Bioinformatics Advance Access published January 7, 2015

Combining tree-based and dynamical systems for the inference of gene regulatory networks

Vân Anh Huynh-Thu [1,*] and Guido Sanguinetti [1,2,*]

[1]School of Informatics, University of Edinburgh, UK
[2]SynthSys - Systems and Synthetic Biology, University of Edinburgh, UK

Associate Editor: Dr. Igor Jurisica

ABSTRACT

Motivation: Reconstructing the topology of gene regulatory networks (GRNs) from time series of gene expression data remains an important open problem in computational systems biology. Existing GRN inference algorithms face one of two limitations: model-free methods are scalable but suffer from a lack of interpretability, and cannot in general be used for out of sample predictions. On the other hand, model-based methods focus on identifying a dynamical model of the system; these are clearly interpretable and can be used for predictions, however they rely on strong assumptions and are typically very demanding computationally.

Results: Here, we propose a new hybrid approach for GRN inference, called Jump3, exploiting time series of expression data. Jump3 is based on a formal on/off model of gene expression, but uses a non-parametric procedure based on decision trees (called "jump trees") to reconstruct the GRN topology, allowing the inference of networks of hundreds of genes. We show the good performance of Jump3 on *in silico* and synthetic networks, and applied the approach to identify regulatory interactions activated in the presence of interferon gamma.

Availability: Our MATLAB implementation of Jump3 is available at `http://homepages.inf.ed.ac.uk/vhuynht/misc/jump3.zip`.

Contact: vhuynht@inf.ed.ac.uk, G.Sanguinetti@ed.ac.uk

1 INTRODUCTION

Computational reconstruction of gene regulation from expression data is a central problem of systems biology (Alon, 2006). Gene regulation is a complex process involving multiple control steps at the chromatin, transcriptional and post-transcriptional level (Alberts *et al.*, 2008); given the difficulty in measuring and modelling all of these individual processes, the identification of a suitable abstraction and associated statistical inference methodology is vital. The gene regulatory network (GRN) abstraction aims at explaining the joint variability in the expression levels of a group of genes through a sparse pattern of interactions; elucidating the topology of GRNs can provide important insights in the fundamental biology of the system, and suggest possible intervention points in biomedical applications.

Inferring the topology of a GRN from gene expression time-series data has been a subject of intense research in computational biology over the last fifteen years (De Smet and Marchal, 2010; Bansal *et al.*, 2007; Penfold and Wild, 2011). Current approaches can be broadly divided into model-based and model-free approaches. Model-based methods start by formulating a computational model of the system, usually in the form of differential or difference equations, and recast the network inference problem as learning the parameters of such a model. In order to achieve a sparse pattern of interactions, such methods usually employ sparsity-inducing priors in a Bayesian setting, or regularisation penalties in an optimisation-based scenario. Model-based methods have many appealing qualities: the assumptions made are transparently stated and, most importantly, the generative perspective enables principled predictions of expression levels under perturbations. However, model-based methods are not free from limitations: they tend to be computationally intensive, particularly in a Bayesian setting, and their parametric nature usually implies very stringent assumptions about the dynamics (e.g. linear), which may be difficult to justify biologically. Model-free methods avoid the pitfalls of model-based methods by greedily optimising information-theoretic measures of co-variation between pairs of genes (Margolin *et al.*, 2006; Faith *et al.*, 2007; Huynh-Thu *et al.*, 2010). Such methods typically have good scalability, enabling reconstructions of networks of hundreds of genes, and have consistently achieved state-of-the-art reconstruction performance in comparative evaluations (Marbach *et al.*, 2012). The lack of an underpinning model also enables great flexibility, as the interactions between genes are not constrained to follow a parametric functional representation. Such flexibility comes at a cost though: model-free methods, by their very nature, do not have clearly defined semantics in terms of dynamical systems, and cannot be used for prediction in a straightforward way. Furthermore, incorporation of side information, which is natural in model-based methods, is generally challenging in model-free methods.

In this paper, we aim to bridge the gap between model-based and model-free methods by proposing a hybrid approach to the network inference problem, called Jump3. Our approach starts from a well defined, biologically plausible model of gene expression, the *on/off model* of gene expression (Ptashne and Gann, 2002; Ocone *et al.*, 2013), which we use to model the dynamics of individual nodes. Reconstruction of the edges is instead based on a non-parametric, tree-based method modelled on the state-of-the-art GENIE3 method (Huynh-Thu *et al.*, 2010). Adapting the tree-based method to the probabilistic setting is a novel challenge in machine learning, and involves devising a novel decision function for learning the tree. Here we introduce the "jump tree", which uses the marginal

*to whom correspondence should be addressed

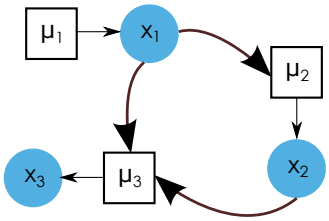

Fig. 1. Example of GRN. Circles represent the observed gene expressions, and squares represent the latent promoter states. Thick arrows model the promoter activations and show the network topology.

likelihood of the node's dynamical model as a decision function. This choice has several benefits: it embeds the tree-based learning procedure in the probabilistic model, effectively grounding it as a greedy solution to structure learning in a large latent-variable model. Furthermore, the use of the marginal likelihood means our method inherits the ease with which side information can be incorporated in probabilistic models. Our experiments with both synthetic and real data show that Jump3 has good scalability, and achieve competitive or better results than state-of-the-art alternatives.

2 MODEL AND METHODS

Here we describe Jump3, a hybrid approach for GRN inference that is based on a formal dynamical model of the expression of each gene of the GRN, and that employs a greedy, non-parametric, method for reconstructing the topology of the GRN. Exploiting time series of expression data, Jump3 assigns a confidence score to each putative regulatory link of the GRN. Note that in this paper, we leave open the problem of choosing a threshold on the weights in order to obtain a practical network, and focus on providing a ranking of the regulatory links.

2.1 Gene expression model

At the heart of our framework, we use the *on/off model* of gene expression (Ptashne and Gann, 2002), a simple, yet plausible, model where the rate of transcription of a gene can vary between two levels depending on the activity state μ of the promoter of the gene. The expression x of a gene is modelled through the following stochastic differential equation (SDE):

$$\mathrm{d}x_i = (A_i\mu_i(t) + b_i - \lambda_i x_i)\mathrm{d}t + \sigma \mathrm{d}w(t), \qquad (1)$$

where subscript i refers to the i^{th} target gene. Here, the promoter state $\mu_i(t)$ is a binary variable (the promoter is either active or inactive), which depends on the expression levels of the transcription factors that bind to the promoter (see Figure 1). $\Theta_i = \{A_i, b_i, \lambda_i\}$ is the set of kinetic parameters. A_i represents the efficiency of the promoter in recruiting polymerase when being in the active state, b_i denotes the basal transcription rate, and λ_i is the exponential decay constant of x_i. The term $\sigma \mathrm{d}w(t)$ represents a white noise-driving process with variance σ^2.

For a given trajectory of the promoter state, i.e. when we are given the states $\mu_i(t), \forall t$, the SDE (1) is linear and its solution $x_i(t)$ is equivalent to a Gaussian Markov process, i.e. an Ornstein-Uhlenbeck (OU) process (Gardiner, 1996). The mean $m_i(t)$ and covariance $c_i(t,t')$ functions of this OU process are given by:

$$m_i(t) = x_i(0)e^{-\lambda_i t} + A_i \int_0^t e^{-\lambda_i(t-\tau)}\mu_i(\tau)\mathrm{d}\tau + \frac{b_i}{\lambda_i}(1 - e^{-\lambda_i t})$$

$$c_i(t,t') = \frac{\sigma^2}{2\lambda_i}(e^{-\lambda_i|t-t'|} - e^{-\lambda_i(t+t')})$$

Note that the covariance function contains two terms, one that is stationary ($e^{-\lambda_i|t-t'|}$) and one that is non-stationary ($e^{-\lambda_i(t+t')}$). The second term is typically much smaller than the first one and thus could be neglected in practice. We however assume that a perturbation is applied to the network at $t = 0$, and we use the covariance function with its non-stationary term in order to take into account the initial transient behaviour of the network.

Let us assume that the gene expression x_i is observed with i.i.d. Gaussian noise at a finite number N of time points:

$$\hat{x}_{i,k} = x_i(t_k) + \epsilon_{i,k},$$
$$\epsilon_{i,k} \sim \mathcal{N}(0, s^2_{i,k}), k = 1, \ldots, N,$$

where $s^2_{i,k}$ is the variance of the observation noise at time point t_k. As a consequence, the observed expression levels follow a multivariate normal distribution:

$$\hat{\mathbf{x}}_i \sim \mathcal{N}(\mathbf{m}_i, C_i + D_i),$$

where $\mathbf{m}_i = [m_i(t_1), m_i(t_2), \ldots, m_i(t_N)]^\top$, $C_i \in \mathbb{R}^{N\times N}$ denotes the covariance matrix, with $C_i[k,l] = c_i(t_k, t_l)$, and $D_i \in \mathbb{R}^{N\times N}$ is a diagonal matrix with the values $s^2_{i,k}$ along the diagonal. One can therefore compute the marginal log-likelihood of the observations, given by:

$$\mathcal{L} = \log p(\hat{\mathbf{x}}_i) = -\frac{N}{2}\log(2\pi) - \frac{1}{2}\log|C_i + D_i| - \frac{1}{2}(\hat{\mathbf{x}}_i - \mathbf{m}_i)^\top (C_i + D_i)^{-1}(\hat{\mathbf{x}}_i - \mathbf{m}_i). \qquad (2)$$

Notice that this probabilistic formulation allows for a natural incorporation of replicate information by simply multiplying the likelihoods of replicate profiles.

Within this context, our goal is, for each target gene i:

1. To identify the promoter state trajectory $\boldsymbol{\mu}_i$ over the time interval $[0, t_N]$ that maximises the log-likelihood $\mathcal{L}$;
2. To identify the regulators of the target gene, i.e. the genes whose expression levels influence μ_i.

Both problems are jointly addressed by using a non-parametric approach described in the next section and illustrated in Figure 2.

2.2 Network reconstruction with jump trees

In our model, we make the assumption that the state of the promoter of a target gene i is a function of the expression levels of the genes that are direct regulators of gene i, i.e. the genes that are directly connected to gene i in the targeted network (Figure 1). Denoting by $\mathbf{x}_{\text{reg},i}(t)$ the vector containing the expression levels at time t of the regulators of gene i, we can write:

$$\mu_i(t) = f_i(\mathbf{x}_{\text{reg},i}(t)) + \xi_t, \forall t,$$

where ξ_t is a random noise with zero mean. Recovering the regulatory links pointing to gene i thus amounts to finding the genes

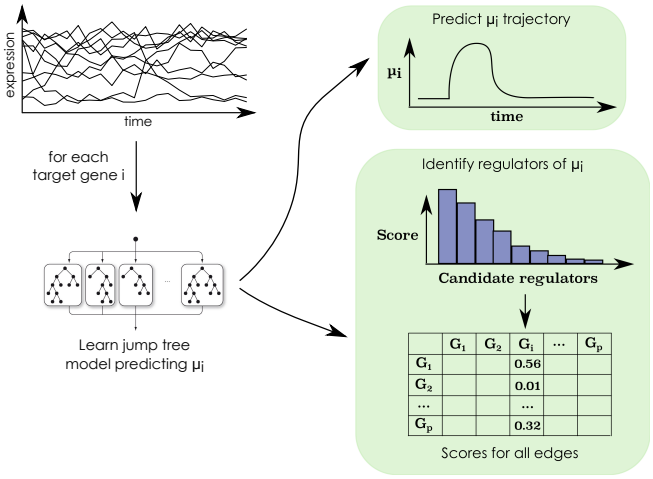

Fig. 2. The Jump3 framework. For each target gene $i = 1, \ldots, p$, a function f_i in the form of an ensemble of jump trees is learned from the time series of expression data. The trajectory of the state of the promoter of gene i (μ_i) is predicted from the jump tree model and an importance score is computed for each candidate regulator. The score of a candidate regulator j is used as weight for the regulatory link directed from gene j to gene i.

whose expression is predictive of the promoter state μ_i. To achieve this goal, we propose a procedure based on decision trees, which computes confidence scores $w_{j,i}, \forall j \neq i$, measuring the importance of each gene j in the prediction of the state μ_i.

2.2.1 Decision trees with a latent output variable Tree-based methods have been applied successfully in the inference of GRNs (Huynh-Thu *et al.*, 2010), and have appealing properties (Geurts *et al.*, 2009). First, they are non-parametric, and hence do not make any assumption about the nature of the function f_i, which can be non-linear. Another advantage of tree-based methods is their ability to detect multivariate interacting effects between features. This is a non-negligible advantage when inferring GRNs, since the regulation of gene expression is expected to be combinatorial, i.e. to involve several regulators. Tree-based methods are also essentially parameter-free, and since their computational complexity is at most linear in the number of features, they can deal with high-dimensionality.

The basic idea of our GRN inference procedure is to learn for each target gene i a model f_i in the form of a decision tree (or an ensemble of decision trees), which predicts the promoter state μ_i at any time t from the expression levels of the candidate regulators at the same time t. However, standard tree-based methods can not be applied here since the output $\mu_i(t)$ is a latent variable. We therefore propose a new decision tree algorithm called "jump tree".[1] Briefly, a jump tree is constructed top-down using a greedy algorithm, and partitions the set of observation time points into different subsets based on tests on the expression levels of the candidate regulators. Each terminal node (or leaf) of the jump tree then corresponds to a subset of time points at which μ_i is either 0 or 1. While in a standard decision tree the observations are split based on the minimisation of the entropy of the output variable, in a jump tree the split is performed based on the maximisation of the likelihood of the observations $\hat{\mathbf{x}}_i$.

More formally, the different steps for learning a jump tree predicting the latent variable μ_i are the following:

1. **Initialisation.** Start with the simplest tree, which is only composed of one leaf. This leaf contains the whole set of N observation time points, and $\mu_i(t) = 0, \forall t$, with a corresponding log-likelihood $\mathcal{L}$.
2. **Creation of a split node.** Each iteration of the greedy algorithm consists in creating a split node from a leaf $\mathcal{N}$, and updating the promoter state trajectory and the likelihood. Given the current jump tree, the current promoter state trajectory $\boldsymbol{\mu}_i$, and the current log-likelihood $\mathcal{L}$ (obtained after the previous iteration), the set $T_\mathcal{N}$ of observation time points of the leaf $\mathcal{N}$ is partitioned using the following procedure:

   a. **Definition of a split.** Given the observed expression $\hat{\mathbf{x}}_j$ of a candidate regulator $j \neq i$[2] and a threshold value c, a candidate promoter state trajectory $\boldsymbol{\mu}_i^{j,c}$ is obtained by setting:

   $$\mu_i^{j,c}(t_k) = \begin{cases} 0, \text{if } \hat{x}_j(t_k) < c, \\ 1, \text{if } \hat{x}_j(t_k) \geq c, \end{cases}$$

   for each time point $t_k \in T_\mathcal{N}$. For the time points that do not belong to $T_\mathcal{N}$, the promoter states are kept the same:

   $$\mu_i^{j,c}(t_k) = \mu_i(t_k), \forall t_k \notin T_\mathcal{N}.$$

   Between two observation time points t_k and t_{k+1}, the states $\mu_i^{j,c}(t), t_k < t < t_{k+1}$, are merely set to the state obtained at time point t_k.

   b. **Evaluation of the split.** The best candidate regulator j_* and threshold c_* are selected, i.e. those ones that yield the maximum likelihood:

   $$\begin{aligned} \mathcal{L}_* &= \max_{j,c} \mathcal{L}(\boldsymbol{\mu}_i^{j,c}), \\ (j_*, c_*) &= \arg\max_{j,c} \mathcal{L}(\boldsymbol{\mu}_i^{j,c}), \end{aligned}$$

   where $\mathcal{L}(\boldsymbol{\mu}_i^{j,c})$ is the likelihood obtained with the trajectory $\boldsymbol{\mu}_i^{j,c}$.

   c. **Decision and update.** If the likelihood is increased, i.e. $\mathcal{L}_* > \mathcal{L}$, then:

   - Replace the leaf $\mathcal{N}$ with a split node containing the optimal test "$x_{j_*} < c_*$";
   - Split $T_\mathcal{N}$ into two subsets T_0 and T_1 according to this test;
   - The child nodes of the new split node are two leaves, containing respectively T_0 and T_1;
   - Update the promoter state trajectory: $\boldsymbol{\mu}_i \leftarrow \boldsymbol{\mu}_i^{j_*,c_*}$;

[1] In stochastic process theory, the discrete variable $\mu_i(t)$ is called a jump process. The term "jump tree" thus refers to a tree that predicts such a jump process.

[2] This condition can be relaxed to incorporate autoregulation; however, in our experiments we have kept it to improve identifiability.

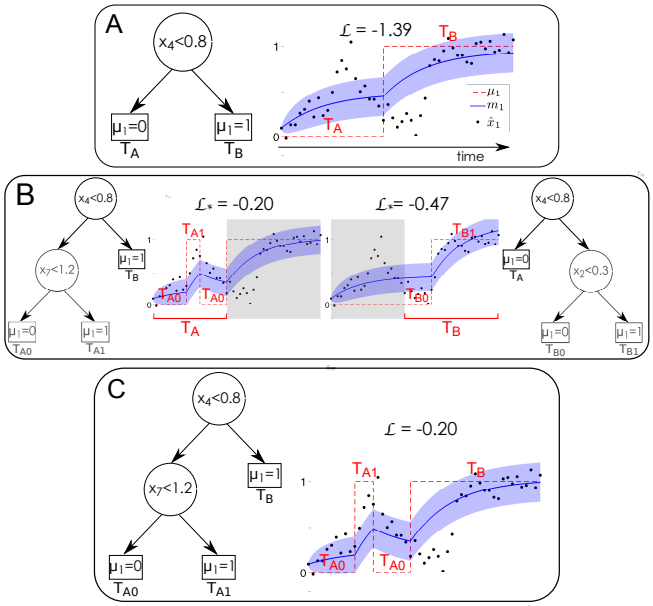

Fig. 3. Growing a jump tree predicting the state of the promoter of gene 1 (μ_1). (A) Each iteration of the jump tree algorithm results in a new tree and a new trajectory $\boldsymbol{\mu}_1$ (dashed line) yielding a likelihood $\mathcal{L}$. In this example, the current tree splits the set of observation time points in two subsets T_A and T_B, each one corresponding to a leaf of the tree. The plot also shows the posterior mean $\mathbf{m}_1$ of the expression of gene 1 (solid line), with confidence intervals (shaded area), and the observed expression levels of gene 1 (dots). (B) For each leaf of the current tree, the optimal split of the corresponding set of time points is identified. (C) The leaf for which the optimal split yields the highest likelihood is replaced with a split node.

- Update the log-likelihood: $\mathcal{L} \leftarrow \mathcal{L}_*$.

3. **Selection of the leaf.** The order in which the leaves are turned into split nodes change the final value of the likelihood $\mathcal{L}$. In our procedure, the jump tree is grown using a best-first strategy, i.e. at each iteration, steps 2.a and 2.b are repeated for each leaf of the current tree and the leaf that yields the highest maximum likelihood $\mathcal{L}_*$ is selected. Step 2.c is then applied to this leaf. This procedure is illustrated in Figure 3.
4. **Stop.** The algorithm stops when $\mathcal{L}$ can not be increased anymore. i.e. when $\mathcal{L}_* \leq \mathcal{L}$ for each leaf of the current tree. The algorithm then outputs the current jump tree and the current trajectory of the promoter state $\boldsymbol{\mu}_i$.

The jump tree pseudo-code can be found in Section 1 of the supplementary information.

2.2.2 Ensemble of decision trees A fully grown decision tree typically overfits the observed data, and significant improvements can be obtained with ensemble methods that average the predictions of several randomised trees, e.g. Random Forests (Breiman, 2001) or Extra-Trees (Geurts *et al.*, 2006).

In Jump3 we use the Extra-Trees procedure, which randomises the test at each split node of a tree (in step 2 of the jump tree algorithm). Rather than testing all the possible combinations of candidate regulator j and threshold c, the best split is determined among K random splits, each obtained by randomly selecting one candidate regulator (without replacement) and a threshold value. The prediction of $\mu_i(t)$ is then averaged over the different trees of the ensemble, yielding a probability for the promoter state to be active at time t.

2.2.3 Importance measure The learned tree-based model is used to derive an importance score for each candidate regulator, quantifying the relevance of that candidate regulator for the prediction of $\mu_i(t)$. The importance $w_{j,i}$ of a candidate regulator j is then used as weight for the putative regulatory link of the network that is directed from gene j to gene i.

We propose a measure that, at each split node $\mathcal{N}$, computes the increase of the likelihood due to the split:

$$I(\mathcal{N}) = \mathcal{L}(\boldsymbol{\mu}_i^{j_*,c_*}) - \mathcal{L}(\boldsymbol{\mu}_i),$$

where $\mathcal{L}(\boldsymbol{\mu}_i)$ and $\mathcal{L}(\boldsymbol{\mu}_i^{j_*,c_*})$ are the log-likelihoods respectively obtained before and after the split on $\mathcal{N}$. For a single tree, the overall importance $w_{j,i}$ of one candidate regulator j is then computed by summing the I values of all tree nodes where this regulator is used to split:

$$w_{j,i} = \sum_{k=1}^{n} I(\mathcal{N}_k).g(\mathcal{N}_k, j),$$

where n is the number of split nodes in the tree and $\mathcal{N}_k$ denotes the k^{th} split node. $g(\mathcal{N}_k, j)$ is function that is equal to one if the candidate regulator j is the one selected at node $\mathcal{N}_k$ and zero otherwise. The candidate regulators that are not selected at all thus obtain an importance score of zero and those ones that are selected close to the root node of the tree typically obtain high scores. Importance measures can be easily extended to ensembles of trees, by simply averaging the importances scores over all the trees of the ensemble.

2.2.4 Regulatory link ranking Each tree-based model f_i yields a separate ranking of the genes as potential regulators of a target gene i in the form of importance scores $w_{j,i}$. For a single tree, the sum of the importance scores of all candidate regulators is equal to the total increase of likelihood yielded by the tree:

$$\sum_{j \neq i} w_{j,i} = \mathcal{L}(\boldsymbol{\mu}_i) - \mathcal{L}(0),$$

where $\mathcal{L}(0)$ is the initial log-likelihood obtained with $\mu_i(t) = 0, \forall t$, and $\mathcal{L}(\boldsymbol{\mu}_i)$ is the final log-likelihood obtained after the tree has been grown. As a consequence, if we trivially order the regulatory links according to the scores $w_{j,i}$, this is likely to introduce a positive bias for the regulatory links that are directed towards the genes for which the overall likelihood increase is high. To avoid this bias, we normalise the importance scores obtained from each tree, so that they sum up to one:

$$w_{j,i} \leftarrow \frac{w_{j,i}}{\mathcal{L}(\boldsymbol{\mu}_i) - \mathcal{L}(0)}.$$

2.3 Computational complexity

Since the value of the parameter λ_i is not optimised (see the details in the supplementary information), the computation of the

Table 1. Running times for varying network sizes and numbers of observations

Network	# Genes	# Observations	Time
DREAM4	10	105	3 min
DREAM4	100	210	48 hours
IFNγ	1000	25	4 hours

covariance matrix C_i and the inversion of the matrix $C_i + D_i$, which are required for the computation of the log-likelihood $\mathcal{L}$, are done only once for each target gene. Therefore, the runtime complexity of Jump3 comes mainly from the optimisation of the parameters A_i and b_i and the matrix multiplication in the last term of Equation (2), which are iteratively repeated during tree growing. Both parameter optimisation and matrix multiplication have a complexity that is on the order of $O(N^2)$, where N is the number of observations. Let us assume for simplicity that each tree that is learned contains S splits. It can be shown that the complexity for growing an ensemble of jump trees using the Extra-Trees procedure is $O(TKS^2N^2)$, where T is the number of trees and K is the number of randomly chosen candidate regulators when searching for the optimal split at a node. The complexity of Jump3 is thus $O(pTKS^2N^2)$ since it requires to build an ensemble of trees for each of the p genes of the network. At worst, the complexity of the algorithm is thus quadratic with respect to the number of genes (when $K = p - 1$) and $O(N^4)$ with respect to the number of observations (when $S = N - 1$, i.e. each tree is fully developed with each leaf corresponding to one time point). However, this worst case scenario never happens in practice; S is usually much lower than N.

Table 1 gives an idea of the computing times, using our MATLAB implementation with K set to the number of candidate regulators and 100 trees per ensemble. These computing times were measured on a 8GB RAM, 1.7 GHz Intel core i7 computer. Note that the large amount of time required to infer a DREAM4 size-100 network is due to the high number of observations. Such a high number is usually not encountered in real datasets, where the number of observations is typically much lower than the number of genes.

The Jump3 algorithm can be easily parallelised over the p genes, as well as over the different trees of an ensemble.

2.4 Performance metrics

Jump3 provides a ranking of the regulatory links from the most confident to the least confident. To evaluate such a ranking independently of the choice of a specific threshold, we use the precision-recall (PR) curve and the area under this curve (AUPR). The PR curve plots, for different thresholds on the weights of the links, the proportion of true positives among all predictions (precision) versus the percentage of true positives that are retrieved (recall). A perfect ranking, i.e. a ranking where all the positives are located at the top of the list, yields an AUPR equal to one, while a random ranking results in an AUPR close to the proportion of positives (i.e. close to zero since the proportion of true links among all possible links in a network is usually very low).

3 RESULTS

We evaluated the proposed Jump3 procedure on several *in silico* networks as well as one synthetic network (IRMA). As a case study, we applied the procedure to expression data from macrophages treated with interferon gamma (IFNγ), in order to identify IFNγ-activated regulatory interactions. In all our experiments, ensembles of 100 trees were grown and the main parameter K of the Extra-Trees was set to the number of input candidate regulators. For the *in silico* and IRMA networks, $K = p - 1$, where p is the number of genes in the network, and $K = 40$ in the case of the IFNγ network (see later for the description of that experiment).

3.1 In silico networks

We evaluated Jump3 on the networks of the *DREAM4 In Silico Network challenge* (Marbach *et al.*, 2012; Prill *et al.*, 2010), which are 5 networks of 10 genes and 5 networks of 100 genes. For each network topology, two types of simulated expression data were used:

- *Toy data*: we simulated the expression data using the on/off model based on Equation (1). A network perturbation was simulated through a switch in the promoter state of some genes and given a set of parameters $\Theta_i = \{A_i, b_i, \lambda_i\}$ for each gene i, the model was simulated to produce continuous time series for both promoter states and gene expressions. Noisy observations at discrete time points were obtained from the expression time series by adding i.i.d. Gaussian noise. The toy data are available in the supplementary material.
- *DREAM4 data*: we applied Jump3 to the time series data that was provided in the context of the DREAM4 challenge. Each time series experiment consisted in strongly increasing or decreasing the initial expression of about one third of the genes, thereby simulating a physical of chemical perturbation. The perturbation was applied to the network at time $t = 0$ and was removed after 10 time points, making the system return to its original state.

For each network of 10 (resp. 100) genes and each simulation type, noisy observations were sampled at 21 time points under 5 (resp. 10) different network perturbations, for a total of 105 (resp. 210) observations per gene.

First, we checked the quality of the data modelling that is obtained with Jump3. Results on the toy and DREAM4 data are respectively shown in Figures 4 and S1 (in the supplementary material), for one gene of a size-100 network. We notice from a qualitative point of view that Jump3 returns a good prediction of the promoter state, and that the on/off model has sufficient flexibility to provide a good fit of the gene expression, as shown before (Opper *et al.*, 2010; Ocone *et al.*, 2013).

Next, we evaluated the performance of the method in terms of network reconstruction, and we compared it to other existing network inference procedures: two model-free methods, which are time-lagged variants of GENIE3 (Huynh-Thu, 2012) and CLR (Faith *et al.*, 2007) respectively; two model-based methods, namely Inferelator (Greenfield *et al.*, 2010) and TSNI (Bansal *et al.*, 2006); and G1DBN (Lèbre, 2009), a method based on dynamic Bayesian networks. For TSNI, a separate network was inferred for each perturbation, and a consensus network was computed as the average

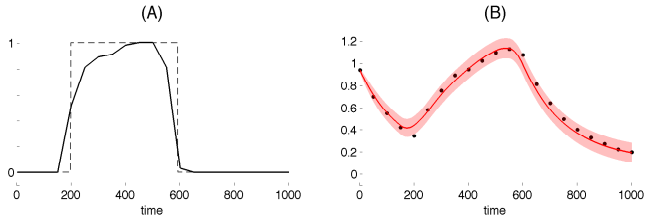

Fig. 4. Modelling results on the toy data, for one target gene. (A) Predicted promoter state $\mu_i(t)$ (solid line) versus true state (dashed line). (B) Posterior mean of gene expression $x_i(t)$, with confidence intervals. Points show observed expression $\hat{\mathbf{x}}_i$.

Table 2. AUPRs for the size-100 networks (toy data)

	Net1	Net2	Net3	Net4	Net5
Jump3	**0.342**	**0.179**	**0.299**	**0.275**	**0.264**
GENIE3-lag	0.121	0.117	0.125	0.103	0.105
CLR-lag	0.092	0.084	0.099	0.088	0.078
Inferelator	0.063	0.071	0.075	0.073	0.062
TSNI	0.017	0.022	0.017	0.023	0.021
G1DBN	0.106	0.064	0.108	0.126	0.114

Table 3. AUPRs for the size-100 networks (DREAM4 data)

	Net1	Net2	Net3	Net4	Net5
Jump3	**0.270**	**0.110**	0.200	**0.180**	**0.174**
GENIE3-lag	0.228	0.096	0.230	0.157	0.168
CLR-lag	0.179	0.109	**0.238**	0.154	0.163
Inferelator	0.126	0.101	0.198	0.147	0.148
TSNI	0.050	0.055	0.041	0.036	0.030
G1DBN	0.089	0.055	0.155	0.153	0.117

Table 4. AUPRs for the IRMA network

	Switch-on	Switch-off
Jump3	0.685	**0.682**
GENIE3-lag	0.620	0.347
CLR-lag	0.423	0.372
Inferelator	**0.718**	0.649
TSNI	0.706	0.511
G1DBN	0.600	0.313

of the different inferred networks. For all the remaining methods, networks were inferred using the complete dataset (all perturbations simultaneously).[3]

AUPR values obtained for the size-100 networks are shown in Tables 2 and 3, for the toy and DREAM4 data respectively. Results on the size-10 networks are shown in Table S2. In the case of the toy data, Jump3 yields the highest AUPR for each network. As expected, its performance decreases when the networks are inferred from the DREAM4 data, due to the mismatch between the on/off model and the one used to simulate the data. For the small networks of 10 genes, CLR, Inferelator and G1DBN have the best performances, without a clear winner. Jump3 seems robust when inferring large networks, since it outperforms the other methods on the size-100 networks.

3.2 The synthetic IRMA network

The different GRN inference methods were applied to reconstruct the IRMA network, a synthetic GRN embedded in the budding yeast *S. cerevisiae* (Cantone *et al.*, 2009). This network is composed of five genes and six regulatory interactions, and can be activated and deactivated in the presence of galactose and glucose respectively. The expression levels of the five genes were measured using quantitative RT-PCR during the transition from glucose to galactose ("switch-on" time series of 16 time points), as well as during the transition from galactose to glucose ("switch-off" time series of 21 time points).

As shown in Table 4, Jump3 is competitive with the two model-based methods (Inferelator and TSNI) when inferring the network from the switch-on data. In the case of the switch-off data, Jump3 yields the best performance. Notice that while the model-free methods (GENIE3 and CLR) typically perform better than the model-based methods on the *in silico* networks, the opposite is observed here on the IRMA network. This shows that model-based methods can be very powerful on very small networks, but their performances rapidly degrade as the number of genes in the network increases.

Promoter state predictions and gene expression fits obtained with Jump3 are shown in Figures S2 and S3.

3.3 The IFNγ network

Finally, we applied Jump3 to gene expression data from murine bone marrow-derived macrophages (Blanc *et al.*, 2011). The macrophages were treated with interferon gamma (IFNγ) and gene expression levels were measured at 25 half-hourly time points over 12 hours, using Agilent microarray platform. We focused our analysis on the 1000 genes whose expression vary the most across the time series. 40 of these genes were classified as transcription factors (TFs) by Gray *et al.* (2004), and we applied Jump3, GENIE3, and CLR in order to identify regulatory interactions between these 40 TFs and all the 1000 genes.

The 500 top-ranked regulatory links predicted by each method are shown in Figure 5A and S4.[4] As can be seen in these figures, the predicted networks are highly modular with a few TFs acting as hubs and regulating a large number of target genes (although the

[3] GENIE3 was applied with the Extra-Trees, the parameter K set to the number of candidate regulators, and ensembles of 100 trees. TSNI was used with two principal components. The other methods were run using the default values of the parameters.

[4] Cytoscape files for these three predicted IFNγ networks are also available in the supplementary material.

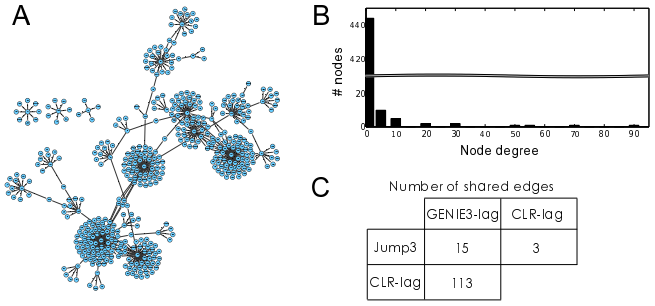

Fig. 5. (A) IFNγ network predicted by Jump3. The network was drawn using Cytoscape (Wang *et al.*, 2012). (B) Node degree distribution of the network in (A). (C) Number of shared edges between the networks predicted by Jump3, GENIE3 and CLR.

Table 5. Hubs of the predicted IFNγ networks

Jump3	Bmyc (31), Egr1 (70), Egr4 (51), Hoxc6 (89), Irf1 (30) Irf7 (12), Mrg2 (22), Myod1 (21), Pbx2 (13), Sox13 (58)
GENIE3-lag	Dlx4(18), Egr1(25), Irf1 (18), Irf7 (139), Klf4 (46), Lhx2 (58), Lyl1 (34), Pou3f1 (27), Sox13 (11), Sp100 (35), Tcfec (15)
CLR-lag	Dlx4 (14), Egr1 (15), Irf7 (53), Klf4 (41), Lhx2 (27), Lyl1 (55), Mrg1 (14), Pou3f1 (38), Sp100 (32), Stat2 (20), Tcfec (33), Tlx2 (12)

For each TF, the number of predicted targets is indicated between parenthesis.

modules of the CLR networks are less distinct). Figure 5B shows the (empirical) node degree distribution of the Jump3 network. While the networks of GENIE3 and CLR share a relatively large number of edges, Jump3 yields very different predictions (Figure 5C), indicating that the addition of a dynamical model significantly alters the networks found.

Several of the hub transcription factors (defined as transcription factors predicted as having >10 targets, and listed in Table 5) have biologically relevant annotations: apart from the interferon responsive TFs Irf1 and Irf7, we find Hoxc6 (associated with cytomegalovirus infection), and cancer-associated TFs such as Egr1, Bmyc and Pbx2, reinforcing the deep connections of the immune response with cancer (de Visser *et al.*, 2006). Quantitative evaluations of these results in terms of enrichment for known regulatory links is hampered by the absence of large-scale gold standards for human regulatory networks. The widely used TRANSFAC database[5] only reports information for a handful of TFs included in this analysis, and the number of known targets among the selected 1000 genes is usually very low (one or two at maximum), precluding a systematic enrichment analysis. The human homologues of three hub TFs (Egr1, Bmyc and Irf1) were assayed using ChIP-Seq by the ENCODE consortium (The ENCODE Project Consortium, 2012), providing a potentially much larger number of putative targets. An analysis of this data is reported in the Section 2 of the supplementary material and shows considerably higher recall for Jump3 (compared to GENIE3 and CLR), and a higher precision for two of the three TFs. Nevertheless, these numbers (only three TFs) are still very small for an enrichment analysis, which is in any case weakened by the data coming from a different organism in different experimental conditions.

4 DISCUSSION

Elucidating the topology of GRNs is a fundamental step towards our understanding of how a cell or an organism can respond to its environment. Despite years of concerted efforts by the computational biology community, this task is still far from complete, and a unified framework for GRN inference remains elusive. Here, we presented Jump3, a novel approach to GRN inference which attempts to combine the interpretability of model-based methods with the scalability of greedy, model-free methods, thus bridging the gap between the two main classes of GRN inference approaches. Experiments on simulated and synthetic data show that Jump3 is always competitive and often outperforms state-of-the-art GRN inference procedures, while an experiment on a real data set shows its potential for biologically meaningful hypothesis generation. It has good scalability with respect to the number of genes, and keeps its good performance when inferring large networks. From a modelling point of view, results show that Jump3 yields good predictions of promoter states and that, despite its simplicity, the on/off model is flexible enough to allow good fits of the data.

While we believe that Jump3 is a step in the right direction, we also acknowledge that the complexity of gene regulation will pose a strict limit to the potential of GRN inference from expression data alone. A first limitation comes from the assumption that the messenger RNA level can be used as a proxy for the protein activity, which is often not correct (Vogel and Marcotte, 2012). A simple improvement of Jump3 would thus be the exploitation of protein data, which are becoming less rare gradually, to predict the target promoter states. Another important direction is the integration in GRN inference algorithms of complementary data, such as microRNA expression, chromatin, protein-protein interactions, or microbiomes, and some promising initial steps in this direction are being taken (e.g. Ellwanger *et al.*, 2014; Greenfield *et al.*, 2013). The probabilistic generative model underlying Jump3 would allow the incorporation of additional information in a natural way via a modification of the likelihood function, while the non-parametric tree-based approach would ensure the scalability of the whole procedure.

Using the method on large networks with relatively few observations may incur in co-linearity problems, i.e. genes that have very similar profiles leading to confounding factors in the inference. A simple fix to this would be to pre-process the data with some clustering algorithm; this would further increase scalability, at the cost of some interpretability.

Ultimately, a major limitation for many studies in computational biology is the lack of systematic, large-scale gold standards on which to evaluate the models; this generalised fact reinforces the need for a tight coupling between experimental and theoretical research, and we hope that inference methods such as Jump3 could be useful in prioritising experimental designs.

[5] http://www.gene-regulation.com/pub/databases.html

ACKNOWLEDGEMENT

We thank Peter Ghazal and Thorsten Forster for useful discussions on interferon gamma biology.

Funding: The authors gratefully acknowledge support from the European Research Council under grant MLCS306999.

REFERENCES

Alberts, B., Johnson, A., Lewis, J., Raff, M., Roberts, K., and Walter, P. (2008). *Molecular Biology of the Cell*. Garland Science, New York.

Alon, U. (2006). *An Introduction to Systems Biology: Design Principles of Biological Circuits*. Chapman & Hall/CRC, London.

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

Bansal, M., Belcastro, V., Ambesi-Impiombato, A., and di Bernardo, D. (2007). How to infer gene networks from expression profiles. *Molecular Systems Biololgy*, **3**, 78.

Blanc, M., Hsieh, W. Y., Robertson, K. A., Watterson, S., Shui, G., Lacaze, P., Khondoker, M., Dickinson, P., Sing, G., Rodríguez-Martín, S., Phelan, P., Forster, T., Strobl, B., Müller, M., Riemersma, R., Osborne, T., Wenk, M. R., Angulo, A., and Ghazal, P. (2011). Host defense against viral infection involves interferon mediated down-regulation of sterol biosynthesis. *PLoS Biology*, **9**(3), e1000598.

Breiman, L. (2001). Random forests. *Machine Learning*, **45**(1), 5–32.

Cantone, I., Marucci, L., Iorio, F., Ricci, M. A., Belcastro, V., Bansal, M., Santini, S., di Bernardo, M., di Bernardo, D., and Cosma, M. P. (2009). A yeast synthetic network for in vivo assessment of reverse-engineering and modeling approaches. *Cell*, **137**(1), 172–181.

De Smet, R. and Marchal, K. (2010). Advantages and limitations of current network inference methods. *Nature Reviews Microbiology*, **8**(10), 717–29.

de Visser, K. E., Eichten, A., and Coussens, L. M. (2006). Paradoxical roles of the immune system during cancer development. *Nat. Rev. Cancer*, **6**, 24–37.

Ellwanger, D. C., Leonhardt, J. F., and Mewes, H.-W. (2014). Large-scale modeling of condition-specific gene regulatory networks by information integration and inference. *Nucleic Acids Reseach*, **42**(21), e166.

Faith, J. J., Hayete, B., Thaden, J. T., Mogno, I., Wierzbowski, J., Cottarel, G., Kasif, S., Collins, J. J., and Gardner, T. S. (2007). Large-scale mapping and validation of *Escherichia coli* transcriptional regulation from a compendium of expression profiles. *PLoS Biology*, **5**(1), e8.

Gardiner, C. W. (1996). *Handbook of Stochastic Methods*. Springer, Berlin.

Geurts, P., Ernst, D., and Wehenkel, L. (2006). Extremely randomized trees. *Machine Learning*, **36**(1), 3–42.

Geurts, P., Irrthum, A., and Wehenkel, L. (2009). Supervised learning with decision tree-based methods in computational and systems biology. *Molecular BioSystems*, **5**(12), 1593–1605.

Gray, P. A., Fu, H., Luo, P., Zhao, Q., Yu, J., Ferrari, A., Tenzen, T., Yuk, D.-i., Tsung, E. F., Cai, Z., Alberta, J. A., Cheng, L.-p., Liu, Y., Stenman, J. M., Valerius, M. T., Billings, N., Kim, H. A., Greenberg, M. E., McMahon, A. P., Rowitch, D. H., Stiles, C. D., and Ma, Q. (2004). Mouse brain organization revealed through direct genome-scale TF expression analysis. *Science*, **306**(5705), 2255–2257.

Greenfield, A., Madar, A., Ostrer, H., and Bonneau, R. (2010). DREAM4: Combining genetic and dynamic information to identify biological networks and dynamical models. *PLoS ONE*, **5**(10), e13397.

Greenfield, A., Hafemeister, C., and Bonneau, R. (2013). Robust data-driven incorporation of prior knowledge into the inference of dynamic regulatory networks. *Bioinformatics*, **29**(8), 1060–1067.

Huynh-Thu, V. A. (2012). *Machine learning-based feature ranking: Statistical interpretation and gene network inference*. Ph.D. thesis, University of Liège, Belgium.

Huynh-Thu, V. A., Irrthum, A., Wehenkel, L., and Geurts, P. (2010). Inferring regulatory networks from expression data using tree-based methods. *PLoS ONE*, **5**(9), e12776.

Lèbre, S. (2009). Inferring dynamic bayesian networks with low order independencies. *Statistical Applications in Genetics and Molecular Biology*, **8**(1), Article 9.

Marbach, D., Costello, J. C., Küffner, R., Vega, N., Prill, R. J., Camacho, D. M., Allison, K. R., the DREAM5 Consortium, Kellis, M., Collins, J. J., and Stolovitzky, G. (2012). Wisdom of crowds for robust gene network inference. *Nature Methods*, **9**(8), 796–804.

Margolin, A. A., Nemenman, I., Basso, K., Wiggins, C., Stolovitzky, G., Dalla Favera, R., and Califano, A. (2006). ARACNE: An algorithm for the reconstruction of gene regulatory networks in a mammalian cellular context. *BMC Bioinformatics*, **7**(Suppl 1), S7.

Ocone, A., Millar, A. J., and Sanguinetti, G. (2013). Hybrid regulatory models: a statistically tractable approach to model regulatory network dynamics. *Bioinformatics*, **29**(7), 910–916.

Opper, M., Ruttor, A., and Sanguinetti, G. (2010). Approximate inference in continuous time Gaussian-jump processes. In *Advances in Neural Information Processing Systems (NIPS 2010)*, volume 23.

Penfold, C. A. and Wild, D. L. (2011). How to infer gene networks from expression profiles, revisited. *Interface Focus*, **1**(6), 857–870.

Prill, R. J., Marbach, D., Saez-Rodriguez, J., Sorger, P. K., Alexopoulos, L. G., Xue, X., Clarke, N. D., Altan-Bonnet, G., and Stolovitzky, G. (2010). Towards a rigorous assessment of systems biology models: The DREAM3 challenges. *PLoS ONE*, **5**(2), e9202.

Ptashne, M. and Gann, A. (2002). *Genes and Signals*. Cold Harbor Spring Laboratory Press, New York.

The ENCODE Project Consortium (2012). An integrated encyclopedia of DNA elements in the human genome. *Nature*, **489**(7414), 57–74.

Vogel, C. and Marcotte, E. M. (2012). Insights into the regulation of protein abundance from proteomic and transcriptomic analyses. *Nature Reviews Genetics*, **13**(4), 227–232.

Wang, P. L., Lotia, S., Pico, A. R., Bader, G. D., and Ideker, T. (2012). A travel guide to cytoscape plugins. *Nature Methods*, **9**(11), 1069–1076.