A Bayesian regression approach to the inference of regulatory networks from gene expression data

Simon Rogers* and Mark Girolami
Department of Computing Science, Bioinformatics Research Centre, University of Glasgow, Glasgow, UK

ABSTRACT

Motivation: There is currently much interest in reverse-engineering regulatory relationships between genes from microarray expression data. We propose a new algorithmic method for inferring such interactions between genes using data from gene knockout experiments. The algorithm we use is the Sparse Bayesian regression algorithm of Tipping and Faul. This method is highly suited to this problem as it does not require the data to be discretized, overcomes the need for an explicit topology search and, most importantly, requires no heuristic thresholding of the discovered connections.

Results: Using simulated expression data, we are able to show that this algorithm outperforms a recently published correlation-based approach. Crucially, it does this without the need to set any ad hoc threshold on possible connections.

Availability: Matlab code which allows all experimental results to be reproduced is available at http://www.dcs.gla.ac.uk/~srogers/reg_nets.html

Contact: srogers@dcs.gla.ac.uk

Supplementary information: Appendices and supplementary figures mentioned in the text can be found at http://www.dcs.gla.ac.uk/~srogers/reg_nets.html

1 INTRODUCTION

The continued development of high-density microarrays in recent years is leading to an explosive growth in available gene expression data. These data provide measurements of the expression of thousands of genes simultaneously, and promise to be a very important tool in furthering the understanding of organisms at a cellular level. Most analysis of microarray data thus far has been interested in discovering differentially expressed genes and grouping cell samples based on their expression profiles. For example, Bhattacharjee et al. (2001) used hierarchical clustering to cluster microarray measurements for various human lung carcinomas. From this clustering, previously unknown sub-types and the corresponding genes that were differentially expressed were discovered. Spellman et al. (1998) used microarrays to discover 800 yeast genes that are regulated by the cell cycle. Recently, however, there has been a shift towards trying to understand how the synthesis of proteins is regulated and how this synthesis is affected by external stimuli. In this study, we are interested in investigating the possibility of reverse-engineering such regulatory interactions from levels of mRNA, such as those created by gene expression microarrays. Measurements of mRNA are only one glimpse into the complicated mechanism of regulation and as such, techniques based solely on them are somewhat limited. Ideally, in the future, mRNA expression would be used in conjunction with various other data-types. However, in the meantime, there is still much potential in studies such as this.

The standard approach to modelling a known network is through sets of coupled differential equations (e.g. Voit, 2000). These equations describe how the concentrations of the various products evolve over time. Although probably the most realistic model of the actual reactions taking place, such a model requires knowledge of the network structure and the various reaction rates. At the other end of the spectrum, the simplest approach is to cluster genes by their expression profiles. Genes with similar expression profiles are likely to regulate each other or be regulated by another shared parent gene. A major drawback of this approach is that it is impossible to infer any notion of causality—which gene is regulated and which gene is regulating.

There is a growing body of literature that takes a middle ground between these two levels of detail. Friedman et al. (2000) was the first such study and uses the framework of Bayesian networks to model regulatory networks in yeast. Due to the small number of arrays relative to the number of genes, a bootstrapping procedure was used to extract features (i.e. regulatory interactions) prominent in the data. The work provided some promising results, but the authors concluded that in order to infer networks from expression data alone much larger datasets were required. Other Bayesian network models have been proposed, including the use of dynamic networks to model time series data (Husmeier, 2003). Rangel et al. (2004) introduced the concept of a set of hidden state variables, producing a Kalman filter type model. Some non-Bayesian network approaches have also been proposed. Yeung et al. (2002) used singular value decomposition to produce a family of possible networks and then performed a robust regression step to extract the best one (in terms of sparsity). Similar methods have also been proposed by Weaver et al. (1999) and Zak et al. (2001, 2002).

The work presented in this paper is inspired by two recent contributions in this area, the work of Husmeier (2003) and that of Rice et al. (2004). Unlike many other publications that evaluate their particular approach on data obtained from actual microarray experiments, in both of these examples data are simulated from artificial networks. Although the realism of such synthetic data will always be questioned, microarray experiments are going to have to increase in size dramatically before it is reasonable to expect algorithms to be able to extract meaningful information (Friedman et al., 2000). Another positive for simulated data is the ability to evaluate the experimental

*To whom correspondence should be addressed.
results objectively, as the true network is known and multiple data-sets can be created. Simply quoting a number of true positives (i.e. regulatory connections discovered by the algorithm) can in itself be misleading, unless the number of false positives (connections discovered that do not exist) is also considered. The main reason for developing such an algorithm is to help narrow down the search through possible connections that must be validated in the laboratory. Any algorithm that produces a large number of false connections (even if there are a large number of true connections) is counter-productive. Both Husmeier (2003) and Rice et al. (2004) use receiver operating characteristic (ROC) analysis that allows characterisation and visualisation of the trade-off present between the number of true and false positive connections. This goes some way to making the results more objective but there is still one major problem. The ROC analysis is created by varying a threshold—i.e. some real value that the connection strengths must exceed in order for them to be considered as present. Therefore, it is only possible to get the best performance if the actual network topology is known. In practice this will not be the case. Various heuristics have been proposed (particularly in Rice et al., 2004) to overcome this caveat and they have been used with some success. The Bayesian algorithm we present here is naturally sparse—it tends towards solutions with fewer connections. This enables us to overcome the problem of setting a threshold and therefore, although it may not always perform as well as techniques requiring a threshold when the best threshold is known, it may prove to be more useful.

Let us consider in slightly more detail the experimental work of Husmeier (2003) and Rice et al. (2004). Husmeier proceeds by taking a network inspired by a regulatory network in yeast and then samples discrete data from it, almost as if it were a finite state machine. He then uses this data to induce Bayesian networks where sparsity is enforced with an appropriate prior distribution on the ‘fan-in’ i.e. the number of regulators for each gene. Central to Husmeier’s method is the discrete nature of the data. Real expression data is essentially an average from a population of cells, and as such is continuous, even if the expression in individual cells could be considered boolean. Therefore, to use a discrete Bayesian network, the data would have to be discretized. This raises the problem of how many discrete levels should be used? Too few, and too much information will be lost. Too many, and the exponential increase in the size of the conditional probability tables rapidly makes network inference computationally infeasible and the inferences statistically unreliable. It would seem more desirable to work directly with continuous data. Rice et al. (2004) does just that. A set of coupled differential equations are used to simulate a synthetic network (itself generated stochastically), which is then subject to artificial knockouts. Data is sampled from steady state, and regulatory connections are inferred by conditional correlation analysis—i.e. do we find significant correlation between genes A and B, given that gene A has been knocked out. The threshold used for ROC analysis is then the level at which the correlation coefficient must be for the connection to be considered present. The authors then develop a heuristic to get around the problem of identifying this unknown threshold. However, the number of microarray experiments that would be needed to perform this kind of analysis would be huge—30 genes, each with 10 wild-type and mutant replications, requiring 600 arrays. Although a large number of arrays is going to be required for any of the inference algorithms published so far (including our method), we argue that our inference method uses the available data in a more efficient manner, and reasonable results can be achieved with just one mutant and one wild-type array for each knockout experiment—something that would be impossible with a correlation-based procedure.

In this paper we investigate the use of a fast Sparse Bayesian linear regression algorithm, recently proposed by Tipping and Faul (2003) to infer regulatory networks from synthetic expression data. We will use the data generation process introduced by Yeung et al. (2002) and further investigated by Rice et al. (2004) but show that we can obtain comparable results with fewer arrays, whilst also going some way to overcoming the threshold selection problem. The remainder of the paper is organised as follows: Section 2 introduces the simulated system from which we will sample our data; Section 3 describes the marginal likelihood maximisation algorithm; Results and conclusions are presented in Sections 4 and 5, respectively.

2 DATA GENERATION

We have adopted the same data generation procedure as Rice et al. (2004). Details are given in Appendix 1 in the supplementary document. Briefly, a network is generated randomly according to an approximate power-law distribution on the number of regulatory connections out of each gene. This network is defined by a connectivity matrix \( A \), where row \( i \) corresponds to the connections out of gene \( g_i \). Connections have values of either +1 or −1 for excitatory and inhibitory connections, respectively. The normal steady-state of the system can then be evaluated by integrating the system of differential equations described in Appendix 1 of the supplementary document. Knock-out experiments are simulated by holding the expression of the knocked-out gene at zero during the integration. We can simulate the replication of experiments by repeatedly sampling \( N \) times from the steady-state. The result of this data generation is a \( 2R \) expression levels for each gene in each knock-out experiment—\( R \) in the normal (wild-type) system and \( R \) in the knock-out (mutant) system. In a network of \( M \) genes, where each is knocked out individually, we thus have an \( M \times (2RM) \) matrix of expression values \( E \).

At this stage, Rice et al. (2004) normalized the expression of each gene across each wild-type/mutant experiment by scaling the values to have zero mean and unit variance. Then, for each knocked-out gene, they measured the correlation between its expression and the expression of the other genes to identify those genes that were regulated by the knocked-out gene. We adopt a different approach by turning the problem around. Rather than examining the effects of knocking out a particular gene, we examine the effects other knock-outs have on this gene. That is, by taking all the data corresponding to knock-outs of genes other than our particular gene of interest, we use a sparse Bayesian regression algorithm to predict the expression of our gene from the expression of the others. Therefore, for each gene, we create a dataset consisting of all of the experiments where this gene was not knocked-out and then perform a normalization to zero mean and unit variance. In the following sections, when considering gene \( i \) we will denote this normalised input data (i.e. all data where \( i \) has not been knocked-out with the expression of \( i \) removed) as \( E \) [genes \( M \times \) experiments \( N \)] and the expression of gene \( i \) as \( e_i \) [experiments \( \times 1 \)]. Figure 1 shows a schematic comparing our approach with that of Rice et al. In the Rice experiment, a particular gene is chosen (say \( g_j \)) and knocked out. Connections between this gene and the others are then inferred by whether or not there is a change in their expression when this gene is knocked out. Therefore, there will always be problems determining whether

S.Rogers and M.Girolami
or not connections are direct or via intermediate genes. Conversely, in our approach, we knock out each gene and measure the connections into \( g_j \) by seeing if its expression changes with the knockout of the other genes. This is similar to attempts in the literature using Bayesian networks as, in essence, we are decomposing the entire network into a set of smaller networks consisting of each gene and its possible set of parents. The problem of being unable to detect whether or not connections are direct or indirect is partly overcome by the natural sparsity of the linear regression. Indirect connections are unlikely to be made, because the indirect connection will provide no information that cannot be provided by the true, direct connection. The way in which the connections into a particular gene are inferred simultaneously is lacking from methods such as correlation where, essentially, the potential connections are assumed to be independent.

### 3 FAST MARGINAL LIKELIHOOD MAXIMIZATION ALGORITHM

As briefly mentioned in the previous section, we decompose the entire network into a set of smaller networks, each corresponding to one particular gene and its possible parents. This is similar to the approach adopted in Bayesian network based methods (Friedman et al., 2000; Husmeier, 2003), as the network score can be factorized over the different genes and their prospective parents. It should be noted though that, in contrast to many Bayesian network approaches, there is no requirement for the network to be acyclic. Thus, we can decompose our system into several separate regressions and use the Marginal Likelihood maximisation algorithm of Tipping and Faul (2003) on each. Therefore, in an \( M \) gene network, we will use the algorithm \( M \) times—each time identifying a set of parents for a particular gene. In this section, we will briefly introduce the algorithm—for a more thorough description, the reader is referred to Tipping and Faul (2003). Algorithmic pseudo-code is given in Appendix 3 in the supplementary document.

We assume that the observed expression values for gene \( i \) are a result of a weighted linear combination of the expression of the other genes. Specifically,

\[
e_i = Ew_i + \zeta,
\]

where, for the sake of notational simplicity, we have assumed that the data corresponding to the knock-out of \( i \) has been removed from both \( E \) and \( e_i \). Also, the \( i \)-th row of \( E \), i.e. \( e_i \), has been removed from \( E \)—therefore, anywhere in the following description, any product over \( M \) or choice from \( M \) does not include the \( i \)-th gene. A bias term can be introduced by appending a row of ones to the design matrix and adding a corresponding weight component—we do this in all experiments. \( w_i \) denotes the weights or connections into gene \( i \). The task of the Sparse Bayesian algorithm is to infer the distribution of these weights, and hence the most probable values, given the data. All \( M \) such weight vectors assembled into one \( M \times M \) matrix (disregarding the bias) make up our approximation of the true connectivity matrix \( A \). In the following description, we will consider the inference task for just one gene, and drop the \( i \) subscript from \( w \) and \( e \).

We proceed by making the following assumptions. Firstly, the noise term \( \zeta \) is isotropic, with variance \( \sigma^2 \). Secondly, we put an hierarchical prior distribution on the weights where, crucially, each weight component has a unique Gaussian prior, with mean zero and precision (inverse variance) \( \alpha \). These priors are assumed to be independent, so the prior on the complete weight vector can be factorized over its components thus

\[
p(w|\alpha) = \prod_{j=1}^{M} \mathcal{N}(w_j|0, \alpha_j^{-1}),
\]

where we use \( \mathcal{N}(w_j|0, \alpha_j^{-1}) \) to denote that the random variable \( w_j \) is normally distributed with mean 0 and variance \( \alpha_j^{-1} \). We have also introduced the hierarchical prior on \( \alpha_j \) which is Gamma, with parameters \( a \) and \( b \); \( p(\alpha) = \mathcal{G}(a|a, b) \). This is not explicitly required for any of the further analysis and so we will not mention it again. For more information see Tipping and Faul (2003) and references therein. From Bayes theory, the posterior distribution of the weights given the data and the hyper-parameters is denoted by

\[
p(w|E, \alpha, \sigma^2) = \frac{p(E|w, \alpha, \sigma^2)p(w|\alpha)}{\int p(E|w, \alpha, \sigma^2)p(w|\alpha) \, dw},
\]

or

\[
p(w|E, \alpha, \sigma^2) = \mathcal{N}(w|\mu, \Sigma),
\]

where

\[
\mu = \sigma^{-2}\Sigma E^T e
\]

\[
\Sigma = (\text{diag}(\alpha) + \sigma^{-2}E^TE)^{-1}.
\]

So, for given values of the hyperparameters, we can analytically obtain a posterior over the weights, which is conveniently multivariate Gaussian. Unfortunately, it is analytically intractable to extend the model to full Bayesian inference over the hyperparameters, so the authors turn to a type-II maximum likelihood solution where the marginal likelihood or the evidence is maximized with respect to the hyperparameters to give point estimates that can then be substituted into the parameter posterior. The log of the marginal likelihood is given by

\[
\mathcal{L}(\alpha) = - \frac{1}{2} \left[ N \log 2\pi + \log |C| + e^T C^{-1} e \right],
\]

where

\[
C = \sigma^2 I + E(\text{diag}(\alpha)^{-1})E^T.
\]

Critically, Tipping and Faul (2003) showed that this likelihood could be decomposed into contributions from each of the \( M \) individual

---

Fig. 1. Schematic of the approach of Rice et al. and the Bayesian regression approach. (a) Conditional correlation. (b) Bayesian regression.
features (i.e. genes). This leads to a fast, iterative maximization algorithm where genes are progressively added and removed until a local maximization of the marginal likelihood is reached. Specific details of the algorithm can be found in Appendix 3 in the supplementary document. This algorithm is very well suited to our task. Firstly, as it is a regression algorithm, it works directly with the continuous expression data, so no ad hoc discretization step is required. Secondly, no discrete topology search is required, as is the case with Bayesian network based approaches. We now have a continuous topology search directly implemented in the algorithm via the marginal likelihood maximization. This means that the method is scalable to much larger networks.

4 RESULTS

We have chosen to evaluate our approach at low values of $R$—the number of replications, as this is the most realistic scenario. In all experiments, we have kept the number of genes fixed at $M = 30$, the excitatory and inhibitory parameters, $\gamma$ and $\beta$ fixed at unity and the minimum and maximum output orders for genes fixed at 1 and 5, respectively. As in Rice et al. (2004), the other parameters $\kappa$ and $\lambda$ that make up our system of differential equations have been fixed at 0.5 and 1, respectively. More details about the numerical integration of the differential equations can be found in Appendix 2 of the supplementary document.

In all experiments, we adopted the following routine. Firstly, for a chosen number of repetitions $R$ and noise level $\epsilon$, a network would be sampled according to the approximate power-law distribution given in Section 2. The system of equations defined by this network is then run to its steady state, from which $R$ levels of expression for each gene are sampled. Genes are then knocked out by forcing their expression to zero whilst the system is converging to steady state. $R$ additional samples are then measured from this new steady state. The total $2R$ samples constitute the knockout data for one gene. This is repeated for each gene in the network, giving a total of $2RM$ expression samples for each gene.

The expression data for each gene is then normalized across all knock-outs, so that it has zero mean and unit variance. An example of a dataset created from a network of $M = 10$ genes with $\epsilon = 0.1$ and $R = 5$ replicates can be found in the supplementary information. A typical $M = 30$ network can be seen in Figure 6 (All network diagrams created using Pajek—http://vlado.fmf.uni-lj.si/pub/networks/pajek/). It is important to note that when we present results for the correlation method, we have normalized data on a knock-out by knock-out basis, as described in Rice et al. (2004).

We now use the Sparse Bayesian algorithm on each gene in turn to infer its regulators. In order to do this for, say gene $j$, we remove any data corresponding to the knock-out of $j$, and separate the expression of $j$ from the expression of the other genes. In the definitions provided in Section 3, the vector of expression of $j$ is denoted by $\mathbf{u}_j$ and the $(M - 1) \times [2RM(M - 1)]$ matrix $\mathbf{E}$ holds the data for the other genes. We also add a row of 1’s to $\mathbf{E}$, corresponding to a bias term. Once all of the data have been collected for each $(R, \epsilon)$ pair, we sample a new network and repeat until we have 20 sets of results. From these we can calculate average ROC curves by varying a threshold on the $j$-th element of $\mathbf{u}_j$. ROC curves show the relationship between the sensitivity ($y$-axis—the proportion of the true edges that are discovered) against the complementary specificity ($x$-axis—one minus the proportion of non-edges that are discovered as edges). More importantly, we can calculate the average values of sensitivity and complementary specificity for the most simple possible threshold of $\mu = 0$ (no connection) and $\mu \neq 0$ (connection) is present. In the remainder of the article we refer to the performance, using this most simple of connection selection schemes as the performance with no threshold. It is worth noting here that the algorithm is performing a non-linear maximisation, and as such is not guaranteed to reach a global maximum. To investigate the variability of results, we have run the algorithm 20 times on the same dataset ($R = 2, \epsilon = 0.01$ and $M = 30$). We find that the sensitivity is constant over the 20 experiments at 0.9211, whilst the complementary specificity is 0.0209 ± 0.0022. This variability is small enough for us to be confident with running the algorithm once on each dataset. In all Sparse Bayesian experiments, updates were not performed on the noise parameter $\sigma^2$. Instead, we left the value fixed for each gene at the variance of that gene’s expression, as this gave improved performance over updating the $\sigma^2$ value as the algorithm progresses. This is due to over-fitting that can occur if $\sigma^2$ is updated. A better way to overcome this problem would be to apply a sensible prior distribution to $\sigma^2$—something that will be considered in future investigation.

The sensitivity and complementary specificity values are given for various values of $R$ and $\epsilon$ in Table 1 and the corresponding ROC curves in Figures 2–4. In these tables, we have also listed the sensitivity and complementary specificity for the correlation method when a threshold of 0.64 is used—this corresponds to the optimal value quoted in Rice et al. (2004) for a 30-node network. Although, this value is unlikely to be optimal for all of the values of $\epsilon$ and $R$ investigated here, it serves as a useful benchmark.

We can see from the ROC analysis in Figures 2–4 that in all cases, although the correlation method is able to get a higher sensitivity at high values of complementary specificity, at low values of complementary specificity, the performance of the Sparse Bayesian approach is superior. This is especially obvious when the level of noise (and hence the realism) ($\epsilon$) is high. This is probably due to the fact that the Sparse Bayesian approach makes an explicit allowance for possible noise in the data through $\sigma^2$. As we would expect, the performance of both techniques is reduced dramatically as the level of noise is increased. We also highlight that the correlation method is not able to handle single replicates. This is obvious as it is very likely that any two pairs of values, when normalized, will have a high correlation. Potentially, this means that although it would be desirable to obtain replicates, with the Sparse Bayesian approach it may not be necessary. Interestingly, we also notice that as the number of replicates is increased, the complementary specificity for the Sparse Bayesian approach actually appears to increase, especially at high levels of noise. This is not something that we would expect—quite the opposite. However, this is likely due to the fact that as more data is used, the data-dependent term (the likelihood) begins to dominate the sparse prior. One way to potentially overcome this is to continuously update the hyper-parameters governing the hierarchical Gamma prior on $\alpha$, rather than just setting them to un-informative values. This is something that will be considered in future investigation.

As mentioned earlier, the ROC analysis only gives part of the story. To obtain the best results requires the setting of a threshold, and although some methods have been proposed to automate this process (e.g. Rice et al., 2004), it would be better if no such threshold were required. The Sparse Bayesian approach goes some way to eliminating the need for a threshold. In this method, the solution found is sparse. That is, most possible connections will be zero.
Table 1. Sensitivity and complementary specificity for sparse Bayesian algorithm and correlation

<table>
<thead>
<tr>
<th>ε</th>
<th>Sparse Bayes Sensitivity</th>
<th>C. Specificity</th>
<th>Correlation Sensitivity</th>
<th>C. Specificity</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>R = 1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.01</td>
<td>0.800 ± 0.077</td>
<td>0.018 ± 0.004</td>
<td>1.000 ± 0.000</td>
<td>0.965 ± 0.001</td>
</tr>
<tr>
<td>0.02</td>
<td>0.811 ± 0.065</td>
<td>0.020 ± 0.003</td>
<td>1.000 ± 0.000</td>
<td>0.965 ± 0.001</td>
</tr>
<tr>
<td>0.05</td>
<td>0.741 ± 0.086</td>
<td>0.091 ± 0.009</td>
<td>1.000 ± 0.000</td>
<td>0.965 ± 0.001</td>
</tr>
<tr>
<td>0.10</td>
<td>0.556 ± 0.093</td>
<td>0.180 ± 0.010</td>
<td>1.000 ± 0.000</td>
<td>0.965 ± 0.001</td>
</tr>
<tr>
<td>R = 2</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.01</td>
<td>0.920 ± 0.072</td>
<td>0.010 ± 0.002</td>
<td>1.000 ± 0.000</td>
<td>0.516 ± 0.043</td>
</tr>
<tr>
<td>0.02</td>
<td>0.893 ± 0.052</td>
<td>0.016 ± 0.005</td>
<td>0.991 ± 0.015</td>
<td>0.493 ± 0.037</td>
</tr>
<tr>
<td>0.05</td>
<td>0.806 ± 0.057</td>
<td>0.148 ± 0.018</td>
<td>0.922 ± 0.044</td>
<td>0.479 ± 0.035</td>
</tr>
<tr>
<td>0.10</td>
<td>0.697 ± 0.062</td>
<td>0.267 ± 0.014</td>
<td>0.753 ± 0.060</td>
<td>0.496 ± 0.043</td>
</tr>
<tr>
<td>R = 3</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.01</td>
<td>0.963 ± 0.040</td>
<td>0.011 ± 0.003</td>
<td>1.000 ± 0.000</td>
<td>0.202 ± 0.038</td>
</tr>
<tr>
<td>0.02</td>
<td>0.960 ± 0.044</td>
<td>0.028 ± 0.007</td>
<td>0.982 ± 0.027</td>
<td>0.170 ± 0.030</td>
</tr>
<tr>
<td>0.05</td>
<td>0.922 ± 0.067</td>
<td>0.233 ± 0.023</td>
<td>0.884 ± 0.054</td>
<td>0.173 ± 0.034</td>
</tr>
<tr>
<td>0.10</td>
<td>0.836 ± 0.051</td>
<td>0.392 ± 0.014</td>
<td>0.605 ± 0.084</td>
<td>0.147 ± 0.030</td>
</tr>
</tbody>
</table>

Fig. 2. ROC comparisons between Sparse Bayesian (solid curve and square) regression and Correlation (dotted line and circle) for R = 1 repetition. (a) ε = 0.01; (b) ε = 0.02; (c) ε = 0.05; (d) ε = 0.1. The individual markers denote the values quoted in the top section of Table 1. In this, and all the following Figures, the x-axis corresponds to the complementary specificity, the y-axis sensitivity.

Fig. 3. ROC comparisons between Sparse Bayesian (solid curve and square) regression and Correlation (dotted line and circle) for R = 2 repetitions. (a) ε = 0.01; (b) ε = 0.02; (c) ε = 0.05; (d) ε = 0.1. The individual markers denote the values quoted in the middle section of Table 1.

and we can simply accept all connections that have |μ_j| greater than zero. To illustrate this point, Figure 5 shows a typical output from the two algorithms. Both give excellent results in ROC analysis (Fig. 3), but the correlation analysis depends greatly on the setting of a threshold, whilst the Sparse Bayesian approach does not. The values for sensitivity and complementary specificity for the Sparse Bayesian method, using zero as the threshold can be seen in Table 1.

Figure 6 shows the networks and the results from Figure 5 in slightly greater detail. Both ROC curves look very good (in fact, the Sparse Bayes curve is only just visible in the top left of the ROC Figure) and the network obtained from Sparse Bayes (threshold of zero) closely resembles the true network shown in Figure 6. However, the three networks obtained from correlation show how critical the setting of the threshold ρ is. A very small change in the threshold from...
Fig. 4. ROC comparisons between Sparse Bayesian (solid curve and square) regression and Correlation (dotted line and circle) for $R = 5$ repetitions. (a) $\epsilon = 0.01$; (b) $\epsilon = 0.02$; (c) $\epsilon = 0.05$; (d) $\epsilon = 0.1$. The individual markers denote the values quoted in the bottom section of Table 1.

Fig. 5. Connection weightings calculated as correlation between genes and from the Sparse Bayesian algorithm. (a) Correlation weights. (b) Sparse Bayes weights. Each row (upward from left to right) represents the weights going out from a particular gene (note, for the Sparse Bayes approach, the lowest row corresponds to the bias). The 30-gene network shown in Figure 6 was used with $R = 2$ and $\epsilon = 0.01$.

0.99 to 0.95 results in a vast increase in the number of problematic false positives.

Finally, we should point out that, due to the sparsity of the network, small values of complementary specificity can still indicate quite a number of false edges. To illustrate this, it is helpful to calculate the precision (the ratio of true edges to the total number of edges discovered). For example, for $\epsilon = 0.01$ and $R = 2$ repetitions, we obtain a precision of close to 1 (i.e. we discover very few false positives), however, if we increase the noise level to $\epsilon = 0.1$, typically the precision can drop to as low as 0.1 (i.e. for each true connection, we are discovering nine false connections), even though the complementary specificity is still reasonably low (see Table 1) owing to the vast number of possible false edges. Unfortunately, this is an unavoidable consequence of attempting to infer networks from small quantities of data in a noisy environment.

5 CONCLUSIONS

In this work, we have performed a preliminary investigation into the use of the Sparse Bayesian algorithm proposed by Tipping and Faul (2003) for inferring regulatory connections from gene knockout data. The study is far from comprehensive and there are many areas in which the work could be extended—for example, investigation of non-linear interactions between genes, simulation of real networks, using real data and optimal knockout design. However, we believe the results presented show that the technique has much promise and could be very useful.

It has several major benefits over other network inference algorithms. Firstly, it does not require explicit discretization of the data; something of an ad hoc step that can have great bearing on the output. Secondly, there is no computationally costly, explicitly discrete topology search required. Thirdly, it is able to work with (relatively) small numbers of repetitions—in fact, reasonable results are achieved with only one mutant and one wild-type array for each knock-out. Correlation-based approaches always require replicates. Finally, and most importantly, it does not require the trimming of connections using an ad hoc thresholding procedure. Most results presented previously have required knowledge of the network topology to choose the best threshold. Obviously, with an unknown network, this would be impossible. In the results presented here,
we have quoted sensitivity and specificity for the Sparse Bayes algorithm with a threshold of zero. Although it may be possible to improve on these values by thresholding, the point is that with no threshold at all the natural sparsity present in the algorithm provides reasonable performance.

The approach is by no means limited to knocking out one gene at a time. It is likely that it could be made much more efficient by performing sets of experiments where a different selection of genes at a time. It is likely that it could be made much more efficient by performing sets of experiments where a different selection of genes are knocked out in each.

In all experiments presented, we have used relatively small networks ($M = 30$ genes), as we consider these adequate for illustrative purposes. Larger experiments (with hundreds or thousands of genes) would become infeasible on a desktop PC. However, the breaking down of the problem into a separate regression for each gene means the computation can easily be done in parallel across several machines in acceptable time.

We have also not currently exploited the variance information given by the Sparse Bayesian algorithm in the weight covariance matrix $\Sigma$. It is possible that this could provide useful information regarding confidence in connections and is an interesting area for future research.

Finally, the use of knock-out data is in contrast to many publications that use time-series expression data. We believe that this is beneficial for several reasons. Firstly, it can be used in all possible regulatory systems—not just confined to those with some periodic activity. Secondly, the system only needs to be measured at steady state and hence the timing of measurements is not critical. Finally, with knock-outs, it should be possible to discover non-linear relationships with a linear model as, even if a relationship between two genes is far from linear, all we require for this method to work is some change in the steady-state condition. The magnitude of the change is irrelevant insofar as determining if a connection is present.

REFERENCES


