat{A}^{(k)}$ is defined by adding the entries selected at point P2) to those selected up to iteration $k - 1$ (including those derived by *a priori* information), and the set of inequality constraints is updated accordingly; then Problem 1.20 for each row, with the additional constraints, is solved to compute $\hat{A}^{(k)}$.

P4) The residuals generated by the identified model are compared with the values obtained at the previous iterations; if the norm of the vector of residuals has decreased, in the last two iterations, at least by a factor ϵ_r with respect to the value at the first iteration, then the procedure iterates from point P2), otherwise it stops and returns the topology described by the sparsity pattern of $\hat{A}^{(k-2)}$. The factor ϵ_r is inversely correlated with the number of edges inferred by the algorithm; on the other hand, using a smaller value of ϵ_r raises the probability of obtaining false positives. By conducting numerical tests for different values of ϵ_r, we have found that setting $\epsilon_r = 0.1$ yields a good balance between the various performance indices.

Concerning the input vector, we assume that the perturbation targets and the qualitative effects of the perturbation are known, thus the pattern (but not the values of the nonzero elements) of $\hat{B}$ is preassigned at the initial step and the corresponding constraints are imposed in all the subsequent iterations.

1.2.4 Convex optimization methods

In this section, two methods for identifying a linear network model by means of a convex optimization procedure cast in the form of LMIs are illustrated [29, 10]. The distinctive feature of these methods is that they easily enable exploitation of any qualitative prior knowledge which may be available from knowledge of the underlying biology, thus significantly increasing the inference performance. The first algorithm is based on an iterative procedure which starts from a fully connected network - the edges are subsequently pruned according to a maximum parsimony criterion. The pruning algorithm terminates when the estimation error exceeds an assigned threshold. The second algorithm, named CORE-Net (Convex Optimisation algorithm for Reverse Engineering biological interaction Networks), is also based on the mechanisms of growth and PA, as in the algorithm described in Section 1.2.3.
A similar approach based on convex optimization was devised by Julius et al. in [30], where a method for identifying genetic regulatory networks using expression profiles from genetic perturbation experiments is presented. A stability constraint is added to the convex optimization problem and the l_1-norm is used in the cost function in order to obtain a sparse connectivity matrix.

Identification of the connectivity matrix via LMI–based optimization

Assuming that $h+1$ experimental observations, $x(k) \in \mathbb{R}^n$, $k = 0, \ldots, h$, are available, we can recast the problem in the discrete–time domain as shown in (1.7). Our aim is to reconstruct the matrix A and vector B from the experimental values $x(k)$, $k = 0, \ldots, h$. The identification problem can be transformed into that of minimizing the norm of $\Xi - \Theta\Omega$, and thus we can state the following problem.

Problem 1 *Given the sampled data set $x(k)$, $k = 0, \ldots, h$, and the associated matrices Ξ, Ω, find*

$$\begin{aligned} &\min_{\Theta} \varepsilon \\ &s.t. \qquad (\Xi - \Theta\Omega)^T (\Xi - \Theta\Omega) < \varepsilon I \,. \end{aligned} \tag{1.24}$$

Note that condition (1.24) is quadratic in the unknown matrix variable Θ. In order to obtain a linear optimization problem we convert it to the equivalent condition

$$\begin{pmatrix} -\varepsilon I & (\Xi - \Theta\Omega)^T \\ (\Xi - \Theta\Omega) & -I \end{pmatrix} < 0 \,, \tag{1.25}$$

The equivalence between (1.24) and (1.25) is readily derived by applying the following lemma.

Lemma 1 *Let $M \in \mathbb{R}^{n\times n}$ be a square symmetric matrix partitioned as*

$$M = \begin{pmatrix} M_{11} & M_{12} \\ M_{12}^T & M_{22} \end{pmatrix} , \tag{1.26}$$

assume that M_{22} is nonsingular and define the Schur complement of M_{22}, $\Delta := M_{11} - M_{12}M_{22}^{-1}M_{12}^T$. The following statements are equivalent

i) M is positive (negative) definite;

ii) M_{22} and Δ are both positive (negative) definite.

Proof. Recall that M is positive (negative) definite iff

$$\forall x \in \mathbb{R}^n \,, \, x^T M x > 0 \quad (< 0) \,,$$

moreover it can be decomposed as ([31], p.14)

$$\begin{aligned} M &= \begin{pmatrix} M_{11} & M_{12} \\ M_{12}^T & M_{22} \end{pmatrix} \\ &= \begin{pmatrix} I & M_{12}M_{22}^{-1} \\ 0 & I \end{pmatrix} \begin{pmatrix} \Delta & 0 \\ 0 & M_{22} \end{pmatrix} \begin{pmatrix} I & M_{12}M_{22}^{-1} \\ 0 & I \end{pmatrix}^T . \end{aligned}$$

The latter is a congruence transformation ([32], p.568), which does not modify the sign definiteness of the transformed matrix; indeed, $\forall x \in \mathbb{R}^n$ and $\forall C, P \in \mathbb{R}^{n \times n}$

$$P \text{ positive (negative) definite} \Rightarrow x^T C^T P C x = z^T P z > 0 \quad (< 0) .$$

Therefore M is positive (negative) definite iff M_{22} and Δ are both positive (negative) definite. □

Problem 1 with the inequality constraint in the form (1.25) is a generalized eigenvalue problem ([33], p. 10), and can be easily solved using efficient numerical algorithms, such as those implemented in the Matlab LMI Toolbox [34].

A noteworthy advantage of the proposed convex optimization formulation is that the approach can be straightforwardly extended to the case when multiple experimental data sets are available for the same biological network. In this case, there are several matrix pairs (Ξ_k, Ω_k), one for each experiment: the problem can be formulated again as in (1.24), but using a number of constraints equal to the number of experiments, that is

$$\begin{aligned} &\min_{\Theta} \varepsilon \\ &\text{s.t.} \qquad (\Xi_k - \Theta\Omega_k)^T (\Xi_k - \Theta\Omega_k) < \varepsilon_k I, \quad k = 1, \ldots, N_e, \end{aligned}$$

where N_e is the number of available experiments.

As discussed in the previous section, the devised technique is based on the assumption that the sparsity pattern of the dynamical matrix and input matrix of system (1.2) can be recovered through the estimation of the corresponding matrices of the associated sampled-data discrete-time system (1.21) (see Section 1.2.5 for a detailed discussion).

Except for the LMI formulation, the problem is identical to the one tackled by classical linear regression, i.e. finding the values of $n(n+1)$ parameters of a linear model that yield the best fitting of the observations in the least–squares sense. Hence, if the number of observations, $n(h+1)$, is greater or equal than the number of explanatory variables, that is $h \geq n$, the problem admits a unique globally optimal solution. In the other case, $h < n$, the interpolation problem is undetermined, thus there exist infinitely many values of the optimization variables that equivalently fit the experimental measurements. In the latter case, as discussed previously in Section 1.2.1, several expedients (clustering techniques, bottom-up reconstruction approach, prior knowledge etc.) can be adopted to solve the undetermination. The key advantage of the LMI formalism is that it makes it possible to take into account prior knowledge about the network topology by forcing some of the optimization variables to be zero and other ones to be strictly positive (or negative), by introducing the additional inequality $A_{ij} > 0$ (< 0) to the set of LMIs. Similarly, we can impose a sign constraint on the i-th element of the input vector, b_i, if we *a priori* know the qualitative (i.e. promoting or repressing) effect of the perturbation on the i-th node. Also, an edge can be easily pruned from the network by setting to zero the corresponding entry in the matrix optimization variable in the LMIs.

Top–down approach

The first algorithm based on an LMI approach consists of an iterative procedure that starts from a fully connected network. Subsequently, the edges are pruned according to a maximum parsimony criterion. The pruning algorithm terminates when the estimation error of the identified model with respect to the experimental data set exceeds an assigned threshold.

The following basic idea underpins the pruning algorithm: Given the identified (normalized) connectivity matrix, the edges connecting non-adjacent (in the original network) nodes have lower weights than the others. This is a straightforward mathematical translation of the reasonable assumption that indirect interactions are weaker than direct ones.

In addition to the loose connectivity assumption, our algorithm is also capable of directly exploiting information about some specific interactions that are *a priori* known, taking into account both the direction of the influence, and its type (promoting or repressing).

The reconstruction algorithm is structured as follows.

P1) A first system is identified by solving Problem 1, and adding all the known sign constraints.

P2) Let $\hat{A}_{(k)} = \left\{\hat{A}_{ij}\right\}_{i,j=1,\ldots,n}$ be the matrix computed at the k-th step; in order to compare the values of the identified coefficients the matrix has to be normalized, so let us define the normalized matrix $\tilde{A}_k = \left\{\tilde{A}_{ij}\right\}_{i,j=1,\ldots,n}$, where

$$\tilde{A}_{ij} = \frac{\hat{A}_{ij}}{\left(\|\hat{A}_{\star,j}\|\|\hat{A}_{i,\star}\|\right)^{1/2}}$$

is obtained by dividing the original value by the norms of its row, $\hat{A}_{\star,j}$, and column, $\hat{A}_{i,\star}$.

P3) The normalized matrix is analyzed to choose the coefficients to be nullified at the next identification step; various rules can be adopted at this step in order to define the threshold below which a given coefficient is nullified. Good performance has been obtained by setting, for each coefficient, two thresholds proportional to the mean values of its row and column. Then the element is nullified only if its absolute value is lower than the minimum of the two thresholds. This rule reflects the idea that an arc is a good candidate for elimination if its weight is low compared to the other arcs arriving to and starting from the same node.

P4) After choosing the coefficients to be nullified, a new LMI problem is cast, eliminating the corresponding optimization variables, and a new solution is computed.

P5) The evolution of the identified system is compared with the experimental data: If the estimation error exceeds a prefixed threshold then the algorithm stops, otherwise another iteration starts from point $P2$).

The algorithm requires tuning two optimisation parameters: 1) The threshold value used in the pruning phase, which affects the number of coefficients eliminated at each step; 2) The upper bound defining the admissible estimation error, which determines the algorithm termination. The first parameter influences the connectivity of the final reconstructed network: the greater its value, the lower the number of connections, thus it is termed the *specificity parameter*. The algorithm terminates when either it does not find new arcs to remove or the estimation error becomes too large.

Concerning the input vector, in this phase, we do not take into account the effects of external perturbations, then $\hat{B} = 0$ and the linear systems, that represent the *in–silico* networks, are not subject to exogenous inputs.

Bottom–up approach – CORE-Net algorithm

The CORE-Net algorithm, based on the convex optimization problem cast in the LMI form as shown above, is specifically suited to infer gene regulatory networks (GRNs) exhibiting a scale-free topology. The procedure adopts a bottom-up reconstruction approach, that allows the iterative increment of the number of edges, exploiting the growth and PA mechanisms, as for the PACTLS algorithm, illustrated in Section 1.2.3.

A key point underlying CORE-Net (as for PACTLS) is the experimental observation that Metabolic [35], protein–protein interaction [36] and transcriptional networks [37], as well as many other genomic properties [38], typically have a small number of hubs and many poorly connected nodes. A plausible hypothesis for the emergence of such a feature, as shown in [39], is the PA mechanism during network growth and evolution, where, when a new node is added to the network, it is more likely to be connected with one of the few hubs than with one of the many other loosely connected nodes. In large networks, this evolution rule may generate particular degree distributions, e.g. the well known power-law distribution of scale-free networks. The technique devised implements the PA mechanism within the reconstruction process, therefore mimicking the evolution of a biological networks to improve the inference performance. Note also that in the case of partially-known interaction networks, it is very likely that the available prior knowledge about the connections of the most widely studied genes/proteins in literature will often coincide with the hubs of the network, and in such cases we can expect that the PA mechanism will provide significant advantages in the network reconstruction process.

It is worth noticing that, differently from the approach based on top–down strategy, where the algorithm starts from a fully connected network and then the edges are subsequently pruned according to a maximum parsimony criterion, CORE-Net iteratively increments the number of edges, thus helping to overcome the undetermination problem, as discussed previously, and to decrease the computational burden.

1.2.5 Sparsity pattern of the discrete–time model

The techniques described in the previous sections are based on the assumption that the sparsity pattern of the dynamical matrix and input matrix of system (1.2) can be recovered through the estimation of the corresponding matrices of the associated sampled-data discrete-time system (1.21). Here we want to validate this hypothesis, by analyzing the relationship between the dynamical matrices of the continuous-time and discrete-time systems.

For the sake of simplicity, in what follows we will assume that A has n distinct real negative eigenvalues, λ_i, $|\lambda_i| < |\lambda_{i+1}|$, $i = 1, \dots, n$ and it is therefore possible to find a nonsingular matrix P such that[1] $A = PDP^{-1}$, with $D = \operatorname{diag}(\lambda_1, \dots, \lambda_n)$. Then, the matrix A_d can be rewritten as ([32], p.525)

$$\begin{aligned} A_d &= I + AT_s + \frac{(AT_s)^2}{2!} + \frac{(AT_s)^3}{3!} + \dots \\ &= P\operatorname{diag}\left(e^{\lambda_1 T_s}, \dots, e^{\lambda_n T_s}\right) P^{-1} . \end{aligned} \tag{1.27}$$

If the sampling time is properly chosen, such as to capture all the dynamics of the system, then $T_s \ll \tau_i := 1/|\lambda_i|$, $i = 1, \dots, n$, which implies $\lambda_i T_s \ll 1$. Therefore the following approximation holds

$$e^{\lambda_i T_s} = \sum_{k=0}^{\infty} \frac{(\lambda_i T_s)^k}{k!} \approx 1 + \lambda_i T_s .$$

From this approximation and (1.27), we obtain

$$A_d \approx I + AT_s .$$

As for the input matrix B, the following approximation holds

$$B_d = A^{-1}\left(e^{AT_s} - I\right) B \approx A^{-1}(AT_s) B = BT_s$$

[1] The case of non-diagonalizable matrices is beyond the scope of the present work and will not be treated here.

Note that the sparsity patterns of $I + AT_s$ and BT_s are identical to those of A and B, respectively; only the diagonal entries of A are different, these, however, are always assumed to be free optimization parameters in our algorithm. What can be concluded from the previous calculations is that, in general, $(A)_{ij} = 0$ does not imply $(A_d)_{ij} = 0$; however, one can reasonably expect $(A_d)_{ij}$ to be much lower than the other elements on the i-th row and j-th column, provided that T_s is much smaller than the characteristic time constants of the system dynamics (the same applies for B and B_d). Such considerations can be readily verified by means of numerical tests.

The algorithms presented in this section are based on these arguments, indeed each algorithm chooses at each step only the largest elements of the (normalized) estimated A_d and B_d matrices, and it is therefore expected to disregard the entries corresponding to zeros in the continuous–time matrices.

1.2.6 Application examples

In this section we provide some results of PACTLS, CORE-Net[10] and TSNI[21] algorithms. In particular we show the performance obtained by these approaches for inferring gene regulatory networks form experimental gene expression data.

PACTLS and CORE-Net have been used to reconstruct a cell cycle regulatory subnetwork in *Saccharomyces cerevisiae* from experimental microarray data. We considered the model proposed by [40] for transcriptional regulation of cyclin and cyclin/CDK regulators and the model proposed by [41], where the main regulatory circuits that drive the gene expression program during the budding yeast cell cycle are considered. The network is composed of 27 genes: ten genes that encode for transcription factor proteins (ace2, fkh1, swi4, swi5, mbp1, swi6, mcm1, fkh2, ndd1, yox1) and seventeen genes that encode for cyclin and cyclin/CDK regulatory proteins (cln1, cln2, cln3, cdc20, clb1, clb2, clb4, clb5, clb6, sic1, far1, spo12, apc1, tem1, gin4, swe1 and whi5). The microarray data have been taken from [42], selecting the data set produced by the alfa factor arrest method. Thus, the raw data set consists of 27 genes and 18 data points. A smoothing algorithm has been applied in order to filter the measurement noise and to increase by interpolation the number of observations. The gold standard regulatory network comprising the chosen 27 genes has been drawn from the BioGRID database [43], taking into account the information of [40] and [41]: the network consists of 119 interactions, not including the self-loops, yielding a value of the sparsity coefficient, defined by $\eta = 1 - \#\text{edges}/(n^2 - n)$, equal to 0.87.

Fig. 1.1 shows the results obtained by PACTLS assuming four different levels of prior knowledge (PK) from 10% to 40% of the network. The performance is evaluated by using two common statistical indices (see [44], p.138):

- *Sensitivity* (Sn), defined as

  $$\text{Sn} = \frac{TP}{TP + FN},$$

  which is the fraction of actually existing interactions (TP:=true positives, FN:=false negatives) that the algorithm infers, also termed *Recall*, and

- *Positive Predictive Value* (PPV),

  $$\text{PPV} = \frac{TP}{TP + FP},$$

  which measures the reliability of the interactions (FP:=false positives) inferred by the algorithm, also named *Precision*.

To compute these performance indexes, the weight of an edge is not considered, but only its existence, so the network is considered as a directed graph. The performance of PACTLS is compared with one of the most popular statistical methods for network inference, dynamic Bayesian networks. For these purposes we used the software BANJO (BAyesian Network inference with Java Objects), a tool developed by Hartemink and coworkers [45], that performs network structure inference for static and dynamic Bayesian networks (DBNs). The performance of both approaches is compared in Fig. 1.1. In order to further validate the inference capability of the algorithms, the figure shows also the results obtained by a random selection of the edges, based on a binomial distribution: given any ordered pair of nodes, the existence of a directed edge between them is assumed true with probability p_r, and false with probability $1 - p_r$. By varying the parameter p_r in $[0, 1]$, the random inference algorithm produces results shown as the solid curves on the (PPV, Sn) plot in Fig. 1.1.

The performance of PACTLS is consistently significantly better than the

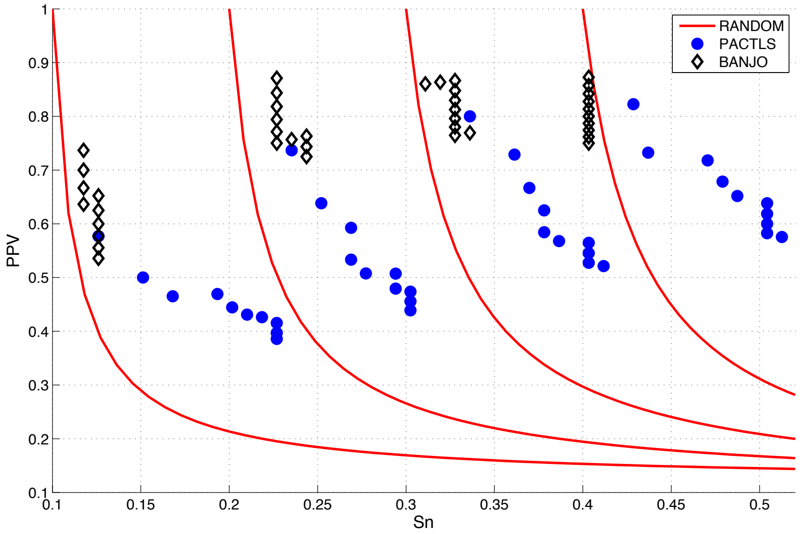

Figure 1.1: Results for the cell cycle regulatory subnetwork of *Saccharomyces cerevisiae* assuming different levels of prior knowledge (PK=10,20,30,40%).

method based on DBNs: the distance of the PACTLS results from the random curve is almost always larger than those obtained with the BANJO software, which is not able to achieve significant Sn levels, probably due to the low number of time points available. Moreover the results show that the performance of PACTLS improves progressively when the level of prior knowledge increases.

Fig. 1.2 shows the regulatory subnetwork inferred by CORE–Net, assuming 50% of the edges are *a priori* known. Seven functional interactions, which are present in the gold standard network, have been correctly inferred. Moreover, seven other functional interactions have been returned, which are not present in the gold standard network. To understand if the latter should be classified as TP or FP, the authors have manually mined the literature and the biological databases, and uncovered the following results:

- The interaction between *mbp1* and *gin4* is reported by the YEASTRACT database [46]: *mbp1* is reported to be a transcription factor for *gin4*;
- A possible interaction between *fkh2* and *swi6* is also reported by the

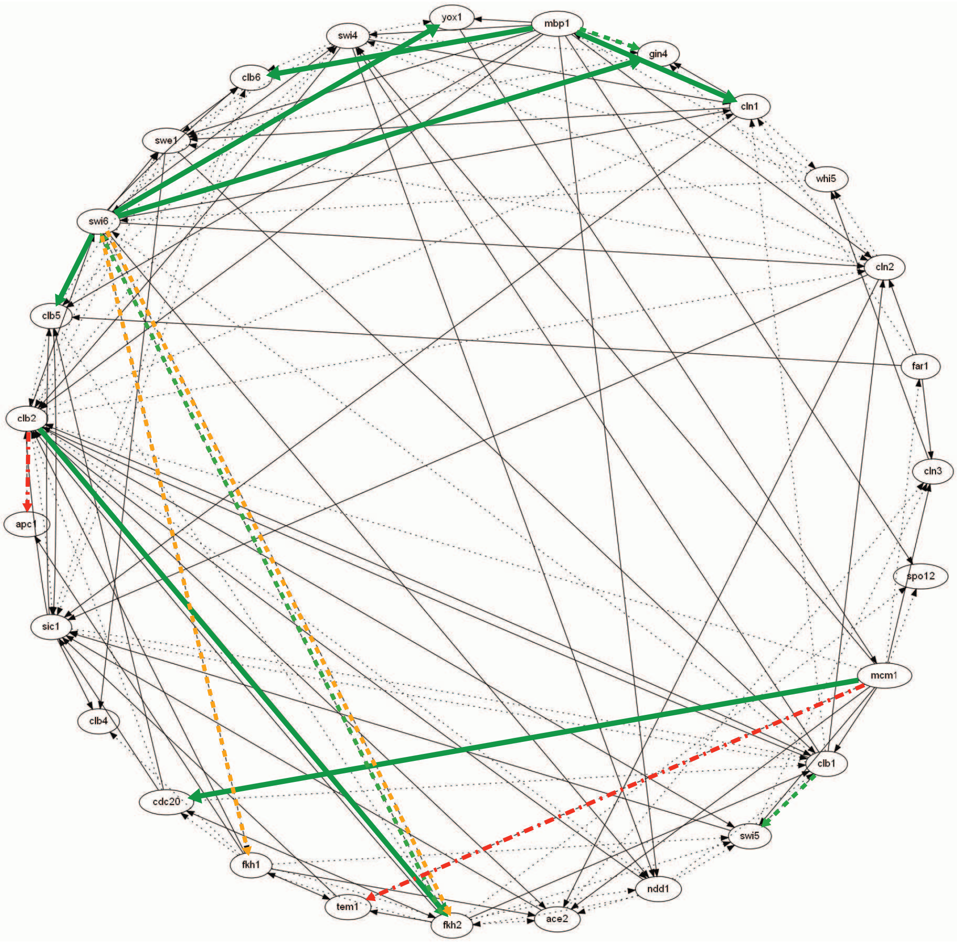

Figure 1.2: Gene regulatory subnetwork of *S. cerevisiae* inferred by CORE–Net with 50% of the edges *a priori* known (thin solid edges). Results according to the gold–standard network drawn from the BioGRID database: TP=thick solid edge, FN=dotted edge, FP=thick dashed edge. The thick green dashed edges are not present in the BioGRID database, however they can be classified as TP according to other sources. The FP thick orange dashed edges are indirect interactions mediated by *ndd1*. No information has been found regarding the interactions denoted by the thick red dash–dot edges.

YEASTRACT database: *fkh2* is reported to be a potential transcription factor for *swi6*);

- The interaction between *clb1* and *swi5* appears in Figure 1 in [41], where the scheme of the main regulatory circuits of budding yeast cell cycle is described.

Thus, these three interactions can be classified as TP as well and are reported as green dashed edges in Fig. 1.2.

Concerning the other inferred interactions, two of them can be explained by the indirect influence of *swi6* on *fkh1* and *fkh2*, which is mediated by *ndd1*: in fact, the complexes SBF (Swi4p/Swi6p) and MBF(Mbp1p/Swi6p) both regulate *ndd1* [40], which can have a physical and genetic interaction with *fkh2*. Moreover, *fkh1* and *fkh2* are forkhead family transcription factors which positively influence the expression of each other. Thus, the inferred interactions are not actually between adjacent nodes of the networks and have to be formally classified as FP (these are reported as orange dashed edges in Fig. 1.2).

Concerning the last two interactions, that is *clb2*→*apc1* and *mcm1*→*tem1*, since we have not found any information on them in the literature, in the absence of further experimental evidences they have to be classified as FP (reported as red dash-dot edges in Fig. 1.2).
The results obtained in this example confirm that the exploitation of prior knowledge together with the PA mechanism significantly improves the inference performances.

The algorithm TSNI has been tested for inferring a 9 gene subnetwork of the DNA-damage response pathway (SOS pathway) in the bacteria *Escherichia coli*. The network is composed of 9 genes: lexA, SsB, dinI, umuDC, rpoD, rpoH, rpoS, recF, recA. The gene expression profiles for all nine genes were obtained by computing the average of the three replicates for each time point following treatment by norfloxacin, a known antibiotic that acts by damaging the DNA. In order to assess the performance of TSNI, the inferred network is compared with the one identified in [5] by NIR and with the one obtained from the interactions known in literature among these nine genes (43 interactions, apart from the self–feedback). In Fig. 1.3 the results obtained by the TSNI algorithm for the *E. coli* time-series data are shown by plotting the average of r_{nz} (the ratio between the identified correct non zero coefficients with correct sign and the true number of non zero coefficients) versus r_z (the ratio between the identified correct zero coefficients and the true number of zero coefficients), obtained by comparing the predicted network of TSNI with the network obtained from literature. The cross on the plot shows the value of r_{nz} and r_z, which is obtained by comparing the network predicted by NIR in [5] with the network from the literature (NIR predicted 22 connections). When the information that there should be five connections for each gene is used, as in [5], and thus four elements in each row of the A matrix to be identified are set to zero, then TSNI finds 20 connections correctly (corresponding to the diamond in Fig. 1.3). The results of TSNI are similar to NIR, even if for TSNI only a single perturbation experiment and 5 time points are used, whereas for NIR nine different perturbation experiments are used and the matrix B is assumed to be known.

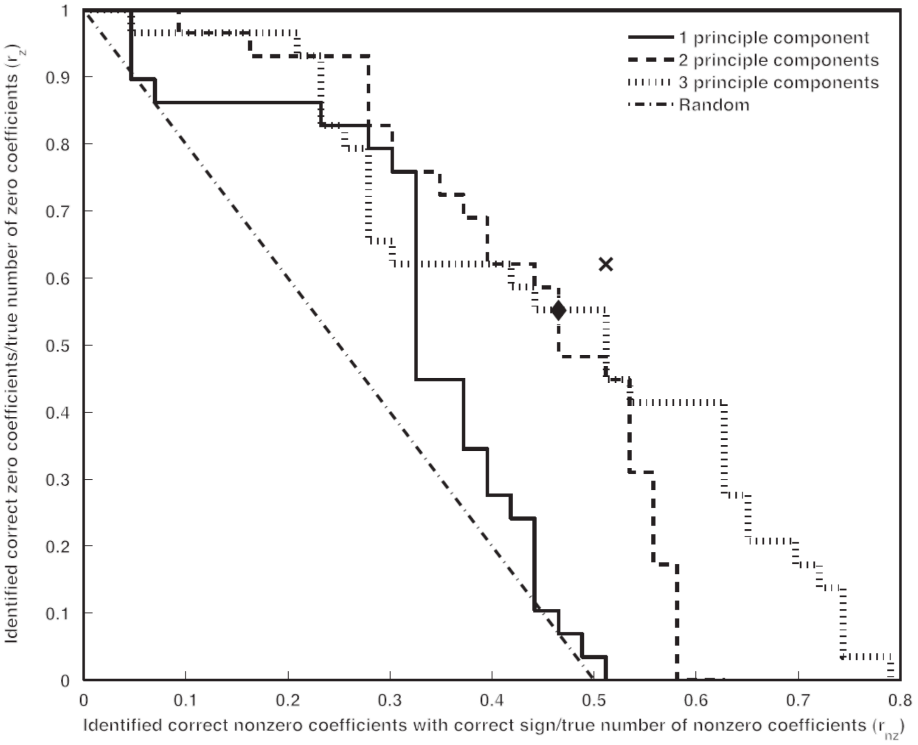

Figure 1.3: Results for the SOS pathway example obtained by TSNI and compared with NIR

1.3 Reconstruction methods based on nonlinear models

Many models for describing biological networks are nonlinear ODE systems, involving polynomial or rational functions or power-law approximations. This section deals with the reverse engineering approaches which are available for these types of nonlinear models.

1.3.1 Approaches based on polynomial and rational models

In the following the method developed by August and Papachristodoulou in [13] is illustrated. The procedure is applied for reverse engineering chemical reaction networks with mass action kinetics (polynomial functions) and gene regulatory networks with dynamics described by Hill functions (rational) from time-series data.

Chemical reaction networks

We consider the dynamical systems described as

$$\dot{x}(t) = Af(x)\,, \tag{1.28}$$

where $x(t) = (x_1(t), \ldots, x_n(t))^T \in \mathbb{R}^n$, the state variables $x_i,\ i = 1, \ldots, n$, represent the levels of the different species present in the system, $A \in \mathbb{R}^{n\times m}$ is the unknown matrix representing the connectivity structure of the different species and $f :\in \mathbb{R}^n \to\in \mathbb{R}^m$ is the vector of known functions which satisfy appropriate smoothness conditions to ensure local existence and uniqueness of solutions. Therefore, the relation (1.28) is linear in the unknown parameters and the main objective is to devise a procedure for reconstruction of the topology of the network represented by the matrix A, from the given experimental data.

The chemical reaction network model, described by the relation (1.5), is equivalent to the dynamical system in the form (1.28), with $A = YA_k$ and $f(x) = \Psi(x)$. Note that the unknown parameters, which determine the network structure, are in A.

For reverse engineering the matrix A, we consider the corresponding discrete-time system of (1.28) by using the Euler discretization:

$$x(t_{k+1}) = x(t_k) + (t_{k+1} - t_k)Af(x(t_k))\,. \tag{1.29}$$

This problem can be solved in a standard least squares setting as shown in Section 1.2.1, but in the presence of extra constraints on the entries of A it does not have a closed-form solution; indeed, it is necessary to take into account the problem of noise in the data and the sparseness of the network, features that can not be obtained by solving a least-squares problem. The problem can, however, be formulated as a Linear Program (LP), a convex optimization problem for which efficient algorithms are available that can treat large data sets efficiently and also deal with uncertainties in data or model parameters:

$$\begin{aligned}
&\min \| \, vec(A) \, \|_1 \\
&\text{s.t.} - \mu_k^- \leq -\hat{x}(t_{k+1}) + \hat{x}(t_k) + (t_{k+1} - t_k)Af(\hat{x}(t_k)) \leq \mu_k^+, \\
&\quad \mu_k^+ \geq 0, \mu_k^- \geq 0, \forall k, k = 1, \dots, p-1\,,
\end{aligned} \tag{1.30}$$

where $vec(A) \in \mathbb{R}^{nm}$ is a vector containing the entries of A, $\hat{x}$ is the set of measurements, μ_k^+ and μ_k^- are scalars which are as small as possible for all k to ensure that the data are in close Euler-fit with the model, thus making the approximation error as small as possible. Minimizing the l_1-norm then allows the computation of a sparse connectivity matrix [24, 47, 30].

In addition, by modeling the uncertainty in the measurements as

$$\begin{aligned}
&\tilde{x}(t_k) - \epsilon(k) \leq \hat{x}(t_k) \leq \tilde{x}(t_k) + \epsilon(k)\,, \tilde{f}(t_k) - \delta(k) \leq \hat{f}(\hat{x}(t_k)) \leq \tilde{f}(t_k) + \delta(k)\,, \\
&\epsilon(k), \delta(k), \tilde{x}(t_k), \tilde{f}(t_k) \geq 0\,, \forall k, A_{ij} \geq 0\,,
\end{aligned}$$

then a robust formulation of the LP (1.30) is given as

$$\begin{aligned}
&\min \| \, vec(A) \, \|_1 \\
&\text{s.t.} - \mu_k^- \leq -\tilde{x}(t_{k+1}) - \epsilon(k+1) + \tilde{x}(t_k) - \epsilon(k) + (t_{k+1} - t_k)A(\tilde{f}(t_k) - \delta(k)), \\
&\quad - \tilde{x}(t_{k+1}) + \epsilon(k+1) + \tilde{x}(t_k) + \epsilon(k) + (t_{k+1} - t_k)A(\tilde{f}(t_k) + \delta(k)) \leq \mu_k^+, \\
&\quad \epsilon(k), \delta(k), \tilde{x}(t_k), \tilde{f}(t_k), \mu_k^+, \mu_k^- \geq 0, \forall k, k = 1, \dots, p-1, A_{ij} \geq 0, \forall i, j\,.
\end{aligned} \tag{1.31}$$

Gene regulatory networks

Consider the model of the gene regulatory network described as

$$\dot{x}_i = \gamma_i + f_i(x) - d_i x_i \,, \qquad (1.32)$$
$$f_i(x) = \frac{\sum\limits_{j=1}^{n} b_{ij} x_j^{n_{ij}}}{1 + \sum\limits_{j=1}^{n} k_{ij} x_j^{m_{ij}}} \,,$$

where γ_i and $d_i > 0$ are the basal transcription and degradation/dilution rates, respectively, f_i are activation ($n_{ij} = m_{ij} > 0$) and repression ($n_{ij} = 0, m_{ij} > 0$) Hill input functions [48], k_{ij} and b_{ij} denote the contribution of the different transcription factors on the transcription rate.

The model (1.32) can be reformulated and cast in a form that allows identification using Linear Programming. Consider the corresponding discrete-time system of (1.32)

$$x_i(t_{k+1}) = x_i(t_k) + \Delta t(\gamma_i + f_i(x_i(t_k))) - d_i x_i(t_k)) \,, \qquad (1.33)$$

where $\Delta t = t_{k+1} - t_k$. If b_{ij}, k_{ij} and m_{ij} are unknown then (1.33) is not affine in the unknown parameters as is the case in (1.29). We rewrite (1.33) as follows:

$$(x_i(t_k)(1-\Delta t d_i) - x_i(t_{k+1}) + \Delta t \gamma_i) + (1 + \sum_j k_{ij} x_j^{m_{ij}}) + \Delta t \sum_j b_{ij} x_j^{\tilde{n}_{ij}} + \Delta t b_i = 0 \,, \qquad (1.34)$$

where, for $n_{ij} = \bar{n}_{ij} = 0$, $b_i = \sum\limits_j b_{ij} \bar{x}_j^{\bar{n}_{ij}} = \sum\limits_j b_{ij}$, whereas, for $n_{ij} > 0$, $\tilde{n}_{ij} = n_{ij}$. For all i, j, let an entry to matrix B be b_{ij} for which $n_{ij} > 0$, and let an entry of matrix K be k_{ij}. As before, given a set of measurements, $\hat{x}$, we can approximate the structure of the gene regulatory network determined by b_{ij}, b_i and k_{ij} if the Hill coefficients m_{ij} and n_{ij} are known and the basal production and degradation rates are known or considered uncertain but within a known range. For instance, we can try to recover B, K through the following LP:

$$
\begin{aligned}
&\min \parallel vec(B\,b\,K) \parallel_1 \\
&\text{s.t.} - \mu < (\hat{x}_i(t_k)(1-\Delta t d_i) - \hat{x}_i(t_{k+1}) + \Delta t \gamma_i) + (1 + \sum_j k_{ij}\hat{x}_j^{m_{ij}}) + \\
&\quad + \Delta t \sum_j b_{ij}\hat{x}_j^{n_{ij}} + \Delta t b_i < \mu\,, \\
&\quad \mu > 0, b_{ij}, k_{ij}, b_i \geq 0, \forall i, j, k, \\
&\quad 0 \leq \epsilon_{1i} \leq \gamma_i \leq \epsilon_{2i}, 0 \leq \varepsilon_{1i} \leq d_i \leq \varepsilon_{2i}, \forall i, \quad (*)\,,
\end{aligned} \tag{1.35}
$$

where the last requirements $(*)$ represent the case of uncertain production and degradation rates. Note that as for (1.32)

$$
k_{ij} = 0, \text{ if and only if}\, b_{ij} = 0 \,\text{or}\, b_i = 0, \forall i, j. \tag{1.36}
$$

In the following case, the solution of (1.35) violates the (1.36), that cannot be implemented in a LP:

- if $k_{ij} \neq 0$, $b_{ij} = 0$ and $b_i = 0$, then the production of X_i is not influenced by X_j, i.e., it is the same case as when k_{ij}=0;
- if $b_{ij} \neq 0$ and k_{ij}=0, then X_j enhances the production of X_i, i.e, it is the same case as when $k_{ij} \neq 0$;
- if $b_i \neq 0$ and $k_{ij} = 0 \quad \forall i$, then the production of X_i is not affected by X_j, i.e., it is the same case when $b_i = 0$.

1.3.2 Approaches based on S-systems

Several numerical techniques have been proposed in the literature for inferring S-systems from time series measurement; most of them use computationally expensive meta-heuristics such as Genetic Algorithms (GA), Simulated Annealing (SA), artificial neural networks function approximation or global optimization methods (see [49] and references therein).

Akutsu et al. in [50] developed a simple approach based on a linear programming method, named SSYS-1. Assume that $\frac{dX_i(t)}{dt} > 0$ in (1.6) at time t. By taking the logarithm of each side of $\alpha_i \prod_{j=1}^{n} X_j{}^{g_{i,j}} > \beta_i \prod_{j=1}^{n} X_j{}^{h_{i,j}}$, we have

$$\log \alpha_i + \sum_{j=1}^{n} g_{i,j} \log X_j(t) > \log \beta_i \sum_{j=1}^{n} h_{i,j} \log X_j(t)\,. \tag{1.37}$$

Since $X_j(t)$ are known, this is a linear inequality if we consider $\log \alpha_i$ and $\log \beta_i$ as parameters. The same is valid in the case of $\frac{dX_i(t)}{dt} < 0$. Therefore, solving these inequalities by using LP, we can determine the parameters. However, the parameters are not determined uniquely even if many data points are given, because the inequality can be rewritten as

$$(\log \alpha_i - \log X_j(t)) + \sum_{j=1}^{n} (g_{i,j} - h_{i,j}) \log X_j(t) > 0\,. \tag{1.38}$$

Therefore, only the relative ratios of $\log \alpha_i - \log \beta_i$ and $g_{i,j} - h_{i,j}$ are computed. However this information is useful for qualitative understanding of S-Systems. Since it seems that $g_{i,j} \neq h_{i,j}$ holds for most (i, j), the fact that $|g_{i,j} - h_{i,j}|$ is not small means that X_i is influenced by X_j.

Aa approach based on alternating regression (AR) was proposed as a fast deterministic method for S-system parameter estimation with low computational cost [51]. A method, inspired by AR and based on multiple linear regression and sequential quadratic programming (SQP) optimization, is proposed in [49] for identification of S-Systems models from time-series, when no information about the network topology is known. Furthermore, the algorithm is extended to the optimization of network topologies with constraints on metabolites and fluxes. This method is based on the substitution of differentials with estimated slopes and the minimization of the differences between two vectors obtained from multiple linear regression (MLR) equations. Consider the following relation

$$S_i(t_n) = PT_i(t_n) - DT_i(t_n)\,, n = 1, \ldots, N, \tag{1.39}$$

$$PT_i(t_n) = \alpha_i \prod_{j=1}^{M} X_j(t_n)^{g_{i,j}}\,, DT_i(T_n) = \beta_i \prod_{j=1}^{M} X_j(t_n)^{h_{i,j}}\,,$$

where $S_i(t_n)$ represents the estimated slope of metabolite i at time t_n, $PT_i(t_n)$, $DT_i(t_n)$ the production and degradation term vectors, respectively, and N is the number of time points and M the number of species (e.g metabolites). Because PT_i must be positive, the (1.39) can be rewritten as

$$\log PT_i = \log(ST_i + DT_i) \tag{1.40}$$

or in matrix form as

$$LV_{pi} = \gamma_i\,, \tag{1.41}$$

where

$$\begin{aligned} L &= \begin{pmatrix} 1 & \log(X_1(t_1)) & \cdots & \log(X_M(t_1)) \\ \vdots & \vdots & \ddots & \vdots \\ 1 & \log(X_1(t_N)) & \cdots & \log(X_M(t_N)) \end{pmatrix}, \\ V_{pi} &= [\log \alpha_i g_{i1} g_{i2} \ldots g_{iM}]\,, \\ \gamma_i &= \log(ST_i + DT_i)\,. \end{aligned} \tag{1.42}$$

By MLR, the production parameter vector V_{pi} can be obtained as

$$V_{pi} = (L^T L)^{-1} L^T \gamma_i \tag{1.43}$$

and substituting the (1.43) in (1.42) we obtain

$$L(L^T L)^{-1} L^T \gamma_i = \gamma_i\,. \tag{1.44}$$

Note that γ_i must be an eigenvector of the matrix $W = L(L^T L)^{-1} L^T$, with an eigenvalue equalling 1. Several standard algorithms to calculate the eigenvector of the matrix W directly were implemented, but none of them returned a satisfactory result. Therefore the task was reformulated as a minimization problem for the logarithm of the squared residuals between the right and left side hands in (1.44), to define this problem in matrix form with the cost function:

$$F = \log((\gamma_i - \hat{\gamma}_i)^T (\gamma_i - \hat{\gamma}_i)) \tag{1.45}$$

where $\hat{\gamma}_i = W\gamma_i$.

1.3.3 A case–study

In the following the applicability of the approach devised in [13] by reconstructing the glycolytic pathway of *Lactococcous lactis*, using the same experimental data from [52], is shown. Lactococcus lactis is a bacterium used in dairy industry for the production of cheese and buttermilk, mainly because of its capacity to convert about 95% of the milk sugar lactose (Lact) to lactic acid. The glycolytic pathway (or glycolysis) consists of chemical reactions that convert glucose (Glu) into pyruvate (Pyru). In the first step, glucose is converted into glucose-6-phosphate (G6P). A conversion of G6P into fructose-1,6-bisphosphate (FBP) follows, which is then converted sequentially to glyceraldehyde-3-phosphate (Ga3P), 3-phosphoglyceric acid (3-PGA) and phosphoenolpyrurate (PEP). Additionally, Glucose and PEP are converted directly to pyruvate and G6P. In [52], since measurement data for the intermediate Ga3P were unavailable, an additional rate denoting depletion of FBP was included. A simplified description of the pathway is illustrated in Fig. 1.4.

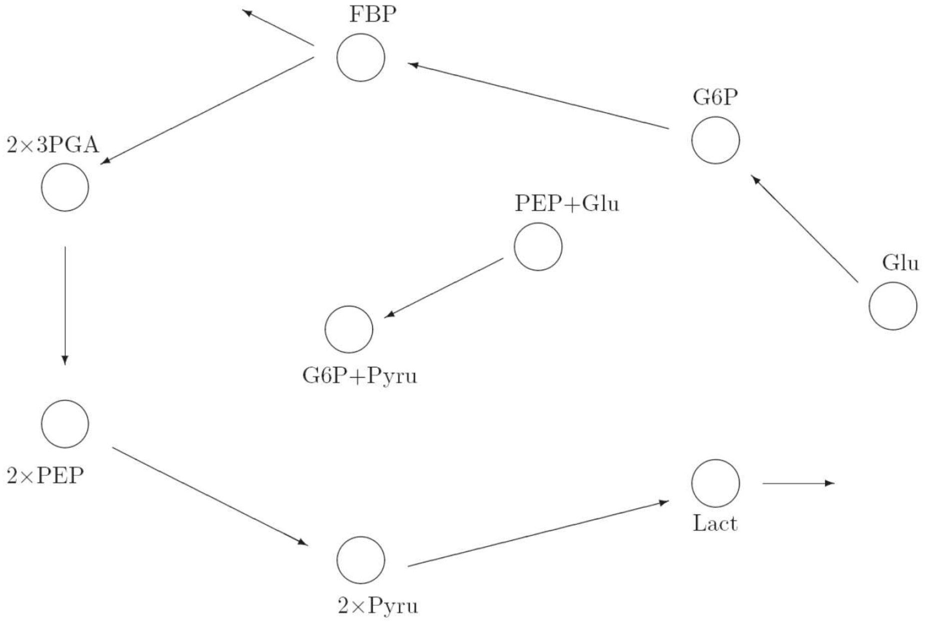

Figure 1.4: The glycolysis of *Lactococcous lactis*

In [13] the following complexes which participate in the chemical reaction network are assumed: Glu, G6P, FBP, 2×3PGA, 2×PEP, 2×Pyru and Lact. Therefore for the glycolytic pathway in the form of system (1.5) the matrix

Y, the vectors of the species (x) and of the functions ($f = \Psi$) are given by

$$Y = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 2 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}, x = \begin{pmatrix} [Glu] \\ [G6P] \\ [FBP] \\ [3PGA] \\ [PEP] \\ [Pyru] \\ [Lact] \end{pmatrix}, f(x) = \begin{pmatrix} [Glu] \\ [G6P] \\ [FBP] \\ [3PGA]^2 \\ [PEP]^2 \\ [Pyru]^2 \\ [Lact] \end{pmatrix}.$$

For reverse engineering the network topology described by the matrix A_k one of the possible LPs to be solved is the following:

$$\begin{aligned} &\text{given} \quad Y \\ &\min \parallel \begin{pmatrix} \hat{x}(t_1) - \hat{x}(t_2) + (t_2 - t_1) A f(\hat{x}(t_1)) \\ \vdots \\ \hat{x}(t_{p-1}) - \hat{x}(t_p) + (t_p - t_{p-1}) A f(\hat{x}(t_{p-1})) \end{pmatrix} \parallel_1 + \alpha \parallel vec(A) \parallel_1 \\ &\quad \text{s.t.} \quad A = Y A_k \\ &\qquad A_{k_{i,j}} \geq 0, i \neq j, \forall i, j, e^T A_k = 0 \,, \end{aligned} \tag{1.46}$$

where α is a nonnegative constant that allows to regulate the sparsity of A explicitly. In [13] the (1.46) is solved for $\alpha = 0\,, \alpha = 2\,, \text{and}\, \alpha = 3$, and the relative pathway is shown in Fig. 1.5, where we can see that the sparse reaction topology was almost reconstructed. Note that a gradual increase of α, for $3 \leq \alpha \leq 75$ does not change the network structure.

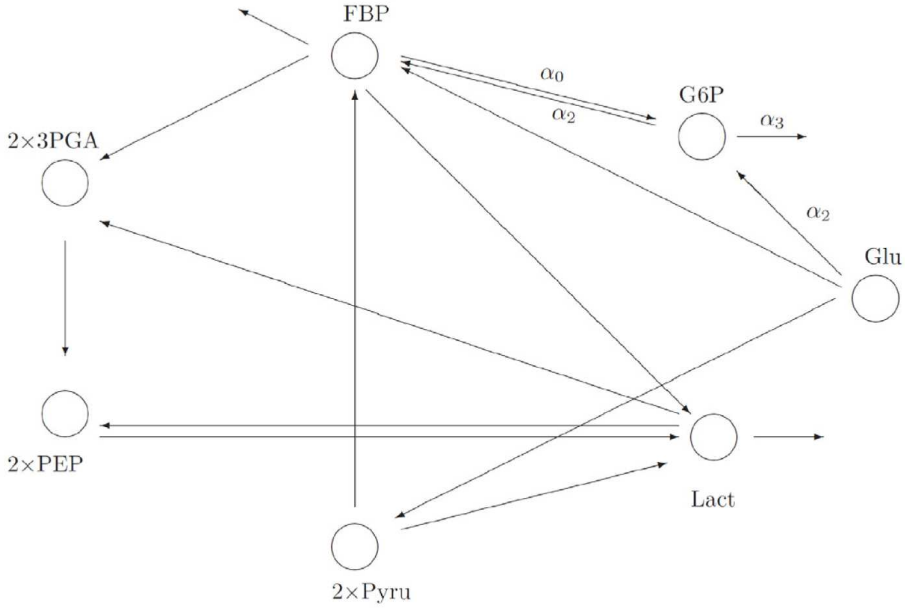

Figure 1.5: Reconstructed reaction topology for the glycolysis of *Lactococcous lactis*: two reactions obtained for $\alpha = 0$ and $\alpha = 2$ are marked with α_0 and α_2, one for $\alpha = 0$ with α_0, one for $\alpha = 3$ with α_3.

Bibliography

[1] P. A. Brown and D. Botstein, "Exploring the new world of the genome with DNA microarrays," *Nat. Genet.*, vol. 21, no. 1, pp. 33–37, 1999.

[2] R. J. Lipschutz, S. P. A. Fodor, T. R. Gingeras, and D. J. Lockhart, "High density synthetic oligonucleotide arrays," *Nat. Genet.*, vol. 21, no. 1, pp. 20–24, 1999.

[3] F. d'Alché Buc and V. Schachter, "Modeling and simulation of Biological networks," in *Proc. International Symposium on Applied Stochastic Models and Data Analysis (ASMDA'05)*, Brest, France, May 2005.

[4] M. Hecker, S. Lambeck, S. Toepfer, E. van Someren, and R. Guthke, "Gene regulatory network inference: Data integration in dynamic models – A review," *BioSystems*, vol. 96, pp. 86–103, 2009.

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

[6] D. di Bernardo, T. S. Gardner, and J. J. Collins, "Robust identification of large genetic networks," in *Proc. Pacific Symposium on Biocomputing (PSB'04): January 2004; Hawaii, USA.*, 2004, pp. 486–497.

[7] K.-H. Cho, J.-R. Kim, S. Baek, H.-S. Choi, and S.-M. Choo, "Inferring biomolecular regulatory networks from phase portraits of time-series expression profiles," *FEBS Letters*, vol. 580, no. 14, pp. 3511–3518, 2006.

[8] S. Kim, J. Kim, and K.-H. Cho, "Inferring gene regulatory networks from temporal expression profiles under time-delay and noise," *Computational Biology and Chemistry*, vol. 31, no. 4, pp. 239–245, 2007.

[9] S. Han, Y. Yoon, and K. H. Cho, "Inferring biomolecular interaction networks based on convex optimization," *Computational Biology and Chemistry*, vol. 31, no. 5-6, pp. 347–354, 2007.

[10] F. Montefusco, C. Cosentino, and F. Amato, "CORE–Net: Exploiting Prior Knowledge and Preferential Attachment to Infer Biological Interaction Networks," *IET Syst. Biol.*, vol. 4, no. 5, pp. 296–310, Sep. 2010.

[11] R. Guthke, U. Moller, M. Hoffman, F. Thies, and S. Topfer, "Dynamic network reconstruction from gene expression data applied to immune response during bacterial infection," *Bioinformatics*, vol. 21, pp. 1626–1634, 2005.

[12] J. Kim, D. G. Bates, I. Postlethwaite, P. Heslop-Harrison, and K. H. Cho, "Linear time-varying models can reveal non-linear interactions of biomolecular regulatory networks using multiple time-series data," *Bioinformatics*, vol. 24, pp. 1286–1292, 2008.

[13] E. August and A. Papachristodoulou, "Efficient, sparse biological network determination," *BMC Systems Biology*, vol. 3:25, 2009.

[14] J. Gim, H.-S. Kim, J. Kim, M. Cho, J.-R. Kim, Y. J. Chung, and K.-H. Cho, "A system-level investigation into the cellular toxic response mechanism mediated by AhR signal transduction pathway," *Bioinformatics*, vol. 26, no. 17, pp. 2169–2175, 2010.

[15] D. Marbach, R. J. Prill, T. Schaffter, C. Mattiussi, D. Floreano, and G. Stolovitzky, "Revealing strengths and weaknesses of methods for gene network inference," *Proc Natl Acad Sci USA*, vol. 107, no. 14, pp. 6286–6291, April 2010.

[16] M. A. Savageau, *Fundamentals of Medical Cell Biology.* Academic Press, New York, 1992, ch. A critique of the enzymologists test tube, pp. 45–108.

[17] R. Heinrich and S. Schuster, *The Regulation of Cellular Systems.* New York: Chapman & Hall, 1996.

[18] D. C. Montgomery and G. C. Runger, Eds., *Applied Statistics and Probability for Engineers.* New York, NY: John Wiley & Sons, 2003.

[19] D. di Bernardo *et al.*, “Chemogenomic profiling on a genomewide scale using reverse-engineered gene networks,” *Nat. Biotechnol.*, vol. 23, pp. 377–383, March 2005.

[20] M. Bansal, V. Belcastro, A. Ambesi-Impiombato, and D. di Bernardo, “How to infer gene regulatory networks from expression profiles,” *Molecular Systems Biology*, vol. 3, no. 78, 2007.

[21] M. Bansal, G. D. Gatta, and D. di Bernardo, “Inference of gene regulatory networks and compound mode of action from time course gene expression profiles,” *Bioinformatics*, vol. 22, no. 7, pp. 815–822, 2006.

[22] L. Ljung, *System Identification: Theory for the User.* Upper Saddle River, NJ: Prentice Hall, 1999.

[23] R. Bonneau, D. J. Reiss, P. Shannon, M. Facciotti, L. Hood, N. S. Baliga, and V. Thorsson, “The Inferelator: an algorithm for learning parsimonious regulatory networks from systems–biology data sets de novo,” *Genome Biology*, vol. 7, no. R36, 2006.

[24] S. Boyd and L. Vandenberghe, *Convex Optimization.* Cambridge University Press: Prentice Hall, 2004.

[25] G. H. Golub and C. F. V. Loan, “An analysis of the total least squares problem,” *SIAM Journal on Numerical Analysis*, vol. 17, no. 6, pp. 883–893, December 1980.

[26] S. Skogestad and I. Postlethwaite, *Multivariable Feedback Control: Analysis and Design.* Chichester, New York, Brisbane, Toronto, Singapore: John Wiley & Sons, 1996.

[27] T. Abatzoglou, J. M. Mendel, and G. A. Harada, “The constrained total least squares technique and its application to harmonic superresolution,” *IEEE Transactions on Signal Processing*, vol. 39, pp. 1070–1087, 1991.

[28] G. F. Franklin, J. D. Powell, and A. Emami-Naeini, *Feedback Control of Dynamic Systems.* Upper Saddle River, New Jersey, USA: Prentice Hall, 2002.

[29] C. Cosentino, W. Curatola, F. Montefusco, M. Bansal, D. di Bernardo, and F. Amato, "Linear matrix inequalities approach to reconstruction of biological networks," *IET Syst. Biol.*, vol. 1, no. 3, pp. 164–173, May 2007.

[30] A. Julius, M. Zavlanos, S. Boyd, and G. Pappas, "Genetic network identification using convex programming," *IET Syst. Biol.*, vol. 3, no. 3, pp. 155–166, 2009.

[31] K. Zhou, *Essentials of Robust Control.* Upper Saddle River, NJ: Prentice Hall, 1998.

[32] C. D. Meyer, *Matrix Analysis and Applied Linear Algebra.* Philadelphia, PA: SIAM Press, 2000.

[33] S. Boyd, L. El Ghaoui, E. Feron, and V. Balakrishnan, Eds., *Linear Matrix Inequalities in System and Control Theory.* Philadelphia, PA: SIAM Press, 1994.

[34] P. Gahinet, A. Nemirovski, A. J. Laub, and M. Chilali, *LMI Control Toolbox.* Natick, MA: The Mathworks, Inc., 1995.

[35] H. Jeong, B. Tombor, R. Albert, Z. N. Oltvai, and A. L-Barabási, "The large-scale organization of metabolic networks," *Nature*, vol. 407, pp. 651–654, 2000.

[36] A. Wagner, "The yeast protein interaction network evolves rapidly and contains few redundant duplicate genes," *Mol. Biol. Evol.*, vol. 18, pp. 1283–1292, 2001.

[37] D. E. Featherstone and K. Broadie, "Wrestling with pleiotropy: genomic and topological analysis of the yeast gene expression network," *Bioessays*, vol. 24, pp. 267–274, 2002.

[38] N. M. Luscombe, J. Qian, Z. Zhang, T. Johnson, and M. Gerstein, "The dominance of the population by a selected few: power-law behaviour applies to a wide variety of genomic properties," *Genome Biol.*, vol. 3, no. 8, 2002.

[39] R. Albert and A. L. Barabási, "Topology of Evolving Networks: Local Events and Universality," *Physical Review Letters*, vol. 85, no. 24, pp. 5234–5237, 2000.

[40] I. Simon, J. Barnett, N. Hannett, C. T. Harbison, N. J. Rinaldi, T. Volkert, J. J. Wyrick, J. Zeitlinger, D. K. Gifford, T. S. Jaakkola, and R. A. Young, "Serial regulation of Transcriptional Regulators in the Yeast Cell Cycle," *Cell.*, vol. 106, no. 1, pp. 697–708, 2001.

[41] J. Bähler, "Cell–cycle control of gene expression in budding and fission yeast," *Annu Rev. Genetic.*, vol. 39, no. 1, pp. 69–94, 2005.

[42] P. T. Spellman, G. Sherlock, M. Q. Zhang, V. R. Iyer, K. Andres, M. B. Eisen, P. O. Brown, D. Botstein, and B. Futcher, "Comprehensive Identification of Cell Cycle–regulated Genes of the Yeast Saccharomyces cerevisiae by Microarray Hybridization," *Mol. Biol. Cell.*, vol. 9, no. 1, pp. 3273–3297, 1998.

[43] C. Stark, B. Breitkreutz, T. Reguly, L. Boucher, A. Breitkreutz, and M. Tyers, "BioGRID: A General Repository for Interaction Datasets," *Nucleic Acids Research*, vol. 34, no. 1, pp. D535–D539, 2006.

[44] D. Olson and D. Delen, Eds., *Advanced Data Mining Techniques.* Springer, 2008.

[45] J. Yu, V. Smith, P. Wang, A. Hartemink, and E. Jarvis, "Advances to Bayesian network inference for generating causal networks from observational biological data," *Bioinformatics*, vol. 20, pp. 3594–3603, 2009.

[46] M. C. Teixeira et al., "The YEASTRACT database: a tool for the analysis of transcription regulatory associations in Saccharomyces cerevisiae," *Nucleic Acids Research*, vol. 34, no. 1, pp. D446–D451, 2006.

[47] M. Zavlanos, A. Julius, S. Boyd, and G. Pappas, "Identification of Stable Genetic Networks using Convex Programming," in *Proc. of the 2008 American Control Conference*, Seattle, WA, USA, 2008.

[48] U. Alon, *An Introduction to Systems Biology: Design Principles of Biological Circuits.* Boca Raton, FL: Chapman & Hall/CRC, 2006.

[49] M. Vilela, I. C. Chou, S. Vinga, A. T. Vasconcelos, E. Voit, and J. Almeida, "Parameter optimization in S-system models," *BMC Syst. Biol.*, vol. 2, no. 35, 2008.

[50] T. Akutsu, S. Miyano, and S. Kuhara, “Inferring qualitative relations in genetic networks and metabolic pathways,” *Bioinformatics*, vol. 16, no. 8, pp. 727–734, 2000.

[51]

[52] A. R. Neves, R. Ventura, N. Ansour, C. Shearman, M. J. Gasson, C. Maycock, A. Ramos, and H. Santos, “Is the Glycolytic Flux in *Lactococcus lactis* Primarily Controlled by the Redox Charge? Kinetics of NAD$^+$ and NADH Pools Determined in vivo by ^{13}C NMR,” *The Journal of Biological Chemistry*, vol. 277, no. 31, pp. 28 088–28 098, 2002.