BIOINFORMATICS

Vol. 19 Suppl. 2 2003, pages ii138–ii148
DOI: 10.1093/bioinformatics/btg1071

Gene networks inference using dynamic Bayesian networks

Bruno-Edouard Perrin[1,], Liva Ralaivola[1], Aurélien Mazurie[2], Samuele Bottani[2], Jacques Mallet[2] and Florence d'Alché–Buc[1]*

[1]Laboratoire d'Informatique de Paris 6, CNRS UMR 7606, 8 rue du capitaine Scott, 75015 Paris, France and [2]Laboratoire de Génétique Moléculaire de la Neurotransmission et des Processus Dégénératifs, CNRS UMR 7091, Hôpital La Pitié-Salpêtrière, 75013, Paris, France

Received on March 17, 2003; accepted on June 9, 2003

ABSTRACT

This article deals with the identification of gene regulatory networks from experimental data using a statistical machine learning approach. A stochastic model of gene interactions capable of handling missing variables is proposed. It can be described as a dynamic Bayesian network particularly well suited to tackle the stochastic nature of gene regulation and gene expression measurement. Parameters of the model are learned through a penalized likelihood maximization implemented through an extended version of EM algorithm.

Our approach is tested against experimental data relative to the *S.O.S. DNA Repair* network of the *Escherichia coli* bacterium. It appears to be able to extract the main regulations between the genes involved in this network. An added missing variable is found to model the main protein of the network. Good prediction abilities on unlearned data are observed. These first results are very promising: they show the power of the learning algorithm and the ability of the model to capture gene interactions.

Keywords: gene regulatory networks, structure extraction, expression profiles, dynamic Bayesian networks, Kalman filter, penalized likelihood, EM algorithm.

Contact: perrin@poleia.lip6.fr

INTRODUCTION

Gene regulatory networks form dynamic and distributed systems which control the expressions of the various genes in the cell. This article deals with the identification of genetic networks from kinetic expression profiles data within the framework of machine learning. This field offers indeed the theoretical and methodological context to tackle the problem of identifying these networks. Given a dynamic model of gene interactions, the problem is equivalent to learning the structural and functional parameters from time series representing the gene expression kinetics. Discovering the network structure is called the problem of the reconstruction of the interactions graph. Identifying all functional parameters knowing the network structure corresponds to the traditional problem of identifying a dynamic system with a prediction aim. The majority of researchers in this field are interested in one or the other of these problems. The objective of our work is to propose a general methodology to solve these two problems simultaneously and to show its relevance on experimental data. We choose the framework of dynamic Bayesian networks with continuous variables which are a particular type of dynamic graphic models. Various arguments are in favour of this choice. Many works show at first that stochastic phenomena play a significant role in molecular biology. A second argument comes from the quality of gene expression measurement techniques: data are in general noisy. Lastly, the probabilistic framework naturally opens the way for likelihood maximization algorithms, such as the 'Expectation-Maximization' algorithm (EM), which allows in particular to infer hidden parameters and deal with missing data. This framework can be extended using a Bayesian approach, which is useful for introducing prior knowledge on the network. Bayesian networks have already been used for genetic interactions inference, but with two significant restrictions that are removed in our work: most of the existing works concern static Bayesian networks, and discrete random variables are used to represent gene expression. The first restriction makes prediction task impossible while the second implies a huge number of parameters for the conditional probabilities laws describing the variables. Our approach relies on a dynamic and continuous modelling making prediction to be possible while restricting the number of parameters considerably.

In this framework, we develop a learning algorithm

*To whom correspondence should be addressed.

based on the EM algorithm and on the likelihood maximization. More precisely, a maximum a posteriori (MAP) principle is applied allowing to penalize the likelihood by a parsimony constraint on connections in the network. We use this technique for a linear dynamical system corresponding to a system of second order differential equations, allowing to model systems with inertia phenomena. We then study the learning algorithm on experimental data relative to the *S.O.S. DNA Repair* network of the *E.coli* bacterium. It enables us to show that our model is rich enough to fit the data sets. When adding a missing variable and learning all its parameters, we notice that its behaviour is akin to the main protein of the network. The prediction ability of our method is also highlighted in various learning experiments.

A STOCHASTIC INERTIAL MODEL FOR GENE NETWORKS

Gene network modelling

Under the term of additive models (D'haeseleer, 2000; Mjoslness *et al.*, 2000) are gathered models which determine the expression of a gene by using a ponderated sum of all expression levels of the others genes. The simpler among these models is purely linear and determines the expression level E_i of the gene i at instant t by

$$\frac{dE_i^t}{dt} = \sum_j w_{ij} E_j^t + b_i \qquad (1)$$

This model does not allow to extract non-linear interactions in the network, but can bring to light the most evident relations. A saturation function can be added to avoid divergences. Such a model has been tested by D'haeseleer (2000) and learned by minimizing a quadratic error criterion. By introducing prior knowledge on hidden variables nature, a plausible model has been inferred. Obviously, this approach has some limits which are inherent to such a linear system.

Neural networks (Weaver *et al.*, 1999) are models that are well adapted to learning temporal series. The prediction capacity of such models is good, but the amount of learning data needed is very large, and it therefore cannot be used on real data.

Bayesian networks can model the expression of each gene as a conditional probability function of the expressions of the other genes. They are therefore well suited for learning from noisy data. Some well-known algorithms for learning Bayesian networks exist (Heckerman, 1995), and new algorithms for learning very complex models have recently been proposed (Ghahramani *et al.*, 2000).

Static Bayesian networks have been used by Friedman *et al.* (2000) for analyzing gene networks. A linear gene interaction model is considered. Hartemink *et al.* (2001) focus on scoring the models they learn. Static Bayesian networks have also been considered by Pe'er *et al.* (2001), and are learned from perturbed expression profiles. Murphy *et al.* (1999) gives a very theoretical point of view on the problem of extracting gene interactions from dynamic data.

Static Bayesian networks cannot handle temporal information, and therefore dynamic Bayesian networks appear to be more adapted to dynamic data. Ong *et al.* (2002) have used dynamic Bayesian networks associated with a discrete model of regulation for modelling regulatory pathways in *E.coli*. An algorithm which identifies interaction networks from dynamic Bayesian networks coupled with a non-parametric regression method has recently been proposed by Kim *et al.* (2003). The introduction of prior knowledge in this approach does not seem to be easy.

A deterministic inertial model of gene regulation

The gene regulation model used in this work is based on the deterministic inertial model proposed by d'Alché *et al.* (2003) who implemented this model as a recurrent artificial neural network to infer parameters from data while the network structure was given. As it is not the main subject of this article, the model will be shortly introduced. This inertial model of gene regulation is based on second order differential equations governing the evolution of each gene of a network. The model allows to reflect the time delay observed during the regulation phenomenon. It is also able to capture eventual dampened oscillatory behaviours. The equations are of the form :

$$\frac{d^2E_i(t)}{dt^2} + 2\lambda_i\omega_i\frac{dE_i(t)}{dt} + \omega_i^2 E_i(t) = \sum_j w_{ij} E_j(t) \qquad (2)$$

where $E_i(t)$ is the expression level of gene i at time t, namely the quantity of mRNA produced by the gene at this time. λ_i plays the role of an absorption coefficient specific to gene i while ω_i acts as a natural frequency of gene i. If no oscillatory behaviour occurs, λ_i should be very high and ω_i will be very low. Biological justification of these two parameters will not be discussed furthermore here.

The model belongs to the family of additive regulatory models (D'haeseleer, 2000) since it assumes that regulatory genes have a cumulative effect on their regulated gene. The interactions between genes are thus captured by the left term of Equation 2 where w_{ij} defines the strength of the regulation of gene i by gene j. Although it could be discussed more in details, it should be emphasized that this model is rich enough to capture genetic expression dynamics, as it will be shown later. Following d'Alché *et al.* (2003), this n second order system of equations can be discretized to be numerically implemented. Assuming that

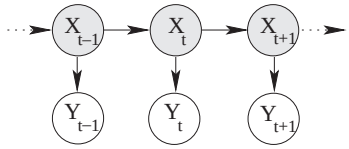

Fig. 1. Linear dynamical system. Dark nodes are hidden. Bright nodes are observed.

the measurement time unit is lower than the gene evolution characteristic time, we shall consider that continuous derivatives are replaced by: $\frac{\Delta E_i(t)}{\Delta t} = E_i(t+1) - E_i(t)$. We also shall define the vector

$$X_t = \left(E_1(t), \cdots, E_n(t), \frac{\Delta E_1(t)}{\Delta t}, \cdots, \frac{\Delta E_n(t)}{\Delta t} \right)' \quad (3)$$

The evolution of the network of n genes can be described by the equation

$$X_{t+1} = AX_t \quad (4)$$

with

$$A = \begin{bmatrix} \text{identity} & \text{identity} \\ W - \Omega^2 & \text{identity} - 2\Omega\Lambda \end{bmatrix} \quad (5)$$

where identity is the identity matrix of size $n \times n$, $W = (w_{ij})_{1 \leq i,j \leq n}$, $\Omega = \text{diag}(\omega_1, \ldots, \omega_n)$ and $\Lambda = \text{diag}(\lambda_1, \ldots, \lambda_n)$ [†].

Incorporating stochasticity in the network model using dynamic Bayesian networks

The model previously exposed is purely deterministic. To handle real data, a stochastic implementation is proposed. We assume that a gene network whose evolution is governed by Equation 4 is corrupted by intrinsic biological noise, as genetic expression is known to present some stochastic aspects (McAdams *et al.*, 1997). Moreover, actual states of the genes are not directly accessible: one can observe them through a measurement process which is also noisy. Let us notice that when informations are given about the process of measurement, it can be possible to assume more realistic probability distribution than the gaussian one. In this work, we have focused on the gaussian hypothesis, either for intrinsic or measurement noise.

Using the linear dynamical system formalism (cf. Fig. 1), these assumptions lead to the following model:

$$\begin{cases} X_{t+1} &= AX_t + \mathbf{u} \\ Y_t &= CX_t + \boldsymbol{\mu}_{obs} + \mathbf{v} \end{cases} \quad (6)$$

[†] In this article $\text{diag}(d_1, \ldots, d_n)$ is the diagonal matrix of size $n \times n$ whose diagonal elements are $d_1, \ldots, d_n$ and the others zero; for a square matrix M, $\text{diag}(M)$ is the diagonal matrix whose diagonal elements are the diagonal elements of M.

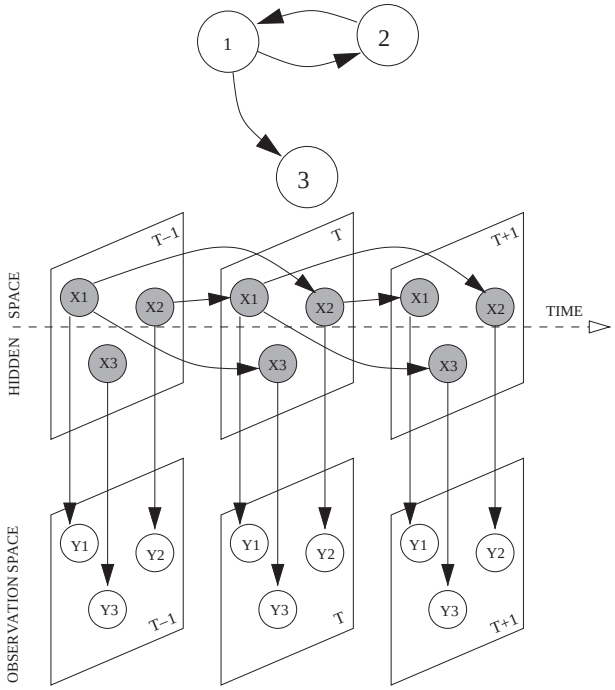

Fig. 2. A 3 genes network, represented as a static Bayesian network (top), and as a dynamic Bayesian network (bottom).

X_t is the hidden state of the gene network at instant t (cf. Equation 3), while Y_t is the observed state of the network, composed of all observations of gene expression levels. A is the transition matrix of Equation 5 and C the projection matrix [identity $\mathbf{0_{n,n}}$] of size $n \times 2n$, $\mathbf{0_{m,n}}$ being the zero matrix of size $m \times n$, and $\boldsymbol{\mu}_{obs}$ a measurement adjustement vector. Elements **u** and **v** are independant and identically distributed (i.i.d.) realizations of two Gaussian random variables with zero mean and variances σ_x^2 et σ_{obs}^2. **u** and **v** express the fact that both biological and measurement phenomena are stochastic. Variables X_t are usually said to be *hidden* because they only are accessible indirectly through observation of Y_t. We make the hypothesis that X_1 follows a Gaussian law of mean $\boldsymbol{\mu}_i$ and variance σ_i^2.

The proposed model (6) can be seen as a *dynamic Bayesian network* (DBN) with hidden nodes (cf. Fig. 2). It also is called a *Kalman filter model*.

Parameters can be learned using a generalization of *Expectation-Maximization* (EM) algorithm (Dempster *et al.*, 1977; Blimes, 1998) which is introduced in the following section.

Let us notice a great feature of our model : if some genes are to play a decisive role in the network, but if no expression data is available for them, it is possible to include them by modifying some of the model features. These genes will be called *missing variables*. Our framework is very adapted to handle them: our model is divided in two

Table 1. Filter and smoother equations. $X^0(1) = \boldsymbol{\mu}_i$, $\Sigma_1^0 = \sigma_i^2$identity, $\hat{X}(T) = X^T(T)$ et $\hat{\Sigma}(T) = \Sigma^T(T)$. $\hat{R}(t)$ and $\hat{R}^{t-1}(t)$ are used for determining the model parameters

Filter	Smoother
$X^{t-1}(t) = AX^{t-1}(t-1)$	$J_{t-1} = \Sigma^{t-1}(t-1)A'(\Sigma^{t-1}(t-1))^{-1}$
$\Sigma^{t-1}(t) = A\Sigma^{t-1}(t-1)A' + \sigma_s^2$identity	$\hat{X}(t-1) = X^{t-1}(t-1) + J_{t-1}(\hat{X}(t) - X^{t-1}(t))$
$\Sigma_e(t) = C\Sigma^{t-1}(t)C' + \sigma_{obs}^2$identity	$\hat{\Sigma}(t-1) = \Sigma^{t-1}(t-1) + J_{t-1}(\hat{\Sigma}(t) - \Sigma^{t-1}(t))J'_{t-1}$
$K_t = \Sigma^{t-1}(t)C'\Sigma_e^{-1}(t)$	$\hat{\Sigma}^{t-1}(t) = \hat{\Sigma}(t)J'_{t-1}$
$\mathbf{e}_t = Y_t - CX^{t-1}(t) - \boldsymbol{\mu}_{obs}$	
$X^t(t) = X^{t-1}(t) + K_t\mathbf{e}_t$	$\hat{R}(t) = \hat{\Sigma}(t) + \hat{X}(t)\hat{X}'(t)$
$\Sigma^t(t) = \Sigma^{t-1}(t) - K_tC\Sigma^{t-1}(t)$	$\hat{R}^{t-1}(t) = \Sigma^{t-1}(t) + \hat{X}(t)\hat{X}'(t-1)$

different spaces, the *hidden space* of X_t and the *observation space* of Y_t; there is only a projection matrix C to go from hidden state to observed state. Let us study a network composed of $n + h$ units, with available data for n genes, but the others h genes being unmeasured. Vectors Y_t of the observation space do not change and are always of size n. On the contrary, vectors X_t are now of size $2(n + h)$, being composed of all genes, including the missing ones. A is now a $2(n + h) \times 2(n + h)$ matrix, and the projection matrix C is [identity $\mathbf{0_{n,h}}$ $\mathbf{0_{n,n+h}}$] of size $n \times 2(n+h)$. Incorporating missing variables in our framework is therefore very easy, and it does not change anything to the use of the EM learning algorithm.

LEARNING THE PARAMETERS OF THE INERTIAL STOCHASTIC MODEL

The goal of learning the model is to identify the parameters that are the most adapted to the observed data. Parameters $\boldsymbol{\theta} := \{W, \Omega, \Lambda, \boldsymbol{\mu}_i, \sigma_i^2, \sigma_x^2, \boldsymbol{\mu}_{obs}, \sigma_{obs}^2\}$ of this model are learned from the observation of a time-course Yserie $= \{Y_1, \ldots, Y_T\}$ using the EM algorithm, which allows to handle the hidden variables X_t, so as the missing data, as seen previously.

The standard *Expectation* phase implements the *filter* and *smoother* processes, as described in Rosti *et al.* (2001). This phase is summarized in Table 1. It allows to determine directly the most probable states X_t given Yserie. The *Maximization* phase is different from the usual one, because of the specific features of A.

The auxiliary function $Q(\boldsymbol{\theta}, \boldsymbol{\theta}^{(k)})$, parametrized by $\boldsymbol{\theta}^{(k)}$, is defined as the expectation (operator $E[\cdot]$) of the penalized log-likelihood (Ormoneit *et al.*, 1998) with respect to Yserie: $Q(\boldsymbol{\theta}, \boldsymbol{\theta}^{(0)}) = E[\text{likelihood}^{pen}(\boldsymbol{\theta})|\text{Yserie}, \boldsymbol{\theta}^{(k)}]$. In the Bayesian framework, likelihood^{pen} corresponds to the maximum a posteriori (MAP) approach which takes into account a prior $P(\boldsymbol{\theta})$ on values of $\boldsymbol{\theta}$:

$$\text{likelihood}^{pen} = \log P(\text{Yserie}|\boldsymbol{\theta}) + \log P(\boldsymbol{\theta}) \qquad (7)$$

Machine learning theory ensures that choosing a prior that controls the complexity of the learned model produces a more efficient model in prediction (Girosi *et al.*, 1995). Moreover, from an optimization point of view, the control of the complexity reduces the presence of local optima of the likelihood. For our task, it is also a way to include prior biological knowledge about the parameters. We consider that $P(\boldsymbol{\theta}) = \alpha \exp(-\lambda||W||)$, where $||\cdot||$ denotes any derivable matricial norm. Such a law allows to favour sparse W matrices, and to consider low connectivity networks. This is particularly interesting for biological networks such as gene interaction networks, which are presumed to be sparse. In particular, we will choose for our experiments the L_1 norm defined by $||W||_1 = \sum_{ij} |w_{ij}|$. λ is called the regularization parameter: it weights the effect of the constraint applied to the parameters.

The M phase used for our algorithm determines $\boldsymbol{\theta}^{(k+1)}$ by making a gradient step in the direction $\nabla_{\boldsymbol{\theta}} Q(\boldsymbol{\theta}, \boldsymbol{\theta}^{(k)})$ from $\boldsymbol{\theta}^{(k)}$, parameters estimated after k EM iterations:

$$\boldsymbol{\theta}^{(k+1)} = \boldsymbol{\theta}^{(k)} + \eta \nabla_{\boldsymbol{\theta}} Q(\boldsymbol{\theta}, \boldsymbol{\theta}^{(k)}) \quad , \quad \eta > 0 \qquad (8)$$

More precisely, the M phase consists in the following computations:

- $\boldsymbol{\mu}_i^{(k+1)} = (1-\eta)\boldsymbol{\mu}_i^{(k)} + \eta\hat{X}(1)$
- $\sigma_i^{2(k+1)} = (1-\eta)\sigma_i^{2(k)} + \eta\left[\frac{1}{2n}\text{trace}(\hat{R}(1) - \boldsymbol{\mu}_i^{(k)}\boldsymbol{\mu}_i^{(k)\prime})\right]$
- $\boldsymbol{\mu}_{obs}^{(k+1)} = (1-\eta)\boldsymbol{\mu}_{obs}^{(k)} + \eta\left[\frac{1}{T}\sum_{t=1}^{T}(Y_t - CX_t)\right]$
- $\sigma_{obs}^{2(k+1)} = (1-\eta)\sigma_{obs}^{2(k)}$
  $+\eta\left[\frac{1}{nT}\text{trace}\left(\sum_{t=1}^{T}(Y_tY_t' - C\hat{X}(t)Y_t' - \boldsymbol{\mu}_{obs}^{(k)}Y_t')\right)\right]$

- $\sigma_x^{2(k+1)} = (1-\eta)\sigma_x^{2(k)}$
  $+\eta\left[\frac{1}{n(T-1)}\text{trace}\left(\sum_{t=2}^{T}(\hat{R}(t) - A^{(k)}\hat{R}'^{t-1}(t))\right)\right]$

For updating parameters $W^{(k)}$, $\Omega^{(k)}$ and $\Lambda^{(k)}$, it is useful to consider the G_k matrix defined by:

$$G_k = \frac{1}{\sigma_x^{2(k)}} \sum_{t=2}^{T} \left[\hat{R}^{t-1}(t) - A^{(k)}\hat{R}(t-1)\right] \qquad (9)$$

and more precisely the matrices of size $n \times n$ $G_k^{\ell\ell}$ and $G_k^{\ell r}$ which denotes respectively the G_k lower left and lower right submatrices. Parameters $W^{(k+1)}$, $\Omega^{(k+1)}$ and $\Lambda^{(k+1)}$ are computed according to:

$$W^{(k+1)} = W^{(k)} + \eta(G_k^{\ell\ell} - \lambda\nabla_W||W||)$$
$$\Omega^{(k+1)} = \Omega^{(k)} - 2\eta\text{diag}(G_k^{\ell\ell}\Omega^{(k)} + G_k^{\ell r}\Lambda^{(k)})$$
$$\Lambda^{(k+1)} = \Lambda^{(k)} - \eta\text{diag}(G_k^{\ell r}\Omega^{(k)})$$

Once the $A^{(k+1)}$ matrix is computed according to Equation (5), it is possible to proceed to a new E phase. At each EM step, the penalized likelihood increases, until a local maximum is reached.

EXPERIMENTAL RESULTS

Experimental data sets

We consider the *S.O.S. DNA Repair* network of the *E.coli* bacterium. This well-known gene network is responsible for repairing the DNA after a damage. The entire system is composed of about 30 genes regulated at the transcriptional level. Usually, when no DNA damage occurs, a master transcription factor *LexA* binds sites in the promoter regions of these genes, repressing all genes of the network. One of the *S.O.S.* proteins, *RecA*, acts as a sensor of DNA damage: by binding to single-stranded DNA, it becomes activated and mediates *LexA* destruction. The drop in *LexA* levels causes the de-repression (i.e. activation) of *S.O.S.* genes. Once damage has been repaired or bypassed, the level of activated *RecA* drops, *LexA* accumulates and represses the *S.O.S.* genes, and cells return to their initial state.

Experimental data have been provided by Uri Alon (they are downloadable on its homepage [‡]). Data are expression kinetics of the main 8 genes of the *S.O.S. DNA Repair* network of *E.coli*. The measurement technology is based on the property of the GFPs (green fluorescent proteins). Alon *et al.* have developed a system for obtaining very precise kinetics (Ronen *et al.*, 2002). Measurements are done after irradiation of the DNA at the initial time with UV light. Four experiments are done for various light intensities (Exp. 1&2 : 5 Jm^{-2}, Exp. 3&4 : 20 Jm^{-2}).

[‡] http://www.weizmann.ac.il/mcb/UriAlon/

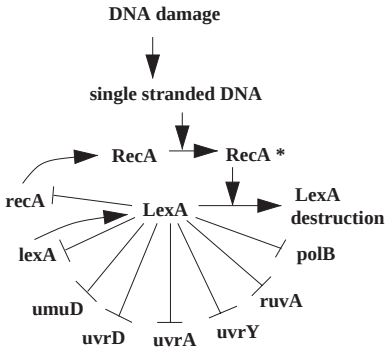

Fig. 3. *S.O.S. DNA Repair* network. Activations are represented by arrows (→) and inhibitions by T (⊣). Genes initials are in lower cases, proteins in capital letters.

Each experiment is composed of 50 instants evenly spaced by 6 minutes intervals, and 8 genes are monitored: *uvrD*, *lexA*, *umuD*, *recA*, *uvrA*, *uvrY*, *ruvA* and *polB*.

We first explain why the data are useful for our purpose and can be incorporated in our model.

Alon *et al.* monitor each *S.O.S.* gene separately, adding its promoter to a gfp sequence in a plasmid and incorporating the plasmids in irradiated *E.coli*. The quantity of present GFP, which is indirectly measured by the amount of fluorescence, is therefore proportional to the quantity of the corresponding *S.O.S.* protein.

The first hypothesis made by Alon *et al.* is that the GFP protein is very stable during the experiments: it is justified when comparing the typical GFP stability to the experiments length (300 minutes). By taking the derivative of the fluorescence amount with respect to time, Alon *et al.* have therefore access to the instantaneous protein production rate, since no protein degradation occurs during the experiments.

By making the standard hypothesis that the protein production rate is proportional to the corresponding mRNA production rate, they consider that the derivatives of the fluorescence amounts are proportional to the promoter activity of the genes. It is important to notice that this hypothesis is not so hard as usual in this case, because all proteins are the same (GFP protein). One more difficult point is why Alon et al. divide also by the OD, but we will not discuss this point here. Figure 3b of Ronen *et al.* (2002) indicates these promoter activities.

To use these data in our model, we have to make a strong hypothesis on the stability of the mRNAs: we consider that mRNA molecules are degraded immediately after their production. Actually mRNA persistance depends on the nucleotide sequence; all mRNAs having the same sequence here because of the experimental technique, they all have the same persistance. It is hence sufficient to

make the hypothesis that the turnover of the GFP mRNAs is very fast to ensure that there is a high unstability of the mRNAs. We therefore consider that the instantaneous promoter activity of each gene is also proportional to the present quantity of corresponding mRNA. A very simple analogy can be made to understand our reasoning : let us imagine water flowing from a tap into a bucket in the middle of a desert. This desert is so hot that the water *quite* instantaneously evaporates in the open air... By measuring the variations of the water level in the bucket, one can directly have access to the variations of the incoming flow, since there is a perfect proportionality between these two quantities. In our case, the flow of incoming water can be treated as the promoter activity, whereas the water volume in the bucket is the present quantity of mRNA. The quantity (dGFP/dt)/OD which indicates the promoter activity is therefore proportional to the mRNA quantity present in the *E.coli* strain.

We hence are able to consider that the data provided by Alon *et al.* directly indicate the observed mRNA quantities (also called expression levels) corresponding to each *S.O.S.* gene. The downloaded data consists in four 8×50 matrices corresponding to the four experiments. The t^{th} column of a given matrix is considered to be the observation vector Y_t, and the entire matrix will be considered as the time course $\{Y_1, \ldots, Y_{50}\}$.

Experiments

We have proceeded to several learning experiments on the data provided by Alon *et al.* The influence of the regularization parameter λ has been studied. For each data set, we have made successive learnings with λ =0, 1, 5, 10, 50, 100, 500, 1000 and 5000. In each case, we have introduced 0, 1 and 2 missing variables. The EM algorithm stops after 100 iterations, which is sufficient, since not discussed here previous experiments have shown that after 80 iterations, more than 95% of the parameters will not change of more than 1%, which does not change significantly the learned model. Parameters are initialized as follows: each coefficient of W is randomly initialized between 0.01 and 0.01, Λ and Ω are chosen from the assumption that unoscillatory behaviour is observed ($\lambda_i \simeq 1$ and $\omega_i \simeq 0 \; \forall i$), each coefficient of μ_i and μ_{obs} between 0 and 1, and finally σ_i^2, σ_x^2 and σ_{obs}^2 between 0 and 0.1. The main crucial point seems to be the initialization of W, Λ and Ω. 50 different learnings under each condition have hence been made: this *multiple random starting points* technique is widely used to handle the problem of likelihood local maxima. These $4 \times 9 \times 3 \times 50 = 5400$ learnings have been made using Java on a AMD Duron, 1.20 GHz in 28 hours and 25 minutes. The mean computing time is 18.94 seconds per learning.

Capturing the network dynamics

Capturing the dynamics characteristics. Our method is able to capture the network dynamics. Figure 4 allows to compare the real profiles of the 8 genes relative to the second experiment and their simulated profiles corresponding to the learned model for $\lambda = 100$. It is important to notice that these simulated profiles are *mean profiles*, since the variances associated to the model (σ_i^2, σ_x^2, σ_{obs}^2) are not taken into account in this simulation. The behaviour of the network according to these elements will be discussed later. The variations of the most expressed genes of the network (*recA*, *lexA*, *uvrA*, and to a lesser extent *umuD* and *uvrD*) are finely modelled. One can notice that the respective maxima of these genes are reached at different instants whose succession is essential to explain the functioning of the *S.O.S.* response system: this succession is respected in the learned model.

No noteworthy dose effect has been observed when comparing results of the different experimental conditions. It only appears that gene expressions levels are higher when the irradiation intensity increases, but the respective maxima and variations do not change between the four experiments, indicating that the *S.O.S.* system has the same type of answer for both experiment intensities.

Capturing the stochastic phenomena. One of the characteristics of our model is its ability to represent stochastic phenomena: one could wonder if the stochastic learned parameters are compatible with the real data. On Figure 5, we have represented five simulated profiles of *lexA* and *recA* using the model learned from the second data set with $\lambda = 50$.

For simulating such profiles, we use Equations 6. At each time point, **u** and **v** are randomly choosen according to their Gaussian distribution with variances σ_x^2 and σ_{obs}^2. The system is also randomly initialized according to $X_1 \sim \mathcal{N}(\mu_i, \sigma_i^2 I)$. We also have represented the mean profiles as in Figure 4 setting the variances to 0, and the measured profiles of the second data set. One can notice that the measured profile and the mean simulated profile do not merge: nevertheless, the measured profile is located in the enveloppe of simulated stochastic profiles (at least during the decrease). This shows that the learned variances are compatible with real variances.

For space commodity, we do not have plotted here such simulated profiles for all the genes under others conditions. Nevertheless experiments allow us to conclude that model variances associated with other learned model parameters are in accordance with the observed profiles. Our learning technique associated with the model choice is hence able to capture the mean dynamics of the network, so as to take into account stochastic phenomena, either due to measurement noise or to regulation itself.

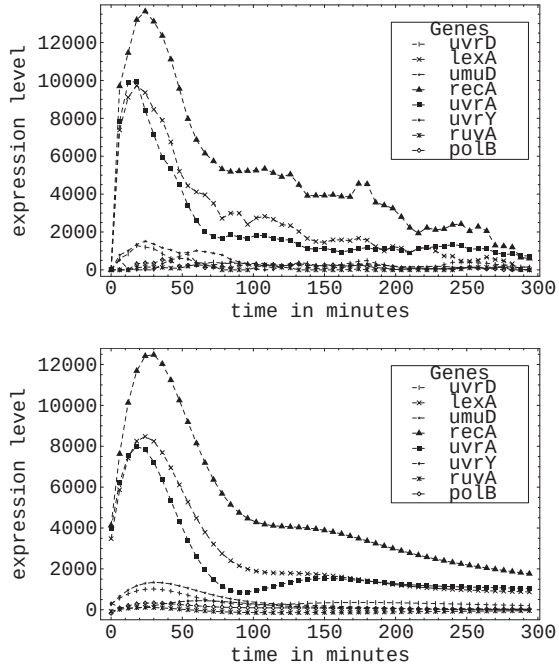

Fig. 4. Top: expression profiles of the 8 genes corresponding to the second second data set. Bottom: learned profiles from this data set for $\lambda = 100$. The vertical scale has no absolute meaning, since we took the (dGFP/dt)/OD values of Ronen *et al.* (2002), because they are proportional to the expression levels of the genes as explained previously. It is important to notice that these simulated profiles are *mean profiles*, since the variances associated to the model are not taken into account in this simulation.

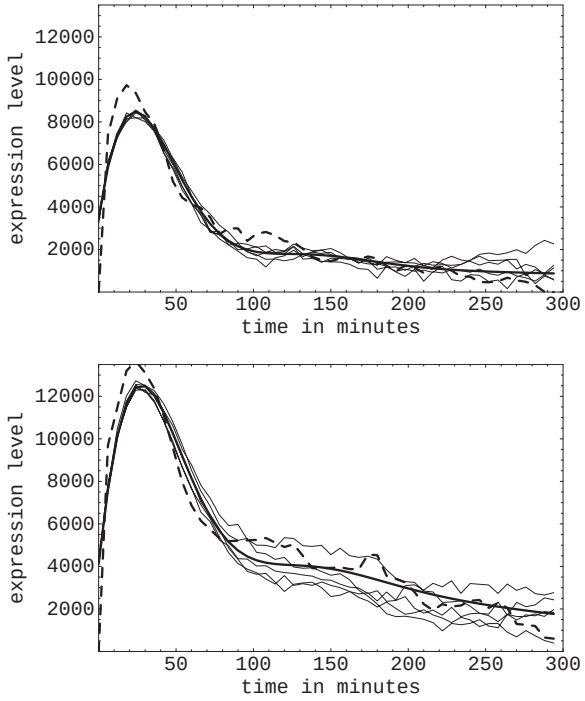

Fig. 5. Profile of *lexA* (top) and *recA* (bottom) under the same conditions as in Figure 4. The thick dotted line is the measured profile given in the data set; the thick continuous line is the mean learned profile already represented on Figure 4; thin continuous lines are simulated profiles, taking into account the learned model variances.

Structure extraction

Principle. It has been shown previously that our model is based on the assumption that regulation is an additive phenomenon. It is obviously biologically not true; nevertheless, as a first approach, it is not totally insane, and at least it has the magnificent advantage to be simple. Such works as (D'haeseleer, 2000) have used such an additive regulation model. We also point out the fact that the identification method, and not the model, is the mean novelty of our work.

This assumption lets us associate a biological meaning with each w_{ij} parameter according to this simple rule :

- $w_{ij} > 0$: gene j activates gene i
- $w_{ij} < 0$: gene j inhibits gene i
- $w_{ij} = 0$: gene j does not regulate gene i

After each learning, a W matrix is identified. According to the previous assumption, it should directly indicate the regulations in the network. Because of the presence of local maxima of the likelihood function, the identified W is not always the same at each experiment. We now show how regularization can alleviate the problem.

Regularization influence. The regularization technique is based on the simple idea that gene networks are known to be usually sparse: most of the genes have few regulators, and in turn regulate few genes. This regularization is very standard in the machine learning framework and is known to favour such sparse networks, which is biologically motivated in our case. One could object that the regularization term does not encourage sparseness, but simply low norm for W, so as a matrix with many weak connections can be as favorable as one with few strong ones. In fact, it does not appear to behave like that. Of course, each w_{ij} coefficient will decrease as the regularization parameter λ increases, but some coefficients decrease more slowly than others.

In order to illustrate this phenomenon, which proves the efficiency of our penalizing term, we have made some statistics on our experimental results. For each data set, we have studied the W matrix learned for three different regularization parameters $\lambda = \{0, 100, 1000\}$ with no added hidden variable. We have computed the mean and

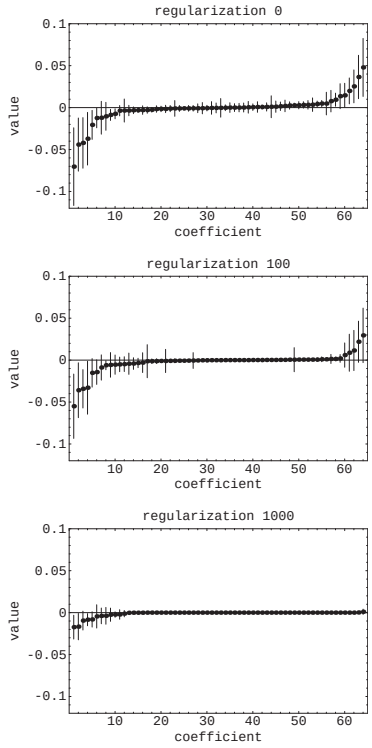

Fig. 6. *W* coefficients learned from the first data set, using respectively regularization 0, 100 and 1000. Coefficients are ordered by their mean on the 50 learnings. Error bars indicate their standard deviation.

standard deviation of all coefficients, and plotted them on Figure 6. Coefficients are ordered by mean. The plotted figure shows the results obtained for the first data set, but the others are similar.

It seems obvious that the curve flattens as the regularization parameter λ increases. This results proves that our regularization term encourages a real sparseness, and not only low coefficients. Of course, one can notice that all means decrease with λ, but some coefficients remain stricly non zero: this is the main goal of our regularization. Moreover, we also can notice that standard deviations decrease with λ: it should suggest us that a good regularization decreases the search space dimension, and in consequence the number of local maxima of the likelihood function. Nevertheless, when λ is too high, all coefficients tend to zero, so that all interesting informations about regulations vanish.

There are several ways to select or learn λ value. If numerous independant time series were available, it would be possible to apply cross-validation to select its right value. Another method consists in applying a full Bayesian approach by assuming a prior on λ and incorporating in the algorithm a learning stage for λ. In our framework, the regularization parameter can be chosen according to the average degree k of the identified graph. The average degree of a graph is given by $k = \frac{2M}{N}$ where N is the number of nodes, and M the number of arcs. It indicates the average number of arcs bound to a node. Some data are available about the topological structure of transcriptional networks: Shen-Orr *et al.* (2002) give 577 interactions for 116 transcription factors in *E.coli* using the RegulonDB database, which leads to $k = 9.95$, while Guelzim *et al.* (2002) propose a yeast transcriptional network composed of 491 genes and 909 transcriptional interactions giving $k = 3.70$. Unfortunately, the *S.O.S.* network has a very particular *star-like* topology, and is much smaller. We hence cannot use these average degrees for finding the best regularization. For this experiment, we hence use the average degree of the actual network. Figure 3 gives $k = 2.25$ for transcriptional regulations in the *S.O.S.* network considering that there are 9 transcriptional regulations between the 8 genes. As we find $k = 4.25$ for $\lambda = 0$, $k = 2.25$ for $\lambda = 100$, and $k = 1$ for $\lambda = 1000$ (the way to obtain the number of identified regulations is discussed further), we will consider that the optimal regularization parameter is 100. This regularization allows to make a compromise between favouring the sparseness of W and keeping information about regulations.

Identifying regulations For each data set, with $\lambda = 100$, 50 learnings starting from random starting points are done as explained previously. The learned values of each parameter w_{ij} are distributed with mean μ_{ij} and variance σ_{ij}^2. The mean and variance of the means of all 64 coefficients named μ and σ^2 can also be computed. Coefficients are then discretized into four classes according to their mean and standard deviation:

- class `[+]` : $\mu_{ij} > \mu + \sigma$ and $\sigma_{ij} < |\mu_{ij}|$
- class `[-]` : $\mu_{ij} < \mu - \sigma$ and $\sigma_{ij} < |\mu_{ij}|$
- class `[0]` : $|\mu_{ij}| < \sigma$ and $\sigma_{ij} < \sigma$
- class `[X]` : others coefficients

Classes are built to represent respectively probable activations, probable inhibitions, probable absences of regulation, and probable presences of unknown regulations. The total number of regulations in the network is obtained adding the number of coefficients in the classes `[+]`, `[-]` and `[X]`.

Figure 7 shows the identified discretized *W* matrix using the first data set. One can notice that 9 probable regulations are identified in the network, which leads to an average degree of 2.25, as said previously. Inhibitions of *lexA*, *recA* and *uvrA* by *lexA* itself are well identified (second column). The activation of *lexA* by *recA* is also identified (W_{24}). False regulations due to *umuD* and *uvrA* (columns

$$\begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & - & + & + & - & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & - & 0 & 0 & - & 0 & 0 & 0 \\ 0 & - & 0 & X & X & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix} \begin{matrix} uvrD \\ lexA \\ umuD \\ recA \\ uvrA \\ uvrY \\ ruvA \\ polB \end{matrix}$$

Fig. 7. W identified after 50 learnings with $\lambda = 100$ on the first data set. Discretization process is described in the article. The jth column shows all identified regulations exerced by jth gene on other genes. Inversely, the ith row shows all regulations the ith gene is submitted to. Genes order is on the right.

3 and 5) are identified (these genes are target genes, and do not act as regulators). A unknown regulation of *uvrA* by *recA* is identified: it could correspond to the indirect regulation $recA \rightarrow RecA \dashv LexA \dashv uvrA$.

For further comparisons between an identified network and the actual network, we have to specify the perfect W matrix which should correspond to the *S.O.S.* network. Of course, as some regulations are not directly transcriptional (for example the activation of *uvrA* by *recA* seen previously) the choice is not immediate. We shall consider that all parameters of the second column, indicating inhibitions on all genes by *lexA* have to be discretized in the class `[-]`, while the activation of *lexA* by *recA* is represented by W_{24} in `[+]`. Other parameters of the fourth columns are not defined precisely and can be either in `[0]` or `[+]`, because the learning is likely to identify undirect regulations.

Structure extraction can be viewed as an information retrieval task. We can transpose *Recall* and *Precision* measures usually used as quality measures in document retrieval to our field of interest. *Recall* defines the number of true oriented interactions predicted as fraction of all existing interactions. *Precision* defines the number of true oriented interactions predicted as fraction all the interactions predicted. Precision thus defines the level of 'noise' in the information presented to the user.

But, it should be emphasized that structure extraction not only aims at finding regulations, but also at identifying absence of interactions. We should hence also measure the number of true predicted coefficients as fraction of all interactions between genes (regulations and non regulations). We define this number as the *Generalized precision*.

Recall mean on all 4 data sets is 0.66, with a standard deviation of 0.056; *Precision* gives a mean of 0.57, with a standard deviation of 0.083; *Generalized precision* is 0.87 associated with a standard deviation of 0.023. For a random matrix, one should notice that *Recall* = 0.60, *Precision* = 0.20, and *Generalized precision* = 0.31. *Generalization* and *Generalized performance* are significantly high. *Recall* seems to be quite low, but we have to remember that the true 'score' of our method is given by the *Generalized precision*, as biologists are not only interested in regulations, but also in the absence of regulations. However the value of the *Recall* needs a comment: it can be enlightened by the fact that the experimental UV light shock was not sufficient to lead to the functioning of all *S.O.S.* genes. Figure 4 (top) shows that several genes were not induced during the experiment. These genes are activated only when the damage is sufficiently high.

In order to evaluate the similarity between networks identified using the various data sets, the proportion of equal coefficients between these networks are computed, giving a similarity mean of 89% with a standard deviation of 11%. It should be emphasized that these similarities are high, comparing with 25% obtained with a random matrice. These high similarities show that several experiments on the same underlying network let similar networks be identified.

Influence of missing variables

When 1 or 2 missing variables are added, the identified regulations between other genes remain the same as before. Missing variables are found to regulate and also to be regulated by *lexA*, *recA*, *uvrA* and themselves, but regulation coefficients have a large variance, so that they all are discretized in the class `[X]`.

When only one missing variable is introduced, simulations of its level evolution are done using each of the 50 learned models. For 22 of these models, the simulated profile is akin to the *LexA* protein concentration profile under the same experimental conditions measured in Sassanfar *et al.* (1990). Moreover, when considering only this cluster of models, variances concerning the regulations involving the added missing variable are lower than previously, so that W discretization shows that missing variable is inibited by *recA* and inhibits *lexA* and itself. An attractive hypothesis is that the added missing variable 'takes the role' of the protein *LexA*. Further experiments need to be done to show if this hypothesis is acceptable. In particular, the meaning of the model parameters when a protein is concerned has to be clarified.

Prediction

Within the machine learning theory, the choice of a model and the identification of its parameters should lead to generalization ability. For sequential data, this ability can be measured by two properties: the model ability to make k-step ahead prediction and its ability to reflect dynamics

of other i.i.d. sequences. In the context of the available data, our learned model easily succeded in making k-step prediction using the first 2/3 data points for training and the 1/3 lasting time for prediction. This does not prove very much since prediction is quite easy (back to equilibrium).

We also used one time course as training data and others as test data. Data provided by Alon *et al.* are particularly adapted to this type of experiments, because the four available time-series concern the same network. Two kinds of prediction abilities are evaluated. The first one is called the *one time step prediction*: for each instant, an expectation of the observation at the next instant is computed using the test data observation and the model learned from the training data. The second prediction ability is called the *multi step prediction*: a filtering phase is done on the 10 first instants of the testing data to estimate its hidden state at instant 10, and the model learned from the training data is then used to predict the kinetic profiles of the genes until instant 50 without accessing to their true observation values.

Very good correlations between *one step predicted* sequences and actual sequences are achieved, with a mean of 0.968, and a standard deviation of 15.6 10^{-3}. Correlation between the last part (instants 11 to 50) of *multi step predicted* sequences and actual sequences has a mean of 0.654 and a standard deviation of 0.171. Even if the predicted part of the sequence is the easiest because of the monotonous decrease of gene profiles after their peak, such correlations prove that the learned models are able to predict further evolution.

DISCUSSION AND FUTURE WORK

We introduced a general approach for identifying gene networks based on the use of linear dynamic Bayesian networks with continuous variables. The learning algorithm maximized the likelihood regularized by a parcimony constraint favouring useful connections. This approach was implemented for a new kind of dynamical model of the interactions, called *inertial model*, which uses both gene expressions and their derivatives. The first results on the *S.O.S. DNA Repair* network of *E.coli* were very promising by showing the power of our approach and of the considered model.

Many kinds of 'motifs' occur in gene networks and the results we obtained on the *star-like* topology of the *S.O.S.* network should be completed by the inference of other networks for other organisms.

A useful work will be to draw theoretical results about the sample complexity for our model family: such a study should make use of machine learning theory to get bounds on size of data sets necessary to provide good prediction ability.

Although it seems that well penalized likelihood functions do not have many local maxima, since the variances of many parameters computed after *multiple random starting points* experiments are quite low, a further theoretical study would be necessary to handle clusters of identified networks, in order to let the biologist choose between a few solutions.

A more crucial point is the model choice. The goal of this article was not to present a new model of regulation, but to propose a new network identification method. Nevertheless, one can object that the present model is based on several strong assumptions, such as stationarity or additive regulation. The model is obviously to be improved in order to represent more realistic phenomena, such as non-linear and combinatorial regulations.

Further work has also to be done concerning the introduction of additional missing variables in the models. The *S.O.S.* data set was not sufficiently large to enlighten definitely the power of this functionality, although the similarity concerning profile and regulations between the missing variable and the protein *LexA* is encouraging. Larger data sets with important missing genes could be used. The method should be able to infer their regulations and their expression behaviours. The optimal number of missing variables to add in the model is also an interesting parameter, which could be learned by the EM algorithm in a Bayesian way.

Presently, our approach is not yet capable of being used for large networks. A first extension to solve this problem consists in using full Bayesian inference framework, which could allow us to incorporate more prior knowledge. A second extension is to develop a 'Divide and Conquer' strategy by considering some subnetworks as intermediary nodes.

ACKNOWLEDGEMENTS

We thank Uri Alon (Weizmann Institute) for having suggested us to use the *S.O.S. DNA Repair* data and providing them. We are grateful to Marie Dutreix (Institut Curie) for having introduced us to the functioning of this network. This work was partially supported by Université Pierre et Marie Curie with a Bonus Qualité Recherche grant.

REFERENCES

Alché-Buc,F.(d'), Lahaye,P.-J., Vujasinovic,T., Bottani,S. and Mazurie,A. (2003) *A recurrent artificial neural network based on inertia principle for modeling gene regulatory network*, Accepted chapter for the book 'Bioinformatics using Computational Intelligence Paradigms', Seiffert,U. (ed.), World Scientific Publishing, to appear in 2004

Blimes,J. (1998) *A gentle tutorial of the EM algorithm and its application to parameter estimation for gaussian mixture and hidden markov models*. Berkeley,U.C. (ed.), International Computer Sci-

ence Institue (ICSI) and Computer Science Division, Department of Electrical Engineering and Computer Science.

Dempster,A.P., Laird,N.M. and Rubin,D.B. (1977) Maximum likelihood from incomplete data via the EM algorithm. *JRSSB*, **39**, 1–38.

D'haeseleer,P. (2000) *Reconstructing Gene Networks from Large Scale Gene Expression Data*. University of New Mexico.

Friedman,N., Linial,M., Nachman,I. and Pe'er,D. (2000) Using Bayesian networks to analyze expression data. *RECOMB*, 127–135.

Ghahramani,Z. and Hinton,G.E. (2000) Variational learning for switching state-space models. *Neural Computation*, **12**, 831–864.

Girosi,F., Jones,M. and Poggio,T. (1995) Regularization theory and neural networks architectures. *Neural Computation*, **7**, 219–269.

Guelzim,N., Bottani,S., Bourgine,P. and Képès,F. (2002) Topological and causal structure of the yeast transcriptional regulatory network. *Nat. Genet.*, **31**, 60–63.

Hartemink,A.J., Gifford,D.K., Jaakkola,T.S. and Young,R.A. (2001) Using graphical models and genomic expression data to statistically validate models of genetic regulatory networks. *Proceedings of Pacific Symposium on Biocomputing*. pp. 422–433.

Heckerman,D. (1995) A tutorial on learning with bayesian networks. *Microsoft Research. Technical Report MSR-TR-95-06*. Redmond, Washington.

Kim,S., Imoto,S. and Miyano,S. (2003) Dynamic Bayesian network and nonparametric regression for nonlinear modeling of gene networks from time series gene expression data. pp. 104–113.

McAdams,H.H. and Arkin,A. (1997) Stochastic mechanisms in gene expression. **94**, 814–819.

Mjolsness,E., Mann,T., Castano,R. and Wold,B. (2000) From coexpression to coregulation: an approach to inferring transcriptional regulation among gene classes from large-scale expression data. *Advances in Neural Information Processing Systems*, **12**, 928–934.

Murphy,K. and Mian,S. (1999) *Modelling gene expression data using dynamic Bayesian networks*. University of California, Berkeley.

Ong,I.M., Glasner,J.D. and Page,D. (2002) Modelling regulatory pathways in *E.coli* from time series expression profiles. *Bioinformatics*, **18**, S241–S248.

Ormoneit,D. and Tresp,V. (1998) Averaging, maximum penalized likelihood and bayesian estimation for improving gaussian mixture probability density estimates. *IEEE Trans. Neural Networks*, **9**, 639–650.

Pe'er,D., Regev,A., Elidan,G. and Friedman,N. (2001) Inferring subnetwoks from perturbed expression profiles. *Bioinformatics*, **1**, 1–9.

Ronen,M., Rosenberg,R., Shraiman,B.I. and Alon,U. (2002) Assigning numbers to the arrows: parameterizing a gene regulation network by using accurate expression kinetics. *Proc. Natl Acad. Sci. USA*, **99**, 10555–10560.

Rosti,A.-V.I. and Gales,M.J.F. (2001) *Generalised Linear Gaussian Models*. Cambridge University Engineering Department.

Sassanfar,M. and Roberts,J. (1990) Nature of the SOS-inducing signal in *Escherichia coli*. The involvement of DNA replication. *J. Mol. Biol.*, **212**, 79–96.

Shen-Orr,S.S., Milo,R., Mangan,S. and Alon,U. (2002) Network motifs in the transcriptional regulation network of *Escherichia coli*. *Nat. Genet.*, **31**, 64–68.

Weaver,D.C., Workman,C.T. and Stormo,G.D. (1999) Modeling regulatory networks with weight matrices. *Pac. Symp. Biocomput.*, **4**, 112–123.