Vol. 00 no. 00 2007
Pages 1–7

Comparing association network algorithms for reverse engineering of large scale gene regulatory networks: synthetic vs real data

Nicola Soranzo [a], Ginestra Bianconi [b], Claudio Altafini[a*]

[a]SISSA-ISAS, International School for Advanced Studies, via Beirut 2-4, 34014 Trieste, Italy,
[b]Abdus Salam International Center for Theoretical Physics, Strada Costiera 11, 34014 Trieste, Italy

ABSTRACT

Motivation: Inferring a gene regulatory network exclusively from microarray expression profiles is a difficult but important task. The aim of this work is to compare the predictive power of some of the most popular algorithms in different conditions (like data taken at equilibrium or time courses) and on both synthetic and real microarray data. We are in particular interested in comparing similarity measures both of linear type (like correlations and partial correlations) and of nonlinear type (mutual information and conditional mutual information), and in investigating the underdetermined case (less samples than genes).

Results: In our simulations we see that all network inference algorithms obtain better performances from data produced with "structural" perturbations, like gene knockouts at steady state, than with any dynamical perturbation. The predictive power of all algorithms is confirmed on a reverse engineering problem from *E. coli* gene profiling data: the edges of the "physical" network of transcription factor–binding sites are significantly overrepresented among the highest weighting edges of the graph that we infer directly from the data without any structure supervision. Comparing synthetic and *in vivo* data on the same network graph allows us to give an indication of how much more complex a real transcriptional regulation program is with respect to an artificial model.

Availability: Software and supplementary material are freely available at the URL http://people.sissa.it/~altafini/papers/SoBiAl07/

Contact: altafini@sissa.it

1 INTRODUCTION

Of the various problems one can encounter in Systems Biology, that of reverse engineering gene regulatory networks from high throughput microarray expression profiles is certainly one of the most challenging for a number of reasons. First, the number of variables that come into play is very high, of the order of the thousands or tens of thousands at least, and there is normally no sufficient biological knowledge to restrict the analysis to a subset of core variables for a given biological process. Second, the number of gene expression profiles available is typically much less that the number of variables, thus making the problem underdetermined. Third, there is no standard model of the regulatory mechanisms for the genes, except for a generic cause–effect relationship between transcription factors and corresponding binding sites. Fourth, little is known (and no high throughput measure is available) about the post-transcriptional modification and on how they influence the regulatory pattern we see on the microarray experiments. In spite of all these difficulties, the topic of reverse engineering of gene regulatory networks is worth pursuing, as it provides the biologist with phenomenologically predicted gene–gene interactions.

Many are the methods that have been proposed for this scope in the last few years, like Bayesian networks (Friedman *et al.*, 2000), linear ordinary differential equations (ODEs) models (Yeung *et al.*, 2002), relevance networks (D'haeseleer *et al.*, 1998; Butte and Kohane, 1999) and graphical models (Kishino and Waddell, 2000; de la Fuente *et al.*, 2004; Magwene and Kim, 2004; Schäfer and Strimmer, 2005).

The aim of this work is to compare a few of these methods, focusing in particular on the last two classes of algorithms, that reconstruct weighted graphs of gene–gene interactions. Relevance networks look for pairs of genes that have similar expression profiles throughout a set of different conditions, and associate them through edges in a graph. The reconstruction changes with the "similarity measure" adopted: popular choices for gene networks are covariance-based measures like the Pearson correlation (PC) (D'haeseleer *et al.*, 1998; Butte and Kohane, 1999), or entropy-based like the mutual information (MI) (D'haeseleer *et al.*, 1998; Butte and Kohane, 2000). While PC is a linear measure, MI is nonlinear. These simple pairwise similarity methods are computationally tractable, but fail to take into account the typical patterns of interaction of multivariate datasets. The consequence is that they suffer from a high false discovery rate, i.e., genes are erroneously associated while in truth they only indirectly interact through one or more other genes.

In order to prune the reconstructed network of such false positives, one can use the notion of conditional independence from the theory of graphical modeling (Edwards, 2000), i.e., look for residual PC or MI after conditioning over one or more genes. These concepts are denoted as partial Pearson correlation (PPC) and conditional mutual information (CMI). First and second order PPC were used for this purposes in de la Fuente *et al.* (2004). If n is the number of genes, the exhaustive conditioning over $n-2$ genes is instead used in Schäfer and Strimmer (2005) under the name of graphical Gaussian models (GGM). As for MI, conceptually the CMI plays the same role of the first order PPC. In our knowledge, CMI has never been used before for gene network inference, although an alternative method for pruning the MI graph proposed in Margolin *et al.* (2006), based on the so-called Data Processing Inequality (DPI), relies on

*to whom correspondence should be addressed the same idea of conditioning, namely on searching for triplets of genes forming a Markov chain.

Since we miss a realistic large scale model of a gene regulatory network, it is not even clear how to fairly evaluate and compare these different methods for reverse engineering. A few biologically inspired (small-size) benchmark problems have been proposed, like the songbird brain model (Smith *et al.*, 2002) or the Raf pathway (Werhli *et al.*, 2006), or completely artificial networks, typically modeled as systems of nonlinear differential equations (Zak *et al.*, 2001; Mendes *et al.*, 2003). Since we are interested in large scale gene networks, we shall focus on the artificial network of Mendes *et al.* (2003), in which the genes represent the state variables and the mechanisms of gene–gene inhibition and activation are modeled using sigmoidal-like functions as in the reaction kinetics formalism. This network has several features that are useful for our purposes: (i) its size can be chosen arbitrarily; (ii) realistic (nonlinear) effects like state saturation or joint regulatory action of several genes are encoded in the model; (iii) perturbation experiments like gene knockout, or different initial conditions, or measurement noise are easily included.

Similar comparative studies have appeared recently in the literature (Werhli *et al.*, 2006; Margolin *et al.*, 2006). However, Werhli *et al.* (2006) evaluates Bayesian networks, GGM and PC relevance networks on one specific, very small (11 genes) network. Margolin *et al.* (2006) instead compares Bayesian networks, MI relevance networks and DPI using a number of expression profiles m much larger than the number of genes n, while we are also interested in more realistic scenarios. Our investigation aims at:

- comparing conditional similarity measures (like PPCs, GGM and CMI) with “static” measures (like PC and MI);
- comparing linear measures (PC and PPCs) with nonlinear ones (MI, CMI, DPI).

In particular, for the different reconstruction algorithms we are interested in the following questions:

- what is the predictive power for a number of measurements $m \ll n$? How does it grow with m?
- do the algorithms scale with size?
- what is the most useful type of experiment for the purposes of network inference?

In order to investigate a more realistic setting, the aforementioned methods were applied to a publicly available dataset of 445 gene expression profiles for 4345 genes of *E. coli*. Since a benchmark graph in this case is obviously unknown, in order to evaluate the algorithms we used the network of transcription factors–binding sites (TrF–BS) available in Salgado *et al.* (2006). Needless to say, due to the complexity of the transcriptional and post-transcriptional regulatory mechanisms of a living organism, we expect the TrF–BS network to be only partially reflected in the inferred network. Quite remarkably, though, we find that for all algorithms the 3071 edges of the TrF–BS graph are markedly over-represented among the highest weighting edges of the reconstructed network, thus showing that (i) transcription factors indeed contribute to the regulation of gene expression; (ii) the inference algorithms have some predictive power also in real systems (although the number of false positives remains unavoidably very high).

Furthermore, if we create an artificial dataset starting from the TrF–BS graph of *E. coli*, we can also compare the predictive power on an *in silico* model with that on the *in vivo* system with equal amount of information. We will see that in the regime of much less measurements than variables the differences are not so large. As a byproduct, we also have an indicative estimate of how much our artificial model is a simplification of a real transcriptional regulatory network.

2 METHODS

2.1 The artificial network

The model we used to generate artificial gene expression datasets is the reaction kinetics-based system of coupled nonlinear continuous time ODEs introduced in Mendes *et al.* (2003). The expression levels of the gene mRNAs are taken as state variables, call them x_i, $i = 1, \ldots, n$. The influence on the transcription of each gene due to the other genes is described by a (sparse) matrix of adjacencies $A = (a_{i,j})$ and the rate law for the mRNA synthesis of a gene is obtained by multiplying together the sigmoidal-like contributions of the genes identified as its inhibitors and activators. Consider the i-th row of A, $i = 1, \ldots, n$, and choose randomly a sign to its nonzero indexes. Denote by $j_1, \ldots, j_a$ the indexes with assigned positive values (activators of the gene x_i) and with $k_1, \ldots, k_b$ the negative ones (inhibitors of x_i). The ODE for x_i is then

$$\frac{dx_i}{dt} = V_i \prod_{j \in \{j_1, \ldots, j_a\}} \left(1 + \frac{x_j^{\nu_{i,j}}}{x_j^{\nu_{i,j}} + \theta_{i,j}^{\nu_{i,j}}} \right) \prod_{k \in \{k_1, \ldots, k_b\}} \frac{\theta_{i,k}^{\nu_{i,k}}}{x_k^{\nu_{i,k}} + \theta_{i,k}^{\nu_{i,k}}} - \lambda_i x_i, \qquad (1)$$

where V_i represent the basal rate of transcription, $\theta_{i,j}$ (respectively $\theta_{i,k}$) the activation (resp. inhibition) half-life, $\nu_{i,j}$ (resp. $\nu_{i,k}$) the activation (resp. inhibition) Hill coefficients (in our simulations: $\nu_{i,j}$, $\nu_{i,k} \in \{1, 2, 3, 4\}$), and λ_i the degradation rate constants. The ODE (1) always tends to a steady state, which could be 0 or a (positive) saturation value. When $x_i(0) \geqslant 0$, the abundance $x_i(t)$ remains positive during the entire time course, hence the solution is biologically consistent.

As for the topology of A, we shall consider two classes of directed networks widely used in literature as models for regulatory networks: scale-free (Barabási and Albert, 1999) and random (Erdös and Rényi, 1959).

2.1.1 *Data generated*

For the artificial network (1), a gene expression profile experiment at time t corresponds to a state vector $[x_1(t) \ldots x_n(t)]$ obtained by numerically integrating (1). For the purpose of reconstructing the network of gene–gene interactions from expression profiles, one needs to carry out multiple experiments, in different conditions, typically performed perturbing the system in many different ways. We shall consider the following cases of perturbations:

1. randomly chosen initial conditions in the integration of (1), plus gene knockout obtained setting to 0 the parameter V_i of the respective differential equation, as in Mendes *et al.* (2003);
2. only randomly chosen initial conditions in the integration of (1);

and the following types of measurements:

1. steady state measurements;
2. time-course experiments, in which the solution of the ODE is supposed to be measured at a certain (low) sampling rate.

The numerical integration of (1) is carried out in MATLAB. In all cases, a Gaussian measurement noise is added to corrupt the output.

2.2 Pearson correlation and partial Pearson correlation

Methods based on PC relevance networks were proposed already in D’haeseleer *et al.* (1998). If to each gene i we associate a random variable X_i, whose measured values we denote as $x_i(\ell)$ for $\ell = 1, \ldots, m$, the PC between the random variables X_i and X_j is

$$R(X_i, X_j) = \frac{\sum_{\ell=1}^{m}(x_i(\ell) - \bar{x}_i)(x_j(\ell) - \bar{x}_j)}{(n-1)\sqrt{v_i v_j}},$$

where $\bar{x}_i$, v_i and $\bar{x}_j$, v_j are sample means and variances of $x_i(\ell)$ and $x_j(\ell)$ over the m measurements.

Since correlation alone is a weak concept and cannot distinguish between direct and indirect interactions, (e.g. mediated by a common regulator gene), an algorithm for network inference can be improved by the use of partial correlations (de la Fuente *et al.*, 2004). The minimum first order partial correlation between X_i and X_j is obtained by exhaustively conditioning the pair X_i, X_j over all X_k. If exists $k \neq i, j$ which explains all of the correlation between X_i and X_j, then the partial correlation between X_i and Y_j becomes 0 and the pair X_i, Y_j is conditionally independent given X_k. When this happens, following Edwards (2000) we say that the triple X_i, X_j, X_k has a Markov property: on an undirected graph genes i and j are not adjacent but separated by k. This is denoted in Edwards (2000) as $X_i \perp\!\!\!\perp X_j \,|\, X_k$. In formulas, the minimum first order PPC is

$$R_{C_1}(X_i, X_j) = \min_{k \neq i,j} |R(X_i, X_j \,|\, X_k)|,$$

where

$$R(X_i, X_j \,|\, X_k) = \frac{R(X_i, X_j) - R(X_i, X_k)R(X_j, X_k)}{\sqrt{(1 - R^2(X_i, X_k))(1 - R^2(X_j, X_k))}}.$$

If $R_{C_1}(X_i, X_j) \simeq 0$ then exists k such that $X_i \perp\!\!\!\perp X_j \,|\, X_k$. Sometimes conditioning over a single variable may not be enough, and one would like to explore higher order PPCs. The minimum second order PPC for example is given by

$$R_{C_2}(X_i, X_j) = \min_{k,\ell \neq i,j} |R(X_i, X_j \,|\, X_k, X_\ell)|,$$

with

$$R(X_i, X_j \,|\, X_k, X_\ell) = \frac{R(X_i, X_j \,|\, X_k) - R(X_i, X_\ell \,|\, X_k)R(X_j, X_\ell \,|\, X_k)}{\sqrt{(1 - R^2(X_i, X_\ell \,|\, X_k))(1 - R^2(X_j, X_\ell \,|\, X_k))}}$$

and so on for higher order PPCs. Since the computation is exhaustive over all n genes, the computational cost of the algorithm for the k-th order minimum PPC is of the order of $O(n^k)$, and it becomes quickly prohibitive for $k \geqslant 2$, if n is of the order of the thousands.

The weight matrix R can be used to rank the $(n^2 - n)/2$ possible (undirected) edges of the graph. The use of PPC allows to prune the graph of many false positives computed by PC alone. However, the information provided by PC and PPC is one of *independence* or *conditional independence*, i.e., a low value of PC and PPC for a pair X_i, X_j guarantees that an edge between the two nodes is missing. A high value of the quantities $R(X_i, X_j)$ and $R_{C_1}(X_i, X_j)$ does not guarantee that i and j are truly connected by an edge, as $R_{C_2}(X_i, X_j)$ may be small or vanish.

In de la Fuente *et al.* (2004) it is shown how to choose a cut-off threshold for the weight matrices and how to combine together the effect of R, R_{C_1} and R_{C_2}.

2.3 Graphical Gaussian models

When the $n \times n$ matrix R of elements $R(X_i, X_j)$ is invertible, and we can assume that the data are drawn from a multivariate normal distribution, then the exhaustive conditioning over $n - 2$ genes can be expressed explicitly. Denote $\Omega = R^{-1}$ the concentration matrix of elements $\Omega = (\omega_{i,j})$. Then the partial correlation between X_i and X_j is

$$R_{C_{all}}(X_i, X_j) = -\frac{\omega_{i,j}}{\sqrt{\omega_{i,i}\omega_{j,j}}}.$$

When R is not full rank, then the small-sample stable estimation procedure of Schäfer and Strimmer (2005) can be used. To compute $R_{C_{all}}$, we used the R package GeneNet version 1.0.1, available from CRAN (http://cran.r-project.org).

2.4 Mutual information and conditional mutual information

In an association network, alternatively to PC and PPC, one can use the information-theoretic concept of mutual information (Butte and Kohane, 2000; Margolin *et al.*, 2006; Gardner and Faith, 2005), together with the notion of conditional independence to discern direct from indirect interdependencies. Given a discrete random variable X_i, taking values in the set $\mathcal{H}_i$, its entropy (Shannon, 1948) is defined as $H(X_i) = -\sum_{\phi \in \mathcal{H}_i} p(\phi) \log p(\phi)$, where $p(\phi)$ is the probability mass function $p(\phi) = Pr(X_i = \phi)$, $\phi \in \mathcal{H}_i$. The joint entropy of a pair of variables X_i, X_j, taking values in the sets $\mathcal{H}_i$, $\mathcal{H}_j$ respectively, is

$$H(X_i, X_j) = -\sum_{\phi \in \mathcal{H}_i, \psi \in \mathcal{H}_j} p(\phi, \psi) \log p(\phi, \psi),$$

while the conditional entropy of X_j given X_i is defined as $H(X_j \,|\, X_i) = H(X_i, X_j) - H(X_i)$. The MI of X_i, X_j is defined as $I(X_i; X_j) = H(X_i) - H(X_i \,|\, X_j)$ and can be explicitly expressed as

$$I(X_i; X_j) = \sum_{\phi \in \mathcal{H}_i, \psi \in \mathcal{H}_j} p(\phi, \psi) \log \frac{p(\phi, \psi)}{p(\phi)p(\psi)} \geqslant 0.$$

When the joint probability distribution factorizes, the MI vanishes:

$$p(\phi, \psi) = p(\phi)p(\psi) \quad \Longrightarrow \quad I(X_i; X_j) = 0. \tag{2}$$

The MI conditioned with respect to a third variable X_k is:

$$I(X_i; X_j \,|\, X_k) = H(X_i \,|\, X_k) - H(X_i \,|\, X_j, X_k)$$

or, equivalently,

$$I(X_i; X_j \,|\, X_k) = H(X_i, X_k) + H(X_j, X_k) - H(X_k) - H(X_i, X_j, X_k).$$

All pairs of nodes can be conditioned exhaustively on each of the remaining $n - 2$ nodes and the minimum of such CMIs

$$I_C(X_i; X_j) = \min_{k \neq i,j} I(X_i; X_j \,|\, X_k)$$

can be taken as a measure of conditional independence. When there exists a X_k that explains the whole MI between X_i and X_j, then the triplet has the Markov property

$$I(X_i; X_j \,|\, X_k) = 0 \qquad \Longleftrightarrow \qquad X_i \perp\!\!\!\perp X_j \,|\, X_k, \tag{3}$$

implying $I_C(X_i; X_j) = 0$, otherwise $I_C(X_i; X_j) > 0$.

Just like for the PC and PPC case, the two conditions (2) and (3) can be used to construct the graph of the gene network. I and I_C can also be combined together, and possibly with a cut-off threshold (computed e.g. through a bootstrapping method). An alternative algorithm to implement the Markov property $X_i \perp\!\!\!\perp X_j \,|\, X_k$ is proposed in Margolin *et al.* (2006). It is based on the so-called Data Processing Inequality (DPI) and consists in dropping the edge corresponding to the minimum of the triplet $I(X_i, X_j)$, $I(X_j, X_k)$ and $I(X_i, X_k)$ for all possible triplets $i \neq j \neq k$. This method is shown in Margolin *et al.* (2006) to prune the graph of many false positives. Denote I_{DPI} the matrix obtained by applying the DPI. Although I_{DPI} and I_C derive from the same notion, the information they provide is not completely redundant. In the computation of I and I_C we used the B-spline algorithm of Daub *et al.* (2004). The matrix I obtained in this way is quite similar to the MI one gets from the Gaussian Kernel method used in Margolin *et al.* (2006), see Supplement.

While the definition of CMI can be extended to higher number of conditioning variables, from a computational point of view this becomes unfeasible for n of the order of thousands: the time complexity of our algorithm for complete data matrices is $O(n^3(mp^3 + q^3))$, where p is the spline order and q is the number of bins used.

3 RESULTS

3.1 Synthetic data

In order to evaluate the algorithms, we compare each (symmetric) weight matrix with the corresponding adjacency matrix A and calculate the (standard) quantities listed in Table 1. The ROC (Receiver

Table 1. Quantities of interest in the evaluation of the algorithms

TP (true positives)	=	correctly identified true edges
FP (false positives)	=	spurious edges
TN (true negatives)	=	correctly identified zero edges
FN (false negatives)	=	not recognized true edges
recall (or sensitivity)	=	$\frac{TP}{TP+FN}$
specificity	=	$\frac{TN}{TN+FP}$
precision	=	$\frac{TP}{TP+FP}$

Operating Characteristic) and the PvsR (Precision vs Recall) curves measure the quality of the reconstruction. To give a compact description for varying m, the Area Under the Curve (AUC) of both quantities will be used. The ROC curve describes the trade-off between sensitivity and the false positive rate (1-specificity). An AUC(ROC) close to 0.5 corresponds to a random forecast, AUC(ROC) < 0.7 is considered poor, $0.7 \leqslant$ AUC(ROC) < 0.8 fair and AUC(ROC) $\geqslant 0.8$ good. For gene networks, as A is generally sparse, the ROC curve suffers from the high number of false positives. The PvsR curve instead is based only on comparing true edges and inferred edges, and therefore highlights the precision of the reconstruction (Margolin *et al.*, 2006). All the quantities we consider as well as the ROC and the PvsR curves are based on sorting the edge weights (in absolute values for PC, PPCs and GGM) and on growing the network starting from the highest weight down to the lowest one. Fixing a cut-off threshold only means altering the tail of the curves, thus we shall not make any such choice, but explore the entire range of values for the edge weights.

In Fig. 1, the results for reconstructions of random and scale-free networks of 100 genes with the different similarity measures (R, R_{C_1}, R_{C_2}, $R_{C_{\mathrm{all}}}$, I, I_C and I_{DPI}) are shown for different numbers m of measurements. AUC(ROC), AUC(PvsR) and the number of TP for a fixed value of acceptable FP (here 20) are displayed in the three columns.

By comparing the first two rows of Fig. 1 it is possible to examine the influence of the network topology on the reconstruction. Under equal conditions (type and amount of experiments), all the algorithms performed better for random networks, confirming that they are easier to infer than scale-free ones (de la Fuente *et al.*, 2004). Also another network parameter, the average degree, is influencing the performance of the algorithms: the predictive power is higher for sparser networks than for less sparse ones (see Section 2 of the Supplement).

If we now focus the attention on the scale-free topology (the most similar to known regulatory networks), it can be seen from the graphs that the performances of the reconstructions are much higher with knockout perturbations (rows 2–3) than for data produced without knockouts (row 4). This suggests that knockouts (i.e. node suppression on (1)) help in exploring the network structure, while perturbing only the initial conditions contributes very little predictive information.

Moreover, when perturbing the system with knockouts, steady state measurements (row 2) are able to generate good reconstructions with much less samples than time-course experiments (row 3), in agreement with the results of Bansal *et al.* (2007). For steady states, the performances of the algorithms improve increasing m up to n, then stabilize (for some, like GGM, even decrease). For time-course data, instead, the graphs tend to level off only when each gene has been knocked out once, regardless of the number of samples taken during the time series. This can be seen on the third row of Fig. 1, where the AUCs keep growing until 1000 samples (corresponding to 100 time series each contributing 10 samples) and only then tend to stabilize (data beyond 1000 samples are not shown in Fig. 1). The same trend can be observed increasing the number of samples per series (data not shown). Learning a network by means of time series alone (without any knockout) is very difficult as can be deduced from the low values of AUCs achieved in the forth row of Fig. 1. Notice, however, that these values get much worse (essentially random) if we consider no-knockout and steady state samples.

As for the different algorithms, the PPCs perform well in all conditions, and are significantly improving performances with respect to PC for both AUC(PvsR) and TP for fixed FP. On the contrary, applying the DPI to MI (with a tolerance of 0.1, see Margolin *et al.* (2006)) only slightly improves the precision of the MI. Since the DPI simply puts to zero the weights of the edges it considers false positives, one should not forget that DPI is penalized with respect to the other measures when computing AUC(ROC). Like PPCs, GGM gives good average results, but looks promising especially for time-course experiments, where also CMI is far superior than MI and DPI.

Finally, it is important to remark that the results we obtained for a network of 100 genes are qualitatively and quantitatively similar to those for larger gene networks: as an example in the Supplement a scale-free network of 1000 genes yields AUCs that are comparable to those shown in Fig. 1 for an equal ratio m/n.

3.2 *E. coli* network inference

We downloaded the *Escherichia coli* gene expression database M^{3D} "Many Microbe Microarrays Database" (build E_coli_v3_Build_1 from http://m3d.bu.edu, T. Gardner Lab, Boston University). This dataset consists of 445 arrays from 13 different collections corresponding to various conditions, like different media, environmental stresses (e.g DNA damaging drugs, pH changes), genetic perturbations (upregulations and knockouts), and growth phases. The experiments were all carried out on Affymetrix GeneChip *E. coli* Antisense Genome arrays, containing 4345 gene probes. A global RMA normalization was performed on the data prior to network inference. All methods described above were applied, except R_{C_2}, which is computationally very heavy for thousands of genes and behaves in much the same way as R_{C_1}. Calculating CMI took us approximately 12 days on a 3GHz processor. I_{DPI} was computed from I with a tolerance equal to 0.3 (the tolerance suggested in Margolin *et al.* (2006), 0.1, prunes 95.75% of the TrF–BS edges).

As mentioned before, we chose as "true" matrix the *E. coli K12* transcriptional network compiled in the *RegulonDB* database, version 5.6 (Salgado *et al.*, 2006), from which we derived a direct graph of 3071 interactions. As the number of possible undirected edges is 9437340, this matrix is too sparse for any of the previous statistics to be meaningful, for example AUCs(ROC) are all around 0.6. Furthermore, biologically the transcription regulation cannot be expected to be manifestly dominant over all other processes that

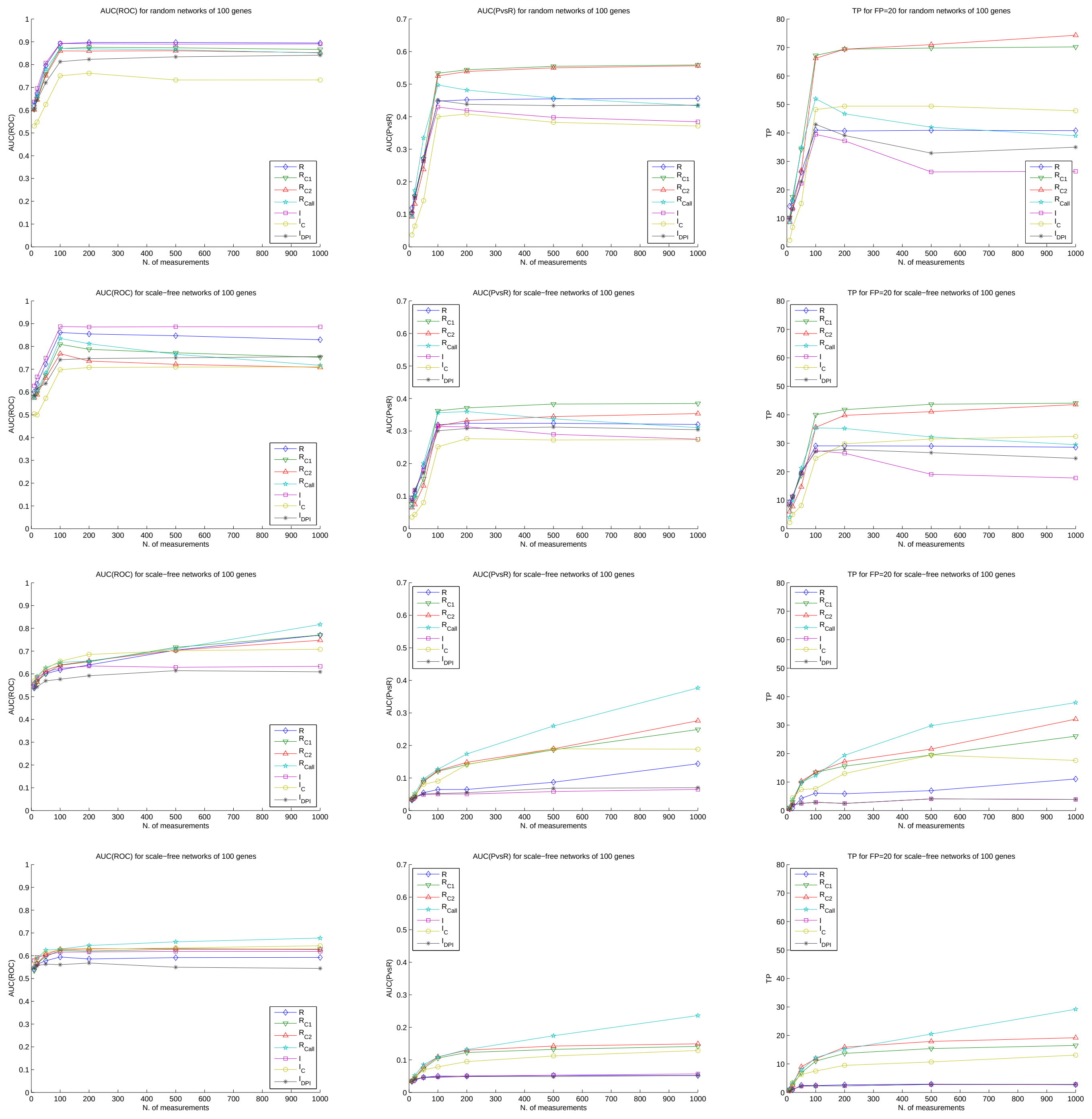

Fig. 1. Evaluating the reconstructions via R, R_{C_1}, R_{C_2}, $R_{C_{\text{all}}}$, I, I_C and I_{DPI} algorithms on 100 gene artificial networks for increasing numbers of measurements. Top row: random topology, knockout perturbations and steady state measurements. Second row: scale-free topology, knockout perturbations and steady state measurements. Third row: scale-free, knockout and time-course experiments. Fourth row: scale-free, only initial conditions perturbations and time-course experiments. On the two time courses 10 (equispaced) samples are taken on each time course. The x axis label "N. of measurements" refers to the total number of samples taken (for example 200 means 200 experiments of steady state type, but only 20 experiments on the two time courses). Left column: AUC(ROC). Central column: AUC(PvsR). Right column: number of TP for a number of FP equal to 20. Values shown are means over 10 repetitions.

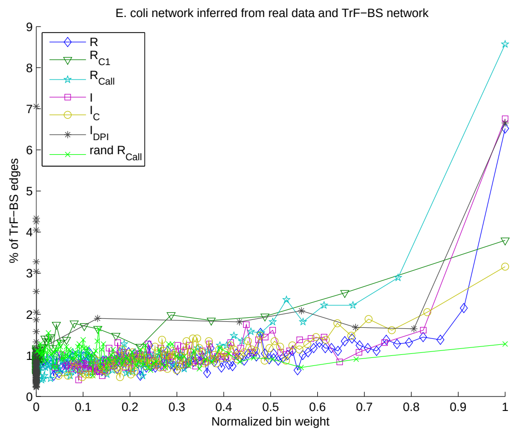

Fig. 2. Histograms showing the percentage of TrF–BS edges in each of the 100 bins in which the values of the similarity matrix (corresponding to calculated edge weights) is subdivided for the different reconstruction algorithms. The binning is according to the inferred edge weights, and each bin contains 94373 edges, and the bin weights (taken as the median of the weights of the edges in each bin) are normalized to 1. Overrepresentation towards the heaviest weighted edges is clearly visible for all the reconstruction algorithms. On the contrary, a randomization of the dataset, applied before the network reconstruction with the GGM algorithm, produces a uniform distribution of the TrF–BS edges, corresponding to a value of ≃1% on each bin.

determine the gene expression levels in a living organism. Nevertheless, if we look at the weights assigned to the TrF–BS edges (“true” edges) by the reconstruction algorithms, we see that they are significantly overrepresented in the highest weighting region (right part of the graph in Fig. 2) that in the medium/low weight ones (center/left in Fig. 2), regardless of the similarity measure adopted. To confirm the validity of our approach, we applied a randomization to the M^{3D} dataset and then inferred the network with the best reconstruction algorithm (GGM). In this case, as one would expect, the TrF–BS edges are uniformly distributed on the bins (rand $R_{C_{\rm all}}$ in Fig. 2).

If we focus only on the highest weighting bin of each reconstruction algorithm, the concordances on the identified edges (i.e. the intersection of TP) among the algorithms are shown in Table 2. Notice the high degree of concordance between correlation and mutual information.

In absolute terms, of course, there is a huge number of edges with high weights not corresponding to any TrF-BS interaction (i.e., FP), reflecting the complexity of the gene expression regulation program.

3.3 Artificial vs in vivo data, given a network

Starting from the *E. coli* TrF–BS direct graph, it is possible to create an artificial dataset using the model (1) and compare the predictive power of the algorithms on synthetic data with the previous real expression profiles. For this scope we generated the same amount of synthetic data (445 measurements), describing experiments of steady state knockout type. The same type of score based on coarse grain binning shown in Fig. 2 is shown in Fig. 3 for these synthetic data. Clearly the predictive power has grown in average, although the difference is not so drastic as one could have expected. Similarly, the concordances of TP in the top bins (Table 2) are better than on the real data for all the intersections. As expected, all these

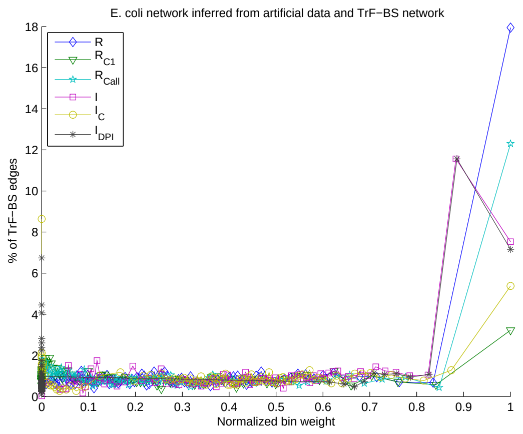

Fig. 3. Percentage of TrF–BS edges in the 100 bins in which the values of the similarity matrix is subdivided for an artificial dataset with the same graph as the *E. coli* TrF–BS. Binning is done as described in Fig. 2. Overrepresentation towards the heaviest weighting edges is on average higher than in Fig. 2.

indexes agree in saying that our artificial network is simpler that the real network, although the difference that emerges from the data is not so dramatic. Finally notice that also here concordances between unconditioned similarity measures (PC, MI) alone are very high. This confirms that conditioning allows to identify edges otherwise not detectable.

Table 2. TP concordances between similarity measures for the TrF–BS network with real and synthetic data.

algorithms	*in vivo*	*in silico*
$R, R_{C_1}, R_{C_{\rm all}}$	120	166
I, I_C, I_{DPI}	126	292
R, I	312	370
$R_{C_1}, R_{C_{\rm all}}, I_C, I_{DPI}$	58	94
all	54	94

4 DISCUSSION

For the networks generated with the model (1), we find that steady state systematic gene knockout experiments are the most informative for the purpose of reconstructing this type of networks, yielding an AUC(ROC) > 0.7 even with $m \ll n$. In particular for this class of perturbations the linear similarity measures are enough. The nonlinear measures MI and CMI instead are less precise. For time series, the situation is different: relevance networks perform poorly even when $m \gg n$. In this context, conditioned measures are relatively good. The marked difference between inference on steady state + knockouts and the more “classical” dynamical inference from time series alone without knockouts, is probably due to the highly nonlinear content of the transient evolution of (1). Reverse engineering nonlinear dynamical systems is notoriously a very difficult problem, and not even the use of nonlinear similarity measures is enough to attain a decent predictive power. At steady state, such nonlinear behavior has collapsed into a set of algebraic relations (corresponding to $dx_i/dt = 0$), which become sufficiently informative if "structurally" perturbed, e.g. by means of node suppressions. In short, structural perturbations are more efficient than dynamical perturbations for the purposes of (nonlinear) network inference.

For a real network like the one of *E. coli*, under the (biologically plausible) assumption that gene expression reflects transcriptional regulation through the TrF–BS interactions, we find that the predictive power of essentially all algorithms is certainly nonzero, and that GGM "guesses" a remarkably high number of edges, with respect to the other similarity measures, but also in absolute value, taking into account that in this case $m/n \approx 1/10$. Using the same graph to compare our artificial network and the "true" network of the *in vivo* system we do not see a dramatic difference in the predictive power between the two. This could be simply due to the above mentioned low ratio m/n.

Other interesting observations are the following:

- After a certain threshold $m_0 \geq n$ the inference ratio of all algorithms tends to stabilize. To improve the predictive capabilities, other types of perturbations should probably be used (like e.g., simultaneous multiple knockouts, external stimuli, etc.).
- AUC(ROC) around 0.9 are reached only by MI, PC and GGM in the steady state knockout simulations.
- Conditioning is useful to improve the false discovery rate, and the TP it identifies are to a large extent different from those detected without conditioning.
- Of all algorithms tested only 2nd order PPC and CMI are too computationally intensive to be used in a truly large network (tens to hundreds of thousands of genes).
- MI, CMI and DPI depend heavily on the implementation algorithm, and, at least in our B-spline implementation, on the underlying model of probability distribution (for time-course experiments the quality of the reconstruction improves considerably with the pre-application of a rank transform to the data). Correlations instead, are much less sensitive. For example replacing PC with Spearman correlation yields no substantial difference.
- The best performances vs runtime are achieved by the GGM algorithm.
- Sparse networks are easier to identify than dense (or less sparse) ones, regardless of the algorithms used, see Supplement.
- Even with $m \ll n$ (realistic situation), using steady state knockout experiments all algorithms have a decent predictive power.

5 CONCLUSION

If unsupervised graph learning problems are notoriously difficult (Edwards, 2000; Pearl, 2000), the conditions under which these problems must be studied for large scale gene regulatory network inference (less data than nodes) are even more challenging. Nevertheless, we can see through simulation and through reasonable biological assumptions on real data that the predictive power of current methods is indeed nonzero, and that a certain amount of structural information can be extracted even in this regime by means of computationally tractable algorithms, although the precision is very low and the number of false positives unavoidably very high.

REFERENCES

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

Barabási, A.-L. and Albert, R. (1999). Emergence of Scaling in Random Networks. *Science*, **286**, 509–512.

Butte, A. J. and Kohane, I. S. (1999). Unsupervised knowledge discovery in medical databases using relevance networks. *Proc. AMIA Symp.*, pages 711–715.

Butte, A. J. and Kohane, I. S. (2000). Mutual information relevance networks: functional genomic clustering using pairwise entropy measurements. *Pac. Symp. Biocomput.*, pages 418–429.

Daub, C. O., Steuer, R., Selbig, J., and Kloska, S. (2004). Estimating mutual information using B-spline functions – an improved similarity measure for analysing gene expression data. *BMC Bioinformatics*, **5**(1), 118.

de la Fuente, A., Bing, N., Hoeschele, I., and Mendes, P. (2004). Discovery of meaningful associations in genomic data using partial correlation coefficients. *Bioinformatics*, **20**(18), 3565–3574.

D'haeseleer, P., Wen, X., Fuhrman, S., and Somogyi, R. (1998). Mining the gene expression matrix: inferring gene relationships from large scale gene expression data. In R. Paton and M. Holcombe, editors, *IPCAT '97: Proceedings of the second international workshop on Information processing in cell and tissues*, pages 203–212, New York, NY, USA. Plenum Publishing.

Edwards, D. (2000). *Introduction to Graphical Modelling*. Springer.

Erdös, P. and Rényi, A. (1959). On random graphs. *Publ. Math. Debrecen*, **6**, 290–297.

Friedman, N., Linial, M., Nachman, I., and Pe'er, D. (2000). Using Bayesian networks to analyze expression data. *J. of Comput. Biol.*, **7**(3), 601–620.

Gardner, T. S. and Faith, J. J. (2005). Reverse-engineering transcriptional control networks. *Physics of Life Rev.*, **2**, 65–88.

Kishino, H. and Waddell, P. J. (2000). Correspondence analysis of genes and tissue types and finding genetic links from microarray data. In A. Dunker, A. Konagaya, S. Miyano, and T. Takagi, editors, *Genome Informatics*, volume 11, pages 83–95, Tokyo. Universal Academy Press.

Magwene, P. M. and Kim, J. (2004). Estimating genomic coexpression networks using first-order conditional independence. *Genome Biol.*, **5**(12), R100.

Margolin, A., Nemenman, I., Basso, K., Wiggins, C., Stolovitzky, G., 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.

Mendes, P., Sha, W., and Ye, K. (2003). Artificial gene networks for objective comparison of analysis algorithms. *Bioinformatics*, **19**(Suppl 2), ii122–ii129.

Pearl, J. (2000). *Causality: Models, Reasoning and Inference*. Cambridge University Press, Cambridge.

Salgado, H., Gama-Castro, S., Peralta-Gil, M., Diaz-Peredo, E., Sanchez-Solano, F., Santos-Zavaleta, A., Martinez-Flores, I., Jimenez-Jacinto, V., Bonavides-Martinez, C., Segura-Salazar, J., Martinez-Antonio, A., and Collado-Vides, J. (2006). RegulonDB (version 5.0): *Escherichia coli* K-12 transcriptional regulatory network, operon organization, and growth conditions. *Nucleic Acids Res.*, **34**(Database issue), D394–397.

Schäfer, J. and Strimmer, K. (2005). An empirical Bayes approach to inferring large-scale gene association networks. *Bioinformatics*, **21**(6), 754–764.

Shannon, C. E. (1948). A Mathematical Theory of Communication. *The Bell System Technical Journal*, **27**, 379–423, 623–656.

Smith, V. A., Jarvis, E. D., and Hartemink, A. J. (2002). Evaluating functional network inference using simulations of complex biological systems. *Bioinformatics*, **18**(Suppl 1), 216S–224.

Werhli, A. V., Grzegorczyk, M., and Husmeier, D. (2006). Comparative evaluation of reverse engineering gene regulatory networks with relevance networks, graphical gaussian models and bayesian networks. *Bioinformatics*, **22**(20), 2523–2531.

Yeung, M. K. S., Tegnér, J., and Collins, J. J. (2002). Reverse engineering gene networks using singular value decomposition and robust regression. *Proc Natl Acad Sci U S A*, **99**(9), 6163–6168.

Zak, D. E., Doyle, F. J., Gonye, G. E., and Schwaber, J. S. (2001). Simulation studies for the identification of genetic networks from cDNA array and regulatory activity data. In *Proceedings of the Second International Conference on Systems Biology*, pages 231–238.