BIOINFORMATICS ORIGINAL PAPER

Vol. 21 no. 3 2005, pages 1–8
doi:10.1093/bioinformatics/bti014

A Bayesian approach to reconstructing genetic regulatory networks with hidden factors

Matthew J. Beal[1], Francesco Falciani[2], Zoubin Ghahramani[3], Claudia Rangel[4,†] and David L. Wild[4,]*

[1]Department of Computer Science & Engineering, State University of New York at Buffalo, 201 Bell Hall, Buffalo, NY 14260-2000, USA, [2]School of Biosciences, University of Birmingham, Edgbaston, Birmingham B15 2TT, UK, [3]Gatsby Computational Neuroscience Unit, University College London, 17 Queen Square, London WC1N 3AR, UK and [4]Keck Graduate Institute of Applied Life Sciences, 535 Watson Drive, Claremont, CA 91171, USA

Received on December 1, 2003; revised on August 30, 2004; accepted on August 31, 2004
Advance Access publication September 7, 2004

ABSTRACT

Motivation: We have used state-space models (SSMs) to reverse engineer transcriptional networks from highly replicated gene expression profiling time series data obtained from a well-established model of T cell activation. SSMs are a class of dynamic Bayesian networks in which the observed measurements depend on some hidden state variables that evolve according to Markovian dynamics. These hidden variables can capture effects that cannot be directly measured in a gene expression profiling experiment, for example: genes that have not been included in the microarray, levels of regulatory proteins, the effects of mRNA and protein degradation, etc.

Results: We have approached the problem of inferring the model structure of these state-space models using both classical and Bayesian methods. In our previous work, a bootstrap procedure was used to derive classical confidence intervals for parameters representing 'gene–gene' interactions over time. In this article, variational approximations are used to perform the analogous model selection task in the Bayesian context. Certain interactions are present in both the classical and the Bayesian analyses of these regulatory networks. The resulting models place JunB and JunD at the centre of the mechanisms that control apoptosis and proliferation. These mechanisms are key for clonal expansion and for controlling the long term behavior (e.g. programmed cell death) of these cells.

Availability: Supplementary data is available at http://public.kgi.edu/~wild/index.htm and Matlab source code for variational Bayesian learning of SSMs is available at http://www.csc.ubuffalo.edu/faculty/mbeal/software.html

Contact: David_Wild@kgi.edu

*To whom correspondence should be addressed.

† Current address: Molecular and Computational Biology, University of Southern California, 1042 West 36th Place, DRB289, Los Angeles, CA 90089, USA

INTRODUCTION

The application of high-density DNA microarray technology to gene transcription analysis has stimulated the development of algorithms to classify and describe the complex transcriptional response of a biological system and to reveal interactions between the components of a cellular system. Many of the tools that have been applied in an exploratory manner to the problem of reverse engineering genetic regulatory networks from gene expression data have been recently reviewed by van Someren *et al*. (2002). Murphy and Mian (1999) were the first to propose the use of a general class of graphical models known as Dynamic Bayesian Networks (DBNs) to model time series gene expression data. Bayesian networks have a number of features that make them attractive candidates for modeling gene expression data, such as their ability to handle noisy or missing data, to handle hidden variables such as protein levels that may have an effect on mRNA expression levels, to describe locally interacting processes and the possibility of making causal inferences from the derived models. However, much published work to date has assumed that all the possibly interacting variables are observed on the microarray. This precludes the existence of hidden causes or unmeasured genes whose involvement might dramatically simplify the network structure and therefore ease interpretability of the mechanisms in the underlying biological process. Although microarray technologies have made it possible to measure time series of the expression level of many genes simultaneously, we cannot hope to measure all possible factors contributing to genetic regulatory interactions, and the ability of Bayesian networks to handle such hidden variables would appear to be one of their main advantages as a modeling tool.

We have previously applied linear state-space modeling to reverse engineer transcriptional networks from highly replicated expression profiling data obtained from

a well-established model of T cell activation in which we have monitored a set of relevant genes across a time series (Rangel *et al*., 2001, 2004a). Linear Gaussian state-space models (SSMs), also known as Linear Dynamical Systems (Roweis and Ghahramani, 1999) or Kalman filter models (Brown and Hwang, 1997) are a subclass of dynamic Bayesian networks used for modeling time series data and have been used extensively in many areas of control and signal processing. SSMs have a number of features that make them attractive for modeling gene expression time series data. They assume the existence of a set of hidden state variables from which we can make noisy continuous measurements, and that evolve with Markovian dynamics. In our application, the noisy measurements are the observed gene expression levels at each time point, and we assume that the hidden variables are modeling effects that cannot be measured in a gene expression profiling experiment, for example: the effects of genes that have not been included on the microarray, levels of regulatory proteins, the effects of mRNA and protein degradation, etc.

The task of deciding upon a suitable dimension for the hidden state space remains a difficult problem. In our previous work (Rangel *et al*., 2001, 2004a), this was determined by a cross-validation experiment in which we increment the number of hidden states and monitor the predictive likelihood using a portion of the data set that has not been used to train the model. However, the hold-out set error in a cross-validation experiment is a noisy quantity and, for a reliable measure, a very large hold-out data set is needed; ideally, we would prefer to utilize all the data for model learning rather than holding out a large portion of it.

More recently, Perrin *et al*. (2003) and Wu *et al*. (2004) have also described related SSMs for genetic regulatory networks. Perrin *et al*. (2003) build models with a small number of hidden states (0, 1 and 2), whilst Wu *et al*. (2004) use factor analysis and a Bayesian Information Criterion (BIC) penalized maximum likelihood approach to determine the hidden state space dimensionality. These approaches suffer from several drawbacks, not least in that they cannot provide us with posterior distributions over all the parameters of the model that are needed to quantify our uncertainty. We discuss the advantages of our approach over these methods in the section Discussion.

A variational Bayesian (VB) treatment of these models provides a novel way to learn their structure, that is to identify the optimal dimensionality of their state space. The VB algorithm provides distributions over the model parameters and, as has been shown in a series of experiments on synthetic data, can be used successfully to determine the structure of the true generating model, including inferring the dimensionality of the hidden state space (Beal, 2003).

The classical approach used in our previous work tests each 'gene–gene' interaction by doing a hypothesis test comparing the bootstrap confidence interval of each parameter to the null hypothesis that it is zero. The analogous Bayesian procedure examines the *posterior distribution* of each parameter to determine whether that parameter can be inferred to have a significantly large positive or negative value. In this article, we demonstrate how variational Bayesian methods can be used to approximate the posterior quantities required for Bayesian learning in SSMs of gene expression time series. Moreover, we show that this powerful approach provides a practical framework for elucidating underlying interactions amongst genes in DNA microarray time-series data.

SYSTEM AND METHODS

The Linear dynamical system model

Variables and topology In SSMs, a sequence $(\mathbf{y_1}, \ldots, \mathbf{y}_T)$ of p-dimensional real-valued observation vectors, denoted $\mathbf{y}_{1:T}$, is modeled by assuming that at each time step t, $\mathbf{y_t}$ was generated from a k-dimensional real-valued hidden state variable $\mathbf{x}_t$, and that the sequence of $\mathbf{x}$s follow a first-order Markov process. The joint probability of a sequence of states and observations is therefore given by:

$$p(\mathbf{x}_{1:T}, \mathbf{y}_{1:T}) = p(\mathbf{x}_1)p(\mathbf{y}_1 \mid \mathbf{x}_1)\prod_{t=2}^{T} p(\mathbf{x_t} \mid \mathbf{x}_{t-1})p(\mathbf{y_t} \mid \mathbf{x_t}). \quad (1)$$

The distribution $p(\mathbf{x}_1)$ over the first hidden state is assumed Gaussian.[1] We focus on models where both the dynamics, $p(\mathbf{x_t} \mid \mathbf{x}_{t-1})$, and output functions, $p(\mathbf{y_t} \mid \mathbf{x_t})$, are linear and time-invariant and the distributions of the state evolution and observation noise variables are Gaussian, i.e. linear-Gaussian SSMs:

$$\mathbf{x}_t = A\mathbf{x}_{t-1} + \mathbf{w}_t, \quad \mathbf{w}_t \sim \mathrm{N}(\mathbf{0}, Q) \quad (2)$$

$$\mathbf{y}_t = C\mathbf{x}_t + \mathbf{v}_t, \quad \mathbf{v}_t \sim \mathrm{N}(\mathbf{0}, R) \quad (3)$$

where A is the $(k \times k)$ state dynamics matrix, C is the $(p \times k)$ observation matrix, and Q $(k \times k)$ and R $(p \times p)$ are the covariance matrices for the state and output noise variables $\mathbf{w_t}$ and $\mathbf{v_t}$. The parameters A and C are analogous to the transition and emission matrices, respectively, in a hidden Markov model (Roweis and Ghahramani, 1999).

A straightforward extension of this model is to allow the dynamics and observation models to include a dependence on a series of d-dimensional driving inputs $\mathbf{u}_{1:T}$:

$$\mathbf{x_t} = A\mathbf{x}_{t-1} + B\mathbf{u_t} + \mathbf{w_t}, \quad (4)$$

$$\mathbf{y_t} = C\mathbf{x_t} + D\mathbf{u_t} + \mathbf{v_t}. \quad (5)$$

[1] For computational reasons in our implementation of the SSM, this prior over the first hidden state $\mathbf{x_1}$ is in fact realized via a Gaussian distributed auxiliary previous hidden state $\mathbf{x_0}$, which is integrated out—the details can be found in Beal (2003), specifically Equation (5.14).

Here, B ($k \times d$) and D ($p \times d$) are the input-to-state and input-to-observation matrices, respectively. If we now augment the driving inputs with a constant bias, then this input driven model is able to incorporate an arbitrary origin displacement for the hidden state dynamics, and also can induce a displacement in the observation space. These displacements can be learnt as parameters of the input-to-state (B) and input-to-observation (C) matrices.

An input-dependent SSM can be used to model control systems. Another possible way in which the inputs can be utilized is to feedback the outputs (data) from previous time steps in the sequence into the inputs for the current time step. This means that the hidden state can concentrate on modeling hidden factors whilst the Markovian dependencies between successive *outputs* are modeled using the output–input feedback construction. We adapt this model for the analysis of gene expression time series data below.

Without loss of generality, we can set the hidden state evolution noise covariance, Q, to the identity matrix. This is possible since an arbitrary noise covariance can be incorporated into the state dynamics matrix A, and the hidden state rescaled and rotated to be made commensurate with this change.

The remaining parameter of a linear-Gaussian SSM is the covariance matrix, R, of the Gaussian output noise, $\mathbf{v}_t$. We assume R to be diagonal and we learn the scale of these diagonal terms. For notational convenience, we collect the above parameters into a single parameter vector for the model: $\boldsymbol{\theta} = (A, B, C, D, R)$.

The parameters of a SSM can be learned using maximum likelihood (ML) methods (Shumway and Stoffer, 1982). However, ML methods are prone to overfitting especially when fitting models with many variables from relatively small amounts of data. We now turn to considering a Bayesian analysis of the SSM, which avoids overfitting and provides error bars on model parameters. For this, we need to define priors over all parameters.

Priors In general, the priors on all model parameters are chosen from the family of conjugate distributions, which makes it possible to obtain efficient analytical updates in the variational Bayesian algorithm (Beal and Ghahramani, 2003). These priors are in turn parameterized by *hyperparameters* that can be optimized so as to adapt to the scale of the data and so as to select relevant and irrelevant variables in the model. The general idea of optimizing hyperparameters to discover which interactions (e.g. between genes and hidden factors) are relevant is known as *Automatic Relevance Determinination* (ARD).

The model interaction parameters A, B, C and D are given zero-mean Gaussian priors with precisions (inverse variances) specified by hyperparameters $\boldsymbol{\alpha}$, $\boldsymbol{\beta}$, $\boldsymbol{\gamma}$ and $\boldsymbol{\delta}$, respectively.

Keeping in mind that the parameters are collected into $\boldsymbol{\theta} = (A, B, C, D, R)$, the marginal likelihood, which integrates over the parameters and hidden states, can then be written $p(\mathbf{y}_{1:T}) = \int p(\boldsymbol{\theta}, \mathbf{x}_{0:T}, \mathbf{y}_{1:T}) d\boldsymbol{\theta}\, d\mathbf{x}_{0:T}$.

All hyperparameters can be optimized during learning. Preliminary experiments on artificial data (Ghahramani and Beal, 2001; Beal, 2003) have shown that the VB approach successfully determines the structure of SSMs.

An SSM for gene expression time series In this article, we use the input-dependent SSM, and we *feed back* the gene expressions from the previous time step into the input for the current time step. In doing this, we attempt to discover gene–gene interactions across time steps, with the hidden state in this model now really representing unobserved variables. An advantage of this architecture is that we can now use the ARD mechanisms to determine which genes are influential across adjacent time slices.

A graphical model for this setup is given in Figure 1. When the input is replaced with the previous time step's observed data, the equations for the SSM can be rewritten from Equations (4) and (5) into the form:

$$\mathbf{x}_t = A\mathbf{x}_{t-1} + B\mathbf{y}_{t-1} + \mathbf{w}_t \tag{6}$$

$$\mathbf{y}_t = C\mathbf{x}_t + D\mathbf{y}_{t-1} + \mathbf{v}_t\ . \tag{7}$$

Here, $\mathbf{y}_t$ denotes the gene expression levels at time step t and $\mathbf{x}_t$ the unobserved hidden factors. In practice, $\mathbf{y}$ is in fact the suitably normalized and transformed values of the gene expression levels. The matrix D in the observation equation captures gene–gene expression level influences at consecutive time points whilst the matrix C captures the influence of the hidden variables on gene expression level at each time point. Matrix B models the influence of gene expression values from previous time points on the hidden states and A is the state dynamics matrix. As a function only of the data at the previous time step $\mathbf{y}_{t-1}$, the data at time t can be written

$$\mathbf{y}_t = (CB + D)\mathbf{y}_{t-1} + \mathbf{r}_t, \tag{8}$$

where $\mathbf{r}_t = \mathbf{v}_t + C\mathbf{w}_t + CA\mathbf{x}_{t-1}$ includes all contributions from noise and previous states. Thus, to first order the interaction between gene j and gene i can be characterized by the

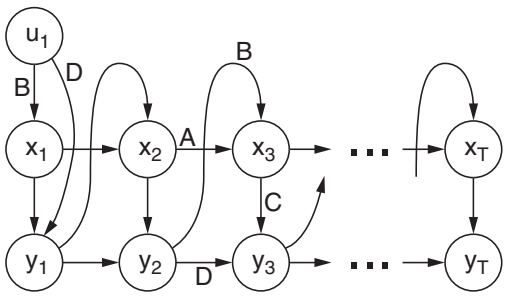

Fig. 1. The feedback graphical model with outputs feeding into inputs. Gene expression levels at time t are represented by y_t, whilst the hidden factors are represented by x_t.

element $[CB+D]_{ij}$ of the matrix. Indeed, this matrix need not be symmetric and the element represents activation or inhibition from gene j to gene i at the next time step depending on its sign. This is the matrix we will concentrate our analysis on, since it captures all of the information related to gene–gene interaction over one time step. We have also shown that, if the gene expression model is stable, controllable and observable, then the $CB + D$ matrix remains invariant to any coordinate transformations of the state and is, therefore, *identifiable* (Rangel *et al*., 2004b). The identifiability property is important, for without it, it would be possible for different values of the SSM parameters (and hence, different values of $CB + D$) to give rise to identically distributed observables, making the statistical problem of estimation ill-posed.

ALGORITHM

Variational Bayesian learning

The classical approach tests each gene–gene interaction by doing a hypothesis test comparing the bootstrap confidence interval of each parameter to the null hypothesis that it is zero. The analogous Bayesian procedure examines the posterior distribution of each parameter to determine whether that parameter can be inferred to have a large positive or negative value. We now describe how variational methods can be used to approximate the posterior quantities required for Bayesian learning in SSMs.

Let $\mathbf{y}$ denote the observed variables, $\mathbf{x}$ the latent variables and $\boldsymbol{\theta}$ the parameters. The marginal likelihood for a model m, $p(\mathbf{y} \mid m) = \int p(\mathbf{y}, \mathbf{x}, \boldsymbol{\theta} \mid m)\ d\boldsymbol{\theta}\, d\mathbf{x}$ tells us how well a particular model structure fits a data set, averaging over all possible latent variable and parameter values. As opposed to the *maximum* likelihood, which always prefers more complex models, the *marginal* likelihood embodies the concept of an automatic Occam's razor, penalizing models with too many free parameters (MacKay, 1992; Kass and Raftery, 1995). This forms the basis of Bayesian model comparison. Unfortunately, the marginal likelihood is computationally intractable to compute for most models of interest, including SSMs (Früwirth-Schnatter, 1995). However, we can lower bound the marginal likelihood by introducing any distribution, over both latent variables and parameters, that has support where $p(\mathbf{x}, \boldsymbol{\theta} \mid \mathbf{y}, m)$ does, and then appealing to Jensen's inequality (due to the concavity of the log function):

$$\begin{aligned} \ln p(\mathbf{y} \mid m) &= \ln \int p(\mathbf{y}, \mathbf{x}, \boldsymbol{\theta} \mid m)\ d\mathbf{x}\, d\boldsymbol{\theta} \\ &= \ln \int q(\mathbf{x}, \boldsymbol{\theta}) \frac{p(\mathbf{y}, \mathbf{x}, \boldsymbol{\theta} \mid m)}{q(\mathbf{x}, \boldsymbol{\theta})}\ d\mathbf{x}\, d\boldsymbol{\theta} \\ &\geq \int q(\mathbf{x}, \boldsymbol{\theta}) \ln \frac{p(\mathbf{y}, \mathbf{x}, \boldsymbol{\theta} \mid m)}{q(\mathbf{x}, \boldsymbol{\theta})}\ d\mathbf{x}\, d\boldsymbol{\theta}. \end{aligned} \tag{9}$$

Maximizing this lower bound with respect to the free distribution $q(\mathbf{x}, \boldsymbol{\theta})$ results in $q(\mathbf{x}, \boldsymbol{\theta}) = p(\mathbf{x}, \boldsymbol{\theta} \mid \mathbf{y}, m)$, which when substituted above turns the inequality into an equality, but is still computationally intractable. Instead, we use a simpler, factorized approximation $q(\mathbf{x}, \boldsymbol{\theta}) = q_{\mathbf{x}}(\mathbf{x}) q_{\boldsymbol{\theta}}(\boldsymbol{\theta})$:

$$\begin{aligned} \ln p(\mathbf{y} \mid m) &\geq \int q_{\mathbf{x}}(\mathbf{x}) q_{\boldsymbol{\theta}}(\boldsymbol{\theta}) \ln \frac{p(\mathbf{y}, \mathbf{x}, \boldsymbol{\theta} \mid m)}{q_{\mathbf{x}}(\mathbf{x}) q_{\boldsymbol{\theta}}(\boldsymbol{\theta})}\ d\mathbf{x}\, d\boldsymbol{\theta} \\ &= \mathcal{F}_m(q_{\mathbf{x}}(\mathbf{x}), q_{\boldsymbol{\theta}}(\boldsymbol{\theta}), \mathbf{y}). \end{aligned} \tag{10}$$

The quantity $\mathcal{F}$ is a function of the free distributions $q_{\mathbf{x}}(\mathbf{x})$ and $q_{\boldsymbol{\theta}}(\boldsymbol{\theta})$. The VB algorithm iteratively maximizes $\mathcal{F}$ in Equation (10) with respect to the free distributions, $q_{\mathbf{x}}(\mathbf{x})$ and $q_{\boldsymbol{\theta}}(\boldsymbol{\theta})$. We use elementary calculus of variations to take functional derivatives of the lower bound with respect to $\boldsymbol{q}_{\mathbf{x}}(\mathbf{x})$ and $q_{\boldsymbol{\theta}}(\theta)$, each while holding the other fixed. This results in the following update equations where the superscript (ℓ) denotes the iteration number:

VB-E step:

$$q_{\mathbf{x}}^{(\ell+1)}(\mathbf{x}) \propto \exp\left[\int \ln p(\mathbf{x},\mathbf{y} \mid \boldsymbol{\theta}, m)\, q_{\boldsymbol{\theta}}^{(\ell)}(\boldsymbol{\theta})\, d\boldsymbol{\theta} \right] \tag{11}$$

VB-M step:

$$q_{\boldsymbol{\theta}}^{(\ell+1)}(\boldsymbol{\theta}) \propto p(\boldsymbol{\theta} \mid m)\ \exp\left[\int \ln p(\mathbf{x},\mathbf{y} \mid \boldsymbol{\theta}, m)\, q_{\mathbf{x}}^{(\ell+1)}(\mathbf{x}) d\mathbf{x} \right]. \tag{12}$$

Clearly, $q_{\boldsymbol{\theta}}(\boldsymbol{\theta})$ and $q_{\mathbf{x}}(\mathbf{x})$ are coupled, so we iterate these equations until convergence. Readers familiar with the EM algorithm (Dempster *et al*., 1977) may note the similarity between this iterative algorithm and EM. We call this procedure the *Variational Bayesian EM Algorithm*, and we name the two update equations the VB-E and VB-M steps in direct analogy to the E and M steps of EM. In Beal and Ghahramani (2003), this relationship is described in detail and it is also shown that this algorithm minimizes the KL divergence between the approximation $q_{\boldsymbol{\theta}}(\boldsymbol{\theta}) q_{\mathbf{x}}(\mathbf{x})$ and the true posterior $p(\mathbf{x}, \boldsymbol{\theta} \mid \mathbf{y}, m)$.

If the prior is conjugate and the joint model (including latent variables) is in the exponential family, the maximization of the VB lower bound on the marginal likelihood results in a simple and analytically tractable generalization of the EM algorithm (Beal and Ghahramani, 2003). The VB–EM algorithm reduces to the ordinary EM algorithm if we restrict the parameter density to a point estimate (i.e. Dirac delta function), $q_{\boldsymbol{\theta}}(\boldsymbol{\theta}) = \delta(\boldsymbol{\theta} - \boldsymbol{\theta}^*)$. The VB-E step has about the same time complexity as the E step, and is, in all ways, identical (except that it is rewritten in terms of the expected natural parameters of the exponential family). The key difference is that the VB-M step computes a Bayesian *distribution* over parameters $q_{\boldsymbol{\theta}}^{(\ell)}(\boldsymbol{\theta})$ rather than a point estimate $\boldsymbol{\theta}^{(\ell)}$. This difference is what makes it possible to capture an automatic Occam's razor and select beween alternative genetic regulatory networks.

Applying the arguments in Equation (10) to this case, we lower bound the log marginal likelihood for the SSM:

$$\begin{aligned}\ln\, & p(\mathbf{y}_{1:T}|\mathbf{u}_{1:T}) \\ & \geq \int d\boldsymbol{\theta}\, d\mathbf{x}_{0:T} q_{\boldsymbol{\theta}}(\boldsymbol{\theta}) q_{\mathbf{x}}(\mathbf{x}_{0:T}) \ln \frac{p(\boldsymbol{\theta}, \mathbf{x}_{0:T}, \mathbf{y}_{1:T}|\mathbf{u}_{1:T})}{q_{\boldsymbol{\theta}}(\boldsymbol{\theta}) q_{\mathbf{x}}(\mathbf{x}_{0:T})} \\ & = \mathcal{F}(q_{\mathbf{x}}(\mathbf{x}_{0:T}), q_{\boldsymbol{\theta}}(\boldsymbol{\theta}), \mathbf{y}_{1:T}) \,. \end{aligned} \tag{13}$$

and use the variational Bayesian EM (VB–EM) algorithm to optimize the lower bound.

The hyperparameters, $\boldsymbol{\alpha}$, $\boldsymbol{\beta}$, $\boldsymbol{\gamma}$, $\boldsymbol{\delta}$, a and b, and the prior parameters, Σ_0 and $\boldsymbol{\mu}_0$, can be updated so as to maximize the lower bound on the marginal likelihood [Equation (13)]. By taking derivatives of $\mathcal{F}$ with respect to the hyperparameters, updates can be derived, which are detailed in Beal (2003). The complete learning algorithm for SSMs consists of repeated iterations of the VB-E step, VB-M step, calculation of $\mathcal{F}$ and hyperparameter updates. In practice, one does not need to compute $\mathcal{F}$ at all for learning. It may also be inefficient to update the hyperparameters after every iteration of VB–EM, and for some applications in which the user is certain of their prior specifications, a hyperparameter learning scheme may not be required at all.

IMPLEMENTATION

Experimental data

The data used in this article are the results of two experiments with interarray replications that we have performed to characterize the response of a human T cell line (Jurkat) to PMA and ionomicin treatment. Details of data collection and preprocessing are described elsewhere (Rangel *et al*., 2004a). In the first experiment, we monitored the expression of 88 genes using cDNA array technology across 10 time points. In the second, an identical experimental protocol was used but additional genes were added to the arrays. Genes that displayed very poor reproducibility between the two experiments were removed, leaving 58 genes. For the purpose of training the SSM, log transformed expression profiles for the same gene in the two experiments were scaled together using a variant of the method of Bolstad *et al*. (2002) adapted to our data with the assumption that all 44 replicates of each time series have a similar underlying distribution.

Results from the variational Bayesian model

We examined the gene–gene influences represented by elements of the matrix $[CB + D]$. The variational Bayesian model provides us with posterior distributions (albeit approximate) for the parameters C, B and D, which are given by the update Equations (11 and 12). Using the posterior distributions for these parameters, we compute the distribution of each of the elements in the combined matrix $[CB + D]$. We consider an element of this matrix as providing evidence for a candidate gene–gene interaction if the element's posterior distribution is positioned significantly far from the zero point (of no influence). Significance in this scenario corresponds to the zero point being more than n standard deviations from the posterior mean for that entry. Since these distributions are Gaussian (Beal, 2003), and may lie above or below the zero point (corresponding to activation or inhibition), we can use the standard Z-statistic for normally distributed variables: the threshold values of n are then given by 2.7478, 2.8782 and 3.0902 for 99.4, 99.6 and 99.8% confidences, respectively.

A number of SSMs were trained using the VB–EM algorithm starting from 10 different random initializations and with $k = 1, \ldots, 20$ hidden state dimensions. Figure 2 shows the variation of the median value of $\mathcal{F}$ with hidden state dimension k (also plotted are the individual $\mathcal{F}$ values from each of the 10 random seeds). The median $\mathcal{F}$ value peaks around $k = 14$. Figure 3 shows the number of significant gene–gene interactions that are consistently repeated in all 10 runs at each value of k; there are three plots, corresponding to three significance levels. Regardless of the significance level we choose, we can see that the number of significant interactions has leveled off by around $k = 14$, which corresponds to the peak in $\mathcal{F}$ graph. Importantly, some interactions appear robustly even in models that incorporate many hidden variables. Note that models with no hidden variables, $k = 0$ [equivalent to the linear models of D'Haeseleer *et al*. (1999) and Holter *et al*. (2001)], give a much higher estimate of the number of direct interactions, which may result in a very misleading impression of the underlying genetic regulatory networks. The resulting networks have been visualized using the software application, Cytoscape (Ideker *et al*., 2002), and a figure representing the robust, highly significant interactions may be found on the supplementary website http://public.kgi.edu/~wild/VBLDS/index.htm.

DISCUSSION

In order to compare the models derived from a classical bootstrap approach (Rangel *et al*., 2001, 2004a) and the VB methods described in this article, we have identified the gene–gene interactions shared between them. Biologically interesting interactions identified in Rangel *et al*. (2004a) are represented in a large number of VB models. For example, the interaction between Fyb and IL-2 target genes is supported in at least 60% of the VB models starting from different random seeds with $k = 9$ hidden states [the number of hidden states used by Rangel *et al*. (2004a)] at a significance level of 99.8%.

A careful analysis of the VB SSMs with $k = 14$ hidden states has also revealed interesting biological properties. The most interesting feature is a sub-network representing the interaction between Jun-D, and Jun-B, with a number of genes involved in programmed cell death (Fig. 4). The model is consistent with a recently proposed hypothesis in Weitzman (2001). Mammals have three members of the Jun protein family. These proteins (Jun-B, c-Jun and Jun-D) form dimers

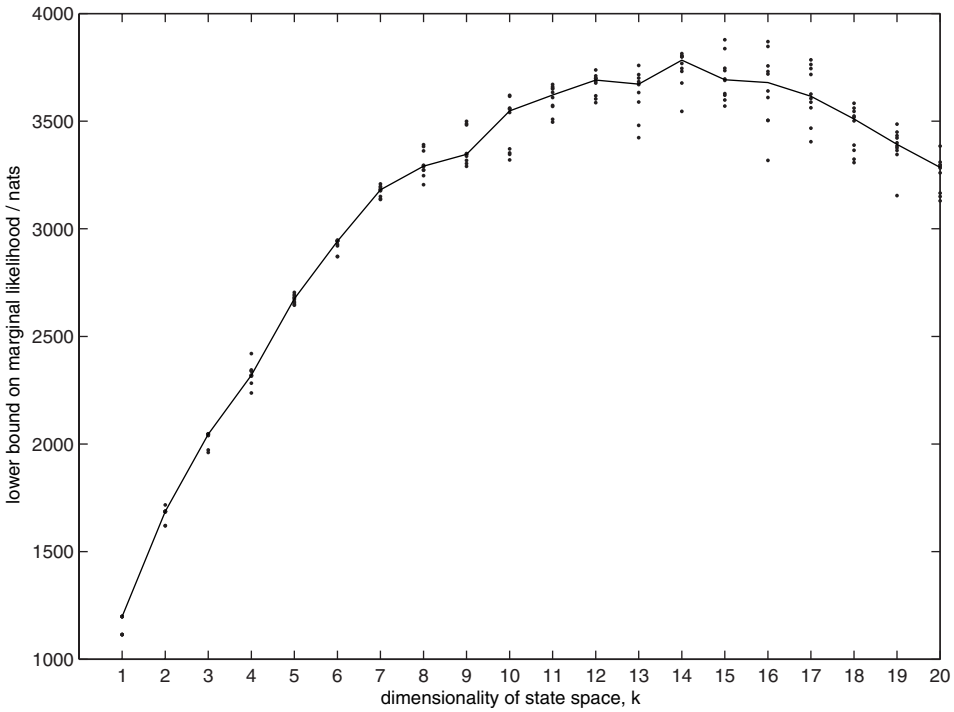

Fig. 2. Variation of $\mathcal{F}$ with hidden state dimension k for 10 random initializations of VBEM. The line represents the median $\mathcal{F}$ value.

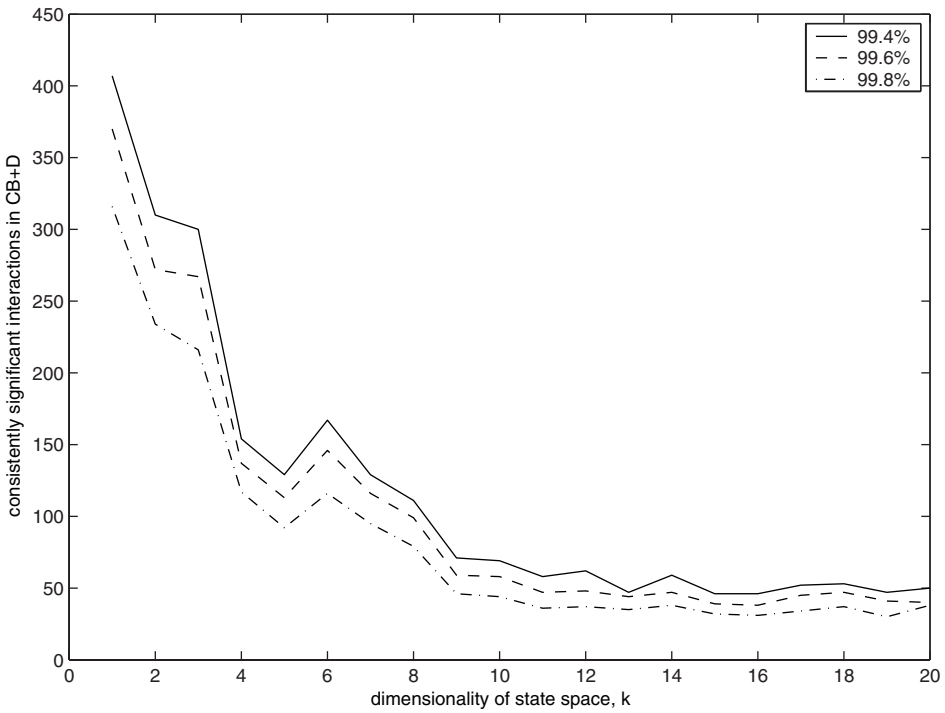

Fig. 3. The number of significant interactions that are repeated in all 10 runs of VB-EM at each value of k. There are 3 plots, each corresponding to a different significance level.

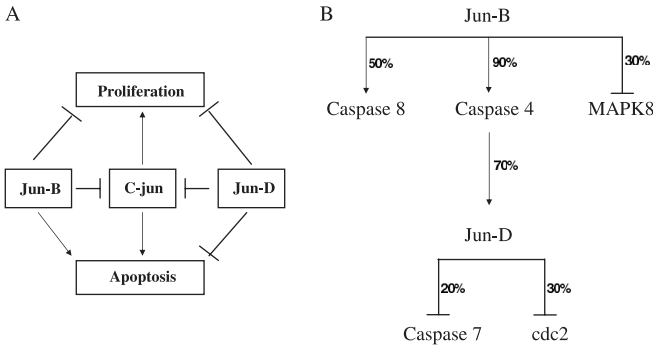

Fig. 4. (**A**) The current hypothesis on the role of Jun proteins in controlling apoptosis and proliferation. Jun-B is pro-apoptotic and inhibits proliferation whereas Jun-D is both an anti-proliferative and anti-apoptotic signal. (**B**) The the connections between Jun proteins and apoptotic genes as revealed by the VB SSM. *Jun-B* (gene 54) is clearly modeled as a pro-apoptotic gene [activating *Caspase 8* (gene 18) and *Caspase 4* (gene 38) and repressing the survival factor *MAPK8* (gene 53)]. On the contrary, *Jun-D* (gene 11) is repressing *Caspase 7* (gene 52) and the cell cycle regulator *cdc2* (gene 31). Numbers on the edges represent the number of models from 10 different randon seeds in which the interaction is supported at a confidence level of 99.8%

with each other or with members of related Fos and ATF families to constitute the AP-1 transcription factor (Angel and Karin, 1991). Experiments in which Jun genes were artificially over-expressed in fibroblasts suggested that Jun members might play distinct roles in regulating cell proliferation, transformation and apoptosis (Weitzman, 2000). Jun-D appears to protect from apoptosis (Weitzman, 2000), whereas Jun-B negatively regulates cell growth through activation of p16INK4a inhibitor and repression of cyclin D1 expression and is thought to inhibit programmed cell death (Bakiri, 2000; Passegu and Wagner, 2000). Our model reproduces this scenario (Fig. 4) and predicts that Jun-B would act as transcriptional activator of the apoptotic genes *Caspase 4* (supported in 90% of the VB models trained with different random seeds) and *Caspase 8* (supported in 50% of the models) and as transcriptional repressor of the survival factor MAPK8 (Hess *et al*., 2000; supported in 30% of the models). The *Jun-D* gene is predicted to repress the expression of another apoptosis related gene (*Caspase 7*) (in 20% of the models) and the cell cycle regulator gene *cdc2* (in 30% of the models).

Ultimately, the validation of different computational approaches to reverse engineering must be experimental. Knock out and over-expression experiments need to be performed to verify that the inferred interactions are really representative of the biological complexity we intend to model. Such experiments are often non-trivial and difficult to do in a systematic way. The models we have generated show that even a perfectly reasonable and well-supported biological hypothesis needs testing. Figure 4 shows the sub-network connecting Jun-B and Jun-D to programmed cell death. This network is well-supported, but the current literature does not directly show that increased levels of *Caspase 4* trigger the expression of *Jun-D* and consequent inhibition of *Caspase 7*. The experimental verification of these models is one focus of our current research, as is the application of our methodology in a well-defined system of biological interest that can be easily manipulated experimentally and for which information about its genome and investigative tools are available. The infection of the human intestine epithelial cells line Caco-2 by the *Escherichia coli* enterohaemorragic strain EHEC O157 is an excellent example of such system. Although a large amount of information is available about regulatory networks in organisms such as *E.coli* (Rosenfeld and Alon, 2003) and *Saccharomyces cerevisiae* (Lee *et al*., 2002), we are unaware of publicly available microarray datasets that contain the necessary degree of replication, which the simulations we have performed with synthetic data indicate are required for accurate network reconstruction (Rangel *et al*., 2004b).

As in our previous work (Rangel *et al*., 2004a), we do not find a one-to-one correspondence between the hidden variables and known biological effects or unmeasured regulatory genes. Two models can have equivalent gene–gene interactions but different implementations of those in terms of hidden variables. The hidden variables were, however, important in practice since they played a large role in mediating the gene–gene interactions over time (Fig. 3). In our model, the hidden variables are likely to represent a combination of complex molecular events (such as a combination of genes and possibly entire pathways) linking two genes. In this scenario, allowing hidden factors is an essential part of our overall goal of developing biologically realistic models.

The variational Bayesian methodology we have described has several distinct advantages over other methods that have recently appeared in the literature. For example, Perrin *et al*. (2003) use a regularized form of ML learning, with a regularization parameter that they suggest is best determined using cross-validation techniques. In addition, they use different initializations of the learning algorithm to obtain empirical estimates of the uncertainty over the parameters. Wu *et al*. (2004) use an EM algorithm for factor analysis to find the hidden states and then fit the state dynamics matrix by solving a separate least squares problem, rather than using an EM algorithm for directly fitting ML parameters of SSMs. This method will not find the ML parameters of the models. Not only does the VB methodology described in this article provide an algorithm that updates *all* parameters of the model during learning on all of the available data, but it also provides analytical forms for the posterior distribution of every parameter without recourse to multiple restarts; indeed Perrin *et al*.'s limitation to just $k = 2$ hidden dimensions may be due to the computational infeasibility of running numerous SSMs to cover the cloud of uncertainty in higher than two dimensions. Wu *et al*. (2004) use a BIC (Bayesian Information Criterion) penalized ML approach to determine

the hidden state space dimensionality, k; unfortunately, the BIC is only valid for very large data sets, tends to over-penalize more complex models and does not provide a mechanism to represent posterior distributions over model parameters and, therefore, is unable to elucidate the sort of gene–gene interactions we are interested in examining. Furthermore, neither the model described by Perrin *et al*. (2003) nor Wu *et al*. (2004) utilize inputs to feed back the outputs from the previous time step. Consequently, these models do not allow genes to affect the hidden states (the B matrix in our model) and do not allow genes to affect other genes directly (the D matrix in our model).

This article has dealt solely with the case of linear dynamics and linear output processes with Gaussian noise. Whilst this is a good first approximation, we expect that a more realistic model of genetic regulatory interactions would have non-linear interactions, reflecting, for example saturation effects in the expression levels of a gene or multiplicative effects where the simultaneous expression of two genes is required to effect a third. The development of such models is another focus of our current research.

REFERENCES

Angel,P. and Karin,M. (1991) The role of Jun, Fos and the AP-1 complex in cell proliferation and transformation. *Biochim. Biophys. Acta.*, **1072**, 129–157.

Bakiri,L. (2000) Cell cycle-dependent variations in c-Jun and Jun-b phosphorylation: a role in the control of cyclin D1 expression. *EMBO J.*, **19**, 2056–2068.

Beal,M.J. and Ghahramani,Z. (2003) The variational Bayesian EM algorithm for incomplete data: with application to scoring graphical model structures. In Bernardo,J.M., Dawid,A.P., Berger,J.O., West,M., Heckerman,D. and Bayarri,M.J. (eds), *Bayesian Statistics 7*. Oxford University Press, Oxford, 453–464.

Beal,M.J. (2003) Variational Algorithms for Approximate Bayesian Inference. PhD Thesis, Gatsby Computational Neuroscience Unit, University College London, UK.

Bolstad,B., Irizarry,R., Astrand,M. and Speed,T. (2002) A comparison of normalization methods for high density oligonucleotide array data based on bias and variance. *Bioinformatics*, **19**, 185–193.

Brown,R.G. and Hwang,P.Y. (1997) *Introduction to Random Signals and Applied Kalman Filtering*. John Wiley and Sons, NY.

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

D'Haeseleer,P., Wen,X., Fuhrman,S. and Somogyi,R. (1999) Linear modeling of mRNA expression levels during CNS development and injury. *Pac. Symp. Biocomput.*, **3**, 41–52.

Früwirth-Schnatter,S. (1995) Bayesian model discrimination and Bayes factors for linear Gaussian state space models. *J. R. Stat. Soc. B*, **57**, 237–246.

Ghahramani,Z. and Beal,M.J. (2001) Propagation algorithms for variational Bayesian learning. In Lean,T.K., Diettenich,T.G. and Tresp,V. (eds), *Advances in Neural Information Processing Systems 13*. MIT Press, Cambridge, MA, 507–513.

Hess,P., Pihan,G., Sawyers,C., Flavell,R. and Davis,R. (2000) Survival signaling mediated by c-Jun $NH(_2)$-terminal kinase in transformed B lymphoblasts. *Nat. Genet.*, **32**, 201–205.

Holter,N.S., Maritan,A., Cieplak,M., Fedoroff,N.V. and Banavar,J.R. (2001) Dynamic modeling of gene expression data. *Proc. Natl Acad. Sci. USA*, **98**, 1693–1698.

Ideker,T., Ozier,O., Schwikowski,B. and Siegel,A. (2002) Discovering regulatory and signaling circuits in molecular interaction networks. *Bioinformatics*, **18**, S233.

Kass,R.E. and Raftery,A.E. (1995) Bayes factors. *J. Am. Stat. Assoc.*, **90**, 773–795.

Lee,T.I., Rinaldi,N.J., Robert,F., Odom,D.T., Bar-Joseph,Z., Gerber,G.K., Hannett,N.M., Harbison,C.T., Thompson,C.M., Simon,I. *et al*. (2002) Transcriptional regulatory networks in *Saccharomyces cerevisiae*. *Science*, **298**, 799–804.

MacKay,D.J.C. (1992) Bayesian interpolation. *Neural Comput.*, **4**, 415–447.

Murphy,K. and Mian,S. (1999) Modelling gene expression data using Dynamic Bayesian Networks. *Technical Report*, University of California, Berkeley, CA.

Passegu,E. and Wagner,E. (2000) JunB suppresses cell proliferation by transcriptional activation of p16INK4a expression. *EMBO J.*, **19**, 2969–2979.

Perrin,B.-E., Ralaivola,L., Mazurie,A., Bottani,S., Mallet,J. and d'Alche Buc,F. (2003) Gene networks inference using dynamic Bayesian networks. *Bioinformatics*, **19**, S138–S148.

Rangel,C., Angus,J., Ghahramani,Z., Lioumi,M., Sotheran,E., Gaiba,A., Wild,D.L. and Falciani,F. (2004a) Modelling T-cell activation using gene expression profiling and state space models. *Bioinformatics*, **20**, 1361–1372.

Rangel,C., Angus,J., Ghahramani,Z. and Wild,D.L. (2004b) Modeling genetic regulatory networks using gene expression profiling and state space models. In Husmeier,D., Roberts,S. and Dybowski,R. (eds), *Probabilistic Modelling in Bioinformatics and Medical Informatics*. Springer-Verlag, Berlin, pp. 269–293.

Rangel,C., Wild,D.L., Falciani,F., Ghahramani,Z. and Gaiba,A. (2001) Modelling biological responses using gene expression profiling and linear dynamical systems. *Proceedings of the 2nd International Conference on Systems Biology*, Omipress, Madison, WI, pp. 248–256.

Rosenfeld,N. and Alon,U. (2003) Response delays and the structure of transcription networks. *J. Mol. Biol.*, **329**, 645–654.

Roweis,S. and Ghahramani,Z. (1999) A unifying review of linear Gaussian models. *Neural Comput.*, **11**, 305–345.

Shumway,R. and Stoffer,D. (1982) An approach to time series smoothing and forecasting using the EM algorithm. *J. Time Ser. Anal.*, **3**, 253–264.

van Someren,E., Wessels,L., Backer,E. and Reinders,M. (2002) Genetic network modeling. *Pharmacogenomics*, **3**, 507–525.

Weitzman,J.B. (2001) Life and death in the jungle. *Trends Mol. Med.*, **7**, 141–142.

Weitzman,J. (2000) JunD protects cells from p53-dependent senescence and apoptosis. *Mol. Cell*, **6**, 1–20.

Wu,F., Zhang,W. and Kusalik,A. (2004) Modeling gene expression from microarray expression data with state-space equations. *Pac. Symp. Biocomput.*, **9**, 581–592.