Hindawi Publishing Corporation
EURASIP Journal on Bioinformatics and Systems Biology
Volume 2009, Article ID 617281, 12 pages
doi:10.1155/2009/617281

*Research Article*

# Reverse Engineering of Gene Regulatory Networks: A Comparative Study

**Hendrik Hache, Hans Lehrach, and Ralf Herwig**

*Vertebrate Genomics-Bioinformatics Group, Max Planck Institute for Molecular Genetics, Ihnestraße 63-73, 14195 Berlin, Germany*

Correspondence should be addressed to Hendrik Hache, hache@molgen.mpg.de

Received 3 July 2008; Revised 5 December 2008; Accepted 11 March 2009

Recommended by Dirk Repsilber

Reverse engineering of gene regulatory networks has been an intensively studied topic in bioinformatics since it constitutes an intermediate step from explorative to causative gene expression analysis. Many methods have been proposed through recent years leading to a wide range of mathematical approaches. In practice, different mathematical approaches will generate different resulting network structures, thus, it is very important for users to assess the performance of these algorithms. We have conducted a comparative study with six different reverse engineering methods, including relevance networks, neural networks, and Bayesian networks. Our approach consists of the generation of defined benchmark data, the analysis of these data with the different methods, and the assessment of algorithmic performances by statistical analyses. Performance was judged by network size and noise levels. The results of the comparative study highlight the neural network approach as best performing method among those under study.



## 1. Introduction

Deciphering the complex structure of transcriptional regulation of gene expression by means of computational methods is a challenging task emerged in the last decades. Large-scale experiments, not only gene expression measurements from microarrays but also promoter sequence searches for transcription factor binding sites and investigations of protein-DNA interactions, have spawned various computational approaches to infer the underlying gene regulatory networks (GRNs). Identifying interactions yields to an understanding of the topology of GRNs and, ultimately, of the molecular role, of each gene. On the basis of such networks computer models of cellular systems are set up and in silico experiments can be performed to test hypotheses and generate predictions on different states of these networks. Furthermore, an investigation of the system behavior under different conditions is possible [1]. Therefore reverse engineering can be considered as an intermediate step from bioinformatics to systems biology.

The basic assumption of most reverse engineering algorithms is that causality of transcriptional regulation can be inferred from changes in mRNA expression profiles. One is interested in identifying the regulatory components of the expression of each gene. Transcription factors bind to specific parts of DNA in the promoter region of a gene and, thus, effect the transcription of the gene. They can activate, enhance, or inhibit the transcription. Changes of abundances of transcription factors cause changes in the amount of transcripts of their target genes. This process is highly complex and interactions between transcription factors result in a more interwoven regulatory network. Besides the transcription factor level, transcriptional regulation can be affected as well on DNA and mRNA levels, for example, by chemical and structural modifications of DNA or by blocking the translation of mRNAs by microRNAs [2]. Usually these additional regulation levels are neglected or included as hidden factors in diverse gene regulatory models. Unfortunately, data on protein concentration measurements are currently not available in a sufficient quantity for incorporation in reverse engineering analysis. Therefore, gene expression profiles are most widely used as input for these algorithms. Probably this will change in future reverse engineering research.

Several reverse engineering methods were proposed in recent years which are based on different mathematical models, such as Boolean networks [3], linear models [4], differential equations [5], association networks [6, 7], static Bayesian networks [8], neural networks [9], state space models [10, 11], and dynamic Bayesian networks [12–14]. There are static or dynamic, continuous or discrete, linear or nonlinear, deterministic or stochastic models. They can differ in the information they provide and, thus, have to be interpreted differently. Some methods result in correlation measures of genes, some calculate conditional independencies, and others infer regulation strengths. These results can be visualized as directed or undirected graphs representing the inferred GRNs. For that, a discretization of the results is necessary for some methods. Each concept has certain advantages and disadvantages. A historical perspective of different methods applied until 2002 is given by van Someren et al. [15]. de Jong [16] and more recently Gardner and Faith [17] discuss further details and mathematical aspects.

In order to perform a comparative study we have chosen six reverse engineering methods proposed in literature based on different mathematical models. We were interested in applications for the analysis of time series. The methods should be freely downloadable, easy in use, and having only a few parameters to adjust. We included two relevance network methods; the application ARACNe by Basso et al. [6], which is based on mutual information and the package ParCorA by de la Fuente et al. [18], which calculates partial Pearson and Spearman correlation of different orders. Further, the neural network approach GNRevealer by Hache et al. [9] is compared. As an example for a Bayesian approach, the Java package Banjo [13] for dynamic models is employed. The state space model LDST proposed by Rangel et al. [10] and a graphical Gaussian model by Schäfer and Strimmer [7] in the GeneNet package are as well included in our study. We implemented the applications in a reverse engineering framework starting with artificially generated data to compare the different applications under the same conditions.

Artificial data has been used because validation and comparison of performances of algorithms have to be accomplished under controlled conditions. It would have been desirable to include experimentally determined gold standard networks that represent the knowledge of all interactions validated by single or multiple experiments. Unfortunately, there are not enough gold standard networks and appropriate experimental data available for a large comparative study. For such a study one needs a sufficiently large amount of data of different sizes, different types, that is, steady state or time series, from different experiments, for example, overexpression, perturbation, or knockdown experiments. Therefore we performed in silico experiments to obtain the required data for our performance tests.

Quackenbush [19] pointed out, that the use of artificially generated data can help to provide an understanding of how data are handled and interpreted by various methods, albeit the datasets usually do not reflect the complexity of real biological data. Their analysis involved various clustering methods. The application to synthetic datasets by computational methods is as well proposed by Mendes et al. [20] for objective comparisons. Repsilber and Kim [21] followed also the approach of using simulated data and presented a framework for testing microarray data analysis tools.

An artificial data generator has to be independent of the reverse engineering algorithms to avoid a bias in the test results. In addition, the underlying artificial GRN of a data generator has to capture certain features of real biological networks, such as the scale-free property. For this study we used the web application GeNGe [22] for the generation of scale-free networks with an mRNA and protein layer with nonlinear dynamics and performed in silico perturbation experiments.

Having specified artificial networks the computed and the true networks can be compared and algorithmic performance can be assessed with statistical measures. We used various measures in this study, such as a sensitivity, specificity, precision, distance measure, receiver operator characteristic (ROC) curves, and the area under ROC curves (AUCs).

By means of these measures we characterized the reverse engineering method performances. It is shown that the sensitivity, specificity, and precision of all analyzed methods are low under the condition of this study. Averaged over all results, the neural network approach shows the best performances. In contrast, the Bayesian network approaches identified only a few interactions correctly. We tested different sets of data, including different sizes and noises to highlight the conditions for better performances of each method.

## 2. Methods and Applications

A variety of reverse engineering methods has been proposed in recent years. Usually a computational method is based on a mathematical model with a set of parameters. These model specific parameters have to be fitted to experimental data. The models vary from a more abstract to a very detailed description of gene regulation. They can be static or dynamic, continuous or discrete, linear or nonlinear, deterministic or stochastic. An appropriate learning technique has to be chosen for each model to find the best fitting network and parameters by analyzing the data. Besides these model driven approaches, for example, followed by Bayesian networks and neural networks, there are statistical approaches to identify gene regulations, for example, relevance networks.

For this study we have chosen reverse engineering applications which belong to one of the following classes: relevance networks, graphical Gaussian models, Bayesian networks, or neural networks. In this section we will give an overview of the basic models and discuss the applications we used. All software can be downloaded or obtained from the algorithm developers. An overview is given in Table 1.

*2.1. Relvance Networks.* Methods based on relevance networks are statistical approaches that identify dependencies or similarities between genes across their expression profiles.

Table 1: *Reverse engineering applications used in this study*. The applications can be downloaded or obtained from the algorithm developers. See references for more details.

| Name | Type | Info | Reference |
|---|---|---|---|
| ARACNe | relevance network with mutual information | C command line | Basso et al. [6] |
| ParCorA | relevance network with partial Pearson or Spearman correlation | C command line | de la Fuente et al. [18] |
| GNRevealer | neural network | C++ command line | Hache et al. [9] |
| Banjo | Bayesian network | Java command line | Yu et al. [13] |
| LDST | state space model | Matlab script | Rangel et al. [10] |
| GeneNet | graphical Gaussian model | R script | Schäfer and Strimmer [7] |

They do not incorporate a specific model of gene regulation. In a first step correlation is calculated for each pair of genes based on different measures, such as Pearson correlation, Spearman correlation, and mutual information. The widely used Pearson correlation indicates the strength of a linear relationship between the genes. In contrast to that Spearman's rank correlation can detect nonlinear correlations as well as mutual information. It is assumed that a nonzero correlation value implies a biological relationship between the corresponding genes. The algorithm ARACNe developed by Basso et al. [6] uses the Data Processing Inequality (DPI) for that purpose. In each triplet of fully connected nodes in the network obtained after the first step, the edges with the lowest mutual information will be removed. In contrast, de la Fuente et al. [18] use partial correlations in their proposed method to eliminate indirect interactions. A partial correlation coefficient measures the correlation between two genes conditioning on one or several other genes. The number of genes conditioning the correlation determines the order of the partial correlation. In the program package ParCorA by de la Fuente et al. [18] the partial correlations up to 3rd order for Pearson and 2nd order for Spearman correlation are implemented. We compared all provided correlation measures.

An inferred network from a relevance network method is undirected by nature. Furthermore, statistical independence of each data sample is assumed, that is, that measurements of gene expression at different time points are assumed to be independent. This assumption ignores the dependencies between time points. Nevertheless, we applied these methods on simulated time series data to study the predictive power of these approaches.

*2.2. Graphical Gaussian Models.* Graphical Gaussian models are frequently used to describe gene association networks. They are undirected probabilistic graphical models that allow to distinguish direct from indirect interactions. Graphical Gaussian models behave similar as the widely used Bayesian networks. They provide conditional independence relations between each gene pair. But in contrast to Bayesian networks graphical Gaussian models do not infer causality of a regulation.Graphical Gaussian models use partial correlation conditioned on all remaining genes in the network as a measure of conditional independence. Under the assumption of a multivariate normal distribution of the data the partial correlation matrix is related to the inverse of the covariance matrix of the data. Therefore the covariance matrix has to be estimated from the given data and to be inverted. From that the partial correlations can be determined. Afterwards a statistical significance test of each nonzero partial correlation is employed.

We used the graphical Gaussian implementation GeneNet by Schäfer and Strimmer [7]. It is a framework for small-sample inference with a novel point estimator of the covariance matrix. An empirical Bayes approach to detect statistically significant edges is applied to the calculated partial correlations.

*2.3. Neural Networks.* A neural network can be considered as a model for gene regulation where each node in the network is associated with a particular gene. The value of the node is the corresponding gene expression value. A directed edge between nodes represents a regulatory interaction with a certain strength indicated by the edge weight. The dynamic of a time-discrete neural network of $n$ nodes is described by a system of nonlinear update rules for each node value $x_i$:

$$x_i[t+\Delta t] = x_i[t] + \Delta t\left[a_i S\left(\sum_j w_{ij}x_j[t] + b_i\right) - d_i x_i[t]\right] \quad \forall i \le n. \tag{1}$$

The parameters of the model are the weights $\mathbf{W} := \{w_{ij} \mid i, j = 1,\ldots,n\}$, where $w_{ij}$ represents the influence of node $j$ on node $i$, activation strengths $\mathbf{a} := \{a_i \mid i = 1,\ldots,n\}$, bias parameters $\mathbf{b} := \{b_i \mid i = 1,\ldots,n\}$, and degradation rates $\mathbf{d} := \{d_i \mid i = 1,\ldots,n\}$. The effects of all regulating nodes are added up and have a combined effect on the connected node. The sigmoidal activation function $S(x) = (1+e^{-x})^{-1}$ realizes a saturation of the regulation strength. Self-regulation and degradation are implemented in the mathematical model as well.

A learning strategy for the parameters is the Backpropagation through time (BPTT) algorithm described by Werbos [23] and applied to genetic data by Hache et al. [9]. The

BPTT algorithm is an iterative, gradient-based parameter learning method which minimizes the error function:

$$E(\mathbf{x}, \hat{\mathbf{x}}) := \frac{1}{2} \sum_t \sum_i [x_i[t] - \hat{x}_i[t]]^2 \tag{2}$$

by varying the parameters of the model $(\mathbf{W}, \mathbf{a}, \mathbf{b}, \mathbf{d})$ during every iteration step. $\mathbf{x}$ is the computed values vector and the values $\hat{\mathbf{x}}$ are the given expression data of the mRNAs at discrete time points. The computed matrix $\mathbf{W}$ of regulation strength is a matrix of real values, which has to be discretized to obtain a binary or ternary matrix, representing, ultimately, the topology.

*2.4. Dynamic Bayesian Networks.* A Bayesian network is a stochastic probabilistic graphical network model defined by a directed acyclic graph (DAG) which represents the topology and a family of conditional probability distributions. In contrast to other models nodes represent random variables and edges conditional dependence relations between these random variables. A dynamic Bayesian network is an unfolded static Bayesian network over discrete time steps. Assuming that nodes are only dependent of direct parents in the previous time layer, the joint probability distribution of a dynamic Bayesian network can be factorized:

$$P(\mathcal{X}) = P(\mathcal{X}[0]) \prod_t \prod_i P\left(X_i[t] \mid \mathcal{X}_{\mathrm{pa}[i]}[t - \Delta t]\right), \tag{3}$$

where $\mathcal{X} = \{\mathcal{X}_1[0], \ldots, \mathcal{X}_n[t]\}$ is the set of random variables $\mathcal{X}_i[t]$ with value $x_i[t]$ for each node $i$ at time $t$. $\mathcal{X}_{\mathrm{pa}[i]}[t - \Delta t]$ represents the set of parents of node $i$ in the previous time slice $t - \Delta t$. The temporal process is Markovian and homogeneous in time, that means a variable $\mathcal{X}_i[t]$ is only dependent of parents at the time point $t - \Delta t$ and the conditional distribution does not change over time, respectively.

For discrete random variables the conditional probability distributions can be multinomial. With such a distribution nonlinear regulations can be modeled, but a discretization of continuous data is needed. The number of parameters in such a model increases exponentially with the number of parents per node. Therefore, this number is often restricted by a maximum. The program package Banjo by Yu et al. [13], which we used in this study as an representative for a Bayesian method, follows a heuristic search approach. It seeks in the network space for the network graph with the best score, based on the Bayesian Dirichlet equivalent (BDe) score. A score here is a statistical criterion for model selection. It can be based on the marginal likelihood $P(\mathcal{D}|\mathcal{G})$ for a dataset $\mathcal{D}$ given a graph structure $\mathcal{G}$. The BDe score is a closed form solution for the integration of marginal likelihood, derived under the assumption of a multinomial distribution with a Dirichlet prior. See, for example, Heckerman et al. [24] for more details. It requires discrete values as input. A discretization is performed by the program. For that, two methods are provided; interval and quantile discretization. The number of discretization levels can be specified as well. We used the quantile discretization with five levels. The output network of Banjo is a signed directed graph.

*2.5. State Space Models.* A further reverse engineering approach is a state space model. They constitute a class of dynamic Bayesian networks where it is assumed that the observed measurements depend on some hidden state variables. These hidden variables capture the information of unmeasured variables or effects, such as regulating proteins, excluded genes in the experiments, degradations, external signals, or biological noise.

A state space model is proposed by Schäfer and Strimmer [7]. The model for gene expression includes crosslinks from an observational layer to a hidden layer:

$$\begin{aligned} \mathbf{x}_t &= A\mathbf{x}_{t-1} + B\mathbf{y}_{t-1} + \mathbf{w}_t, \\ \mathbf{y}_t &= C\mathbf{x}_t + D\mathbf{y}_{t-1} + \mathbf{v}_t. \end{aligned} \tag{4}$$

Here, $\mathbf{y}_t$ denotes the gene expression levels at time $t$ and $\mathbf{x}_t$ the unobserved hidden factors. The matrix $D$ captures gene-gene expression level influences at consecutive time points and the matrix $C$ denotes the influence of the hidden variables on gene expression level at each time point. Matrix $B$ models the influence of gene expression values from previous time points on the hidden states and $A$ is the state dynamics matrix. The matrix $CB + D$ has to be determined, which captures not only the direct gene-gene interactions but also the regulation through hidden states over time. A nonzero matrix element $[CB + D]_{ij}$ denotes activation or inhibition of gene $j$ on gene $i$ depending on its sign.

## 3. Data

For the comparative study of reverse engineering methods we generated a large amount of expression profiles from various GRNs and different datasets. We performed in silico perturbation experiments by varying the initial conditions randomly within the network and data generator GeNGe [22]. A discretization step is followed if required by the reverse engineering application internally, for example, by DBN with a quantile discretization.

In a first step we generated random scale-free networks in GeNGe to obtain GRNs of different sizes. Directed scale-free networks are generated in GeNGe with an algorithm proposed by Bollobás et al. [25], for each generated network a mathematical model of gene regulation is constructed. We assembled a two-layer system, with an mRNA and a protein layer. The kinetics of the concentration of an mRNA and protein pair, associated to an arbitrary gene, are described by

$$\begin{aligned} \frac{\mathrm{d}[\mathrm{mRNA}]}{\mathrm{d}t} &= k_1 \varphi_\nu(x_{t_1}, \ldots, x_{t_\nu}) - k_2[\mathrm{mRNA}], \\ \frac{\mathrm{d}[\mathrm{Protein}]}{\mathrm{d}t} &= k_3[\mathrm{mRNA}] - k_4[\mathrm{Protein}], \end{aligned} \tag{5}$$

where $k_1$ and $k_3$ are the maximal transcription rate of the mRNA and translation rate of the corresponding protein, respectively. $k_2$ and $k_4$ are the degradation rates. $\varphi_\nu(x_{t_1}, \ldots, x_{t_\nu})$ is dependent of $\nu$ concentrations $\{x_{t_i}\}$ of the proteins acting as transcription factors of the gene. A transcription factor is indexed by $t_i \in \mathcal{T}$. Note that all the

parameters, $k_1, \ldots, k_4, \nu$, the transcription function $\varphi_\nu$, and the set $\mathcal{T}$ of transcription factor indices are gene specific and can vary between genes.

In the GRN models we used the logic described by Schilstra and Nehaniv [26] for the transcription kinetics $\varphi_\nu$. We distinguish between input genes, which have no regulatory elements in the model and regulated genes, which have at least one regulator. Input genes have a constant linear production. In contrast, regulated genes have no such production. They can only be expressed, if a regulator bounds to the corresponding promoter region of the DNA. Therefore, the transcription factors are essential for the expression of such genes. Other kinetic schemata are also conceivable but not considered here. With the assumption of noncompetitive binding and an individual, gene-dependent regulation strengths $\{a_{t_i}\}$ of each transcription factor $t_i \in \mathcal{T}$ of the gene, we derived the kinetic law:

$$\varphi_\nu(x_{t_1}, \ldots, x_{t_\nu}) = \begin{cases} \displaystyle\prod_{i=1}^{\nu}\left[1 + (2^{a_{t_i}} - 1)\frac{x_{t_i}}{1 + x_{t_i}}\right] & \\ \qquad \displaystyle -\prod_{i=1}^{\nu}\frac{1}{1 + x_{t_i}}, & \text{for} \quad \nu \neq 0, \\ 1, & \text{for} \quad \nu = 0. \end{cases} \tag{6}$$

A regulation strength $a_{t_i} > 0$ of transcription factor $t_i$ stands for activation and $a_{t_i} < 0$ for inhibition of the gene's transcription. The second term in the first case of (6) implements the assumption that regulated genes do not have a constant production rate. In each generated network we set 70% of all regulators as activators and the others as inhibitors. This ratio is arbitrarily chosen, but is motivated by the network proposed by Davidson et al. [27], where more activators than inhibitors can be found. The regulation strengths $\{a_{t_i}\}$ are randomly chosen from a uniform distribution over the interval $(0, 4)$ and $(0, -4)$ for activators and inhibitors, respectively.

Time series of mRNAs are obtained by first drawing randomly the initial concentrations of each component of the model from a normal distribution with the steady state value of this component as mean and 0.2 as coefficient of variation. Steady states are determined numerically in sufficiently long presimulations where changes of concentrations did not anymore occur. The simulations are then performed using the initial conditions. With this approach we simulated global perturbations of the system. We inspected the time series and selected all time series which show similar behavior, that is, relaxation in the same steady state over time. From the simulated mRNA values we picked 5 values at different time steps during the relaxation of the system as the input data of all reverse engineering algorithms. Note that all values are in an arbitrary unit system.

To simulate experimental errors we added Gaussian noise with different coefficient of variations (cvs) to each expression value in a final step of data generation. The mean of the Gaussian distribution is the unperturbed value. The cv represents the level of noise.

We investigated the impact of different numbers of time series of mRNAs and noise levels on the reconstruction results. For this study we generated randomly five networks of sizes 5, 10, 20, and 30 nodes each. For each network we simulated 5, 10, 20, 30, and 50 time series by repeating the simulation accordingly with different initial values, as described above. For a network of size ten and ten time series, the data matrix contains 500 values (10 nodes $\times$ 10 time series $\times$ 5 time points). We added to the profiles noise with cvs equal to 0.0, 0.1, 0.2, 0.3, 0.4, and 0.5. After that we took from each time series five equidistant time points in the region, where changes in the expression profiles occur. Hence, each reverse engineering application had to analyze 600 datasets (5 $\times$ 4 network sizes $\times$ 5 time series sets $\times$ 6 noise levels). All datasets and models are provided as supplementary material.

## 4. Postprocessing

For all results of the relevance network, graphical Gaussian model, and neural network approaches we performed a postprocess to obtain a resulting network. Most of the entries in the resulting matrices are unequal to zero. This represents a nearly fully connected graph. In contrast the true input networks are sparse. Hence, we discretized each output matrix, representing the correlations or regulation weights between the genes, using an optimized threshold for each method. Such threshold minimizes the distance measure:

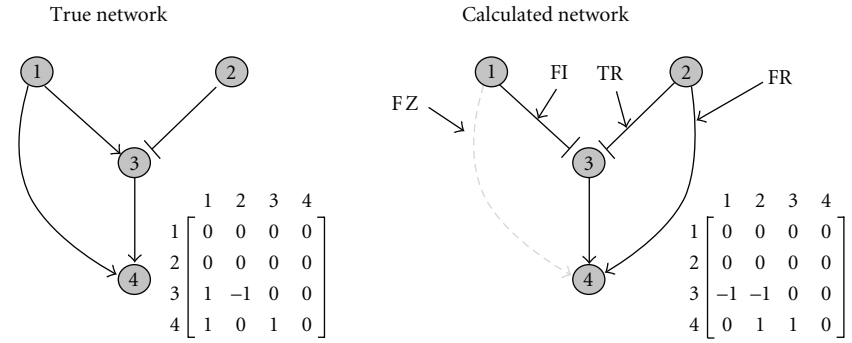
$$d(\text{sen}, \text{spe}) := \sqrt{(1 - \text{sen})^2 + (1 - \text{spe})^2}, \tag{7}$$

where sen is the sensitivity and spe the specificity. See Figure 1 for definitions. A distance of zero is optimal. We considered all 600 results for this optimization strategy. The sensitivity and specificity are the averaged values over all reconstruction results and are equally weighted, that is, the distance is a balance between calculated true regulations and true zeros (nonregulations) among all regulations and nonregulations, respectively, in the model. A lower threshold would result in more true regulations but with more false regulations and less true zeros, that is, the sensitivity is increased while the specificity is decreased. A higher value has the opposite effect.

## 5. Validation

For the validation, we calculated the sensitivity, specificity, and precision as defined in Figure 1. Sensitivity is the fraction of the number of found true regulations to all regulations in the model. Specificity defines the fraction of correctly found noninteractions to all noninteractions in the model. Since the number of noninteractions in the model is usually large compared to false regulations, the specificity is then around one and does not give much information about the quality of the method. Therefore, we calculated as well precision, which is the fraction of the number of correctly found regulations to all found regulations in the result.

The relevance network and graphical Gaussian approaches give no information about the direction of a link. Only undirected graphs can be revealed. Therefore, we used modified definitions for sensitivity $\text{sen}^{\text{U}}$, specificity




(a)

| True network \ Calculated network | 1 | 0 | −1 |
|---|---|---|---|
| 1 | TR | FZ | FI |
| 0 | FR | TZ | FR |
| −1 | FI | FZ | TR |

$$\text{sen:} = \frac{\text{TR}}{\text{TR} + \text{FZ} + \text{FI}} \tag{10}$$

$$\text{spe:} = \frac{\text{TZ}}{\text{TZ} + \text{FR}} \tag{11}$$

$$\text{pre:} = \frac{\text{TR}}{\text{TR} + \text{FR} + \text{FI}} \tag{12}$$

(b)

Figure 1: *Definitions*. (a) Example of a true (model) network and a calculated network. The adjacency matrix represents the network structure. (b) *Left:* gene regulatory models have three discrete states (1: activation, −1: inhibition, 0: nonregulation). We consider the kind of regulation (activation or inhibition) in the classification of the results according to the models: TR: True regulation; TZ: True zero; FR: False regulation; FZ: False zero; FI: False interaction. *Right:* definitions for sensitivity, (10), specificity, (11), and precision, (12).

$\text{spe}^{\text{U}}$, and precision $\text{pre}^{\text{U}}$ that consider a regulation from node $i$ to $j$ in the resulted network as true, if there is a link from node $i$ to $j$ or $j$ to $i$ in the model network, that is, the network is assumed as undirected.

Further we calculated a measure which considers an undirected graph and additionally does not count false interactions, that is, false identified activations or inhibitions. The corresponding networks are assumed as undirected with no interactions type, that is, these are undirected, binary graphs. Equations (10), (11), and (12) are reduced then to the usual definition of sensitivity and specificity, respectively. The modified measures are denoted with $\text{sen}^{\text{B}}$, $\text{spe}^{\text{B}}$, $\text{pre}^{\text{B}}$.

To obtain a single value measure for one result we calculated the combined measure defined in (7). This distance measure $d$ combines the sensitivity and specificity equally weighted to a single value measure. Low values indicate good reconstruction performances. Correspondingly to sensitivity and specificity, the undirected distance measures are indicated by $d^U$ and the binary, undirected measure by $\text{d}^{\text{B}}$.

Rather than selecting an arbitrary threshold for discretizing the resulting matrices it is convenient to use the curves of sensitivity versus specificity or precision versus recall for thresholds in interval $[0; 1]$ to assess the method performances. The measure recall is equal to the sensitivity. These curves are called receiver operator characteristics (ROCs). To obtain a single value measure one can use the area under the curve (AUC). We calculated AUC of the sensitivity versus specificity curves as an additional performance measure. Larger values indicate better performances. Note that a value less than 0.5 does not mean anticorrelation, since a random classifier is not represented by the diagonal.

## 6. Performance Results

We accomplished a systematic evaluation of the performances of six different reverse engineering applications using artificial gene expression data. In the program package ParCorA there are seven correlation measures implemented, including Pearson and Spearman correlation of different orders, which we all used. 600 datasets, with different numbers of genes, dataset sizes, and noise levels, were analyzed by each of the total twelve applications.

For all relevance network methods, graphical Gaussian model, and neural network we determined an optimized threshold for discretization of the results considering all datasets. The thresholds are listed in Table 2.

The averaged reconstruction performances over all datasets with regard to different validation measures are given in Table 3. Since some applications, such as relevance networks give no information about the direction of regulation, we calculated as well undirected measures, denoted with $U$. Additionally, we computed measures, which considers undirected results and neglects the kind of interaction information (activation or inhibition). These measures are indicated by $B$.

None of the reconstruction methods outperforms all other methods. Further, no method is capable of reconstructing the entire true network structure for all datasets. In particular sensitivity and precision are low for all methods. A low precision means that among the predicted regulations, there are only a few true regulations. In the study the precision is always lower than 0.3. This is due to the fact that several input datasets carry a high error level. For example,

Table 2: Discretization thresholds for different types of measures.

| Application | Threshold | Threshold$^U$ | Threshold$^B$ |
|---|---|---|---|
| GNRevealer (neural network) [NN] | 0.14 | 0.18 | 0.16 |
| GeneNet (graphical Gaussian model) [GGM] | — | 0.02 | 0.02 |
| Partial Pearson correlation, 0th order [PC0] | — | 0.15 | 0.10 |
| Partial Pearson correlation, 1st order [PC1] | — | 0.11 | 0.09 |
| Partial Pearson correlation, 2nd order [PC2] | — | 0.09 | 0.06 |
| Partial Pearson correlation, 3rd order [PC3] | — | 0.05 | 0.03 |
| Partial Spearman correlation, 0th order [SC0] | — | 0.10 | 0.10 |
| Partial Spearman correlation, 1st order [SC1] | — | 0.10 | 0.09 |
| Partial Spearman correlation, 2nd order [SC2] | — | 0.07 | 0.06 |

the input data includes time series with noise up to 50% (cv = 0.5). This can bias the performance results. On the other side, the dataset contains small scale time series (5 genes) with up to 50 repetitions and performances are much better with respect to these data (data not shown).

The neural network approach shows the best results among the algorithms tested with regard to the distance measures $d$ and AUC. On average it identifies over 27% of the directed regulations correctly, the highest value among all methods. This is remarkable considering the high error level inherent in several datasets. However, simultaneously the specificity is quite low. That indicates that many false regulations were identified. Less than 10% of the found regulations are true (precision). In contrast, the Bayesian network approaches, DBN and SSM, have a large specificity but with a very low sensitivity. Hence the performances are poor. Only a few regulations were identified and only some of them are true (low precision).

The relevance network approaches using partial Spearman rank correlation show better performances compared to partial Pearson correlation with regard to the distance measure and AUC. This might be explainable by the robustness of the Spearman correlation taking ranks into account rather than actual expression data which is advantageous inspite of noisy data. Surprisingly, with higher orders of partial Pearson and Spearman correlation the distance measures $d^B$ are not increasing. It is around 0.7 for Pearson and 0.68 for Spearman correlation. However, with in average up to 55% ($\text{sen}^B = 0.545$) of true undirected links could be identified by 1st-order Pearson correlation, neglecting the type of interactions. But 0th-order Spearman correlation identified over 55% (in average) of all nonregulations.

The MI method (ARACNE) found the fewest true undirected links (low sensitivity $\text{sen}^B$), except the DBN and SSM methods. In comparison to the relevance network approaches, MI has a considerably larger specificity $\text{spe}^B$, that is, MI identifies more nonregulations in the network correctly (true zeros). GGM shows the opposite behavior. It has a larger sensitivity but a lower specificity compared to MI.

In Figures 2 and 3 more details about the performance of each method are plotted with the error resulting from the averaging. The performance behavior with regard to different number of time series, that is, size of dataset, different noise levels, that is, coefficient of variation, and network size, that is, different number of nodes is shown. The distance measures over the number of time series were averaged over five different networks with four different sizes and six different noise levels, that is, in total of 120 datasets. In case of cv and network size, values were averaged over results from 100 and 150 datasets, respectively.

An overall trend is seen for increasing coefficient of variations. As expected the performances of each method decreases with increasing cv (middle column). Though, the distance measures for Banjo does change only slightly, it remains on a very large value. This indicates a poor reconstruction performance. SSM shows a similar behavior. The distance measure $d^B$ increases very fast for the graphical Gaussian model (GGM). However, the values of the measure decrease noticeable with the size of network. This is in contrast to all other methods, where for larger networks a decrease of reconstruction performance is observable. Surprisingly, the dataset size, that is, the number of time series does not have a large impact on all methods, except for SSM, where the distance measure decreases from a high value. However, in general, more available data would not always result in a better performance.

Among all partial Pearson correlations methods, the 2nd order outperforms the others. It has a slightly better performance measure under all conditions. This is similar to the 2nd order of partial Spearman correlation. It shows the best performance in all plots. Further, it is always below the best partial Pearson correlation.

## 7. Discussion and Conclusion

The comparative study shows that the performances of the reverse engineering methods tested here are still not good enough for practical applications with large networks. Sensitivity, specificity, and precision are always low. Some methods predict only few gene interactions, such as DBN, indicated by a low sensitivity and, in contrast to that, other methods identify many false regulations, such as the correlation measures. We tested different sets of data, including different sizes and noises to highlight the conditions for better performances of each method.

DBN performs poorly on all datasets. Under no condition of the study it shows an appropriate performance. The

Table 3: *Performance results*. Results of each application applied on all 600 datasets. DBN: Dynamic Bayesian network; NN: Neural network; MI: ARACNE; PC: Partial correlation with Pearson correlation of given order; SC: Partial correlation with Spearman correlation of given order; SSM: State space model; GGM: Graphical Gaussian model. Type is the performance measure type, that can be D for directed graph, U for undirected graph, B for binary and undirected graph. sen, spe, pre, and d are defined in (10), (11), (12), and (7), respectively. AUC is the area under the ROC curve. The averaged values are given with standard deviation in parenthesis. The top value of each type and column is highlighted in boldface.

| Name | Type | sen | spe | pre | $d$ | AUC |
|---|---|---|---|---|---|---|
| DBN | D | 0.030(0.084) | 0.953(0.117) | 0.041(0.119) | 0.971(0.067) | — |
| | U | 0.050(0.119) | 0.924(0.173) | 0.064(0.138) | 0.953(0.083) | — |
| | B | 0.084(0.196) | 0.924(0.173) | 0.099(0.193) | 0.919(0.116) | — |
| NN | D | **0.276(0.197)** | 0.660(0.216) | **0.091(0.073)** | **0.800(0.131)** | **0.324** |
| | U | **0.334(0.204)** | 0.617(0.278) | **0.208(0.162)** | **0.768(0.157)** | **0.350** |
| | B | 0.539(0.255) | 0.574(0.278) | **0.281(0.164)** | **0.628(0.147)** | **0.557** |
| SSM | D | 0.027(0.073) | **0.973(0.053)** | 0.052(0.139) | 0.973(0.068) | — |
| | U | 0.030(0.075) | **0.975(0.048)** | 0.094(0.224) | 0.970(0.071) | — |
| | B | 0.049(0.114) | **0.975(0.048)** | 0.153(0.297) | 0.951(0.110) | — |
| GGM | D | — | — | — | — | — |
| | U | 0.238(0.198) | 0.585(0.290) | 0.116(0.157) | 0.868(0.145) | 0.266 |
| | B | 0.442(0.326) | 0.585(0.290) | 0.225(0.212) | 0.695(0.185) | 0.526 |
| MI | D | — | — | — | — | — |
| | U | 0.196(0.129) | 0.745(0.146) | 0.163(0.124) | 0.843(0.130) | — |
| | B | 0.287(0.177) | 0.745(0.146) | 0.239(0.170) | 0.757(0.162) | — |
| PC0 | D | — | — | — | — | — |
| | U | 0.177(0.154) | 0.659(0.234) | 0.106(0.153) | 0.891(0.116) | 0.253 |
| | B | 0.513(0.228) | 0.492(0.223) | 0.220(0.174) | 0.703(0.132) | 0.506 |
| PC1 | D | — | — | — | — | — |
| | U | 0.228(0.161) | 0.541(0.226) | 0.105(0.131) | 0.898(0.126) | 0.249 |
| | B | **0.545(0.225)** | 0.461(0.214) | 0.219(0.168) | 0.705(0.135) | 0.502 |
| PC2 | D | — | — | — | — | — |
| | U | 0.186(0.136) | 0.635(0.154) | 0.108(0.125) | 0.892(0.125) | 0.249 |
| | B | 0.493(0.186) | 0.515(0.151) | 0.217(0.157) | 0.702(0.140) | 0.504 |
| PC3 | D | — | — | — | — | — |
| | U | 0.221(0.154) | 0.573(0.179) | 0.108(0.102) | 0.888(0.135) | 0.250 |
| | B | 0.526(0.223) | 0.484(0.190) | 0.217(0.142) | 0.701(0.130) | 0.506 |
| SC0 | D | — | — | — | — | — |
| | U | 0.285(0.195) | 0.555(0.217) | 0.129(0.135) | 0.842(0.141) | 0.298 |
| | B | 0.491(0.247) | 0.555(0.217) | 0.230(0.175) | 0.676(0.139) | 0.528 |
| SC1 | D | — | — | — | — | — |
| | U | 0.272(0.186) | 0.563(0.215) | 0.127(0.135) | 0.849(0.137) | 0.291 |
| | B | 0.518(0.233) | 0.518(0.209) | 0.231(0.173) | 0.682(0.135) | 0.521 |
| SC2 | D | — | — | — | — | — |
| | U | 0.256(0.165) | 0.588(0.143) | 0.125(0.117) | 0.851(0.137) | 0.290 |
| | B | 0.500(0.188) | 0.542(0.145) | 0.229(0.155) | 0.678(0.129) | 0.523 |

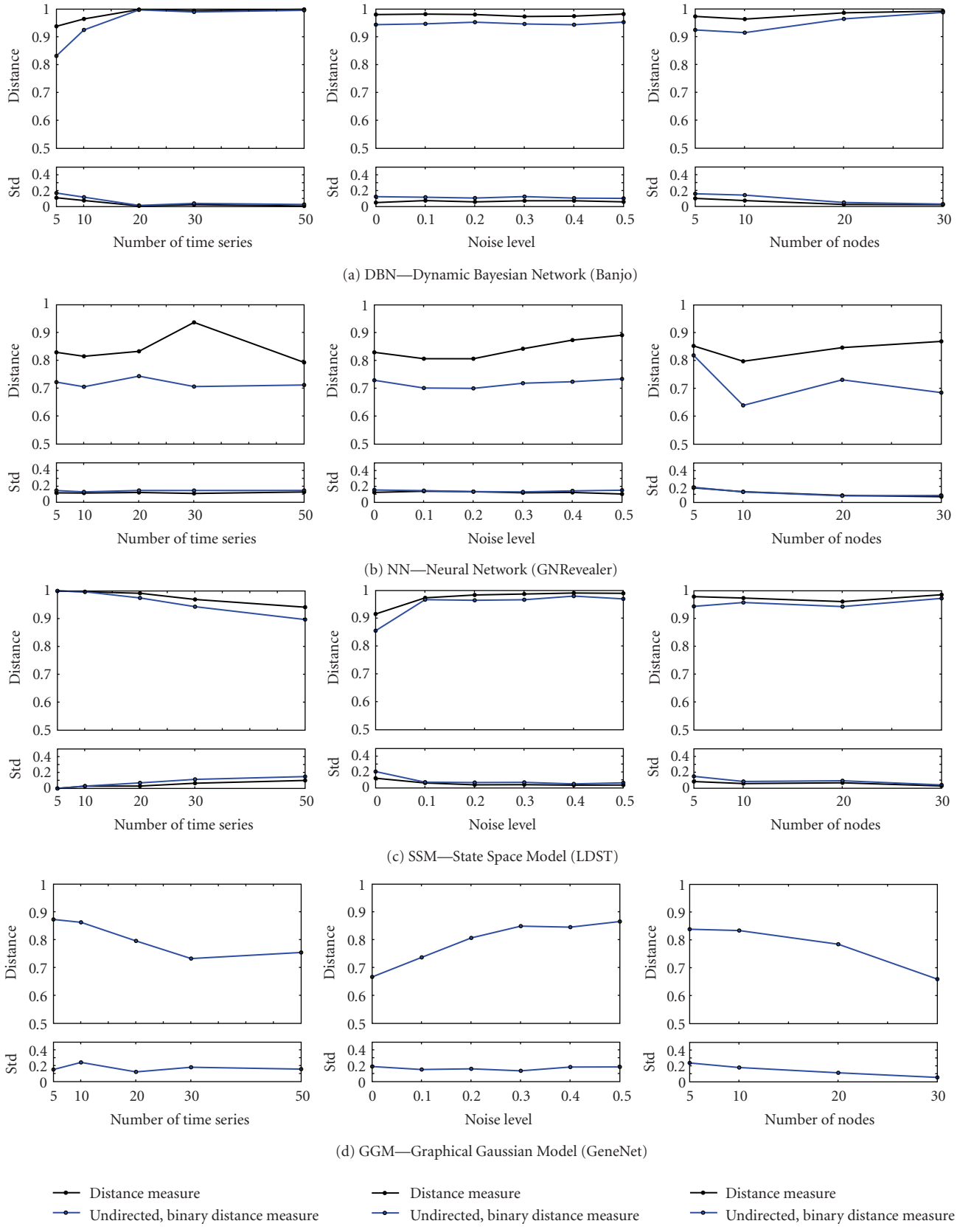



FIGURE 2: *Performances of applications*. The directed distance measure $d$ (black line) and undirected, binary distance measures $d^B$ (blue line) is plotted with standard deviations below. The measure $d$ is not available for all methods. From left to right in each row: distance measures over number of time series, cv, and network sizes. Each value is an average over all results with the given feature of the abscissa (see text for more details). A smaller distance indicates a better performance.

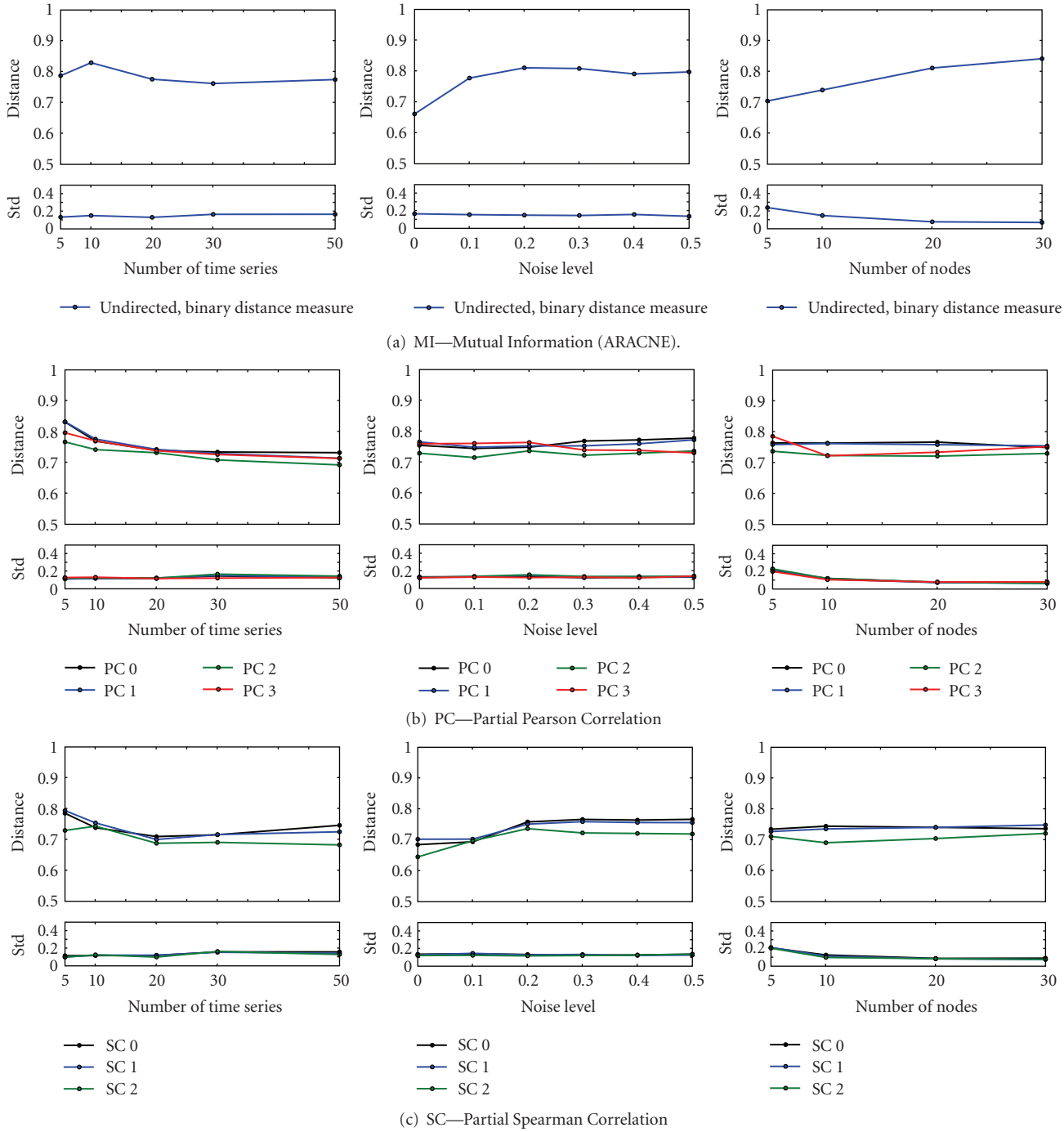



FIGURE 3: *Performance of applications*. See also Figure 2. Only the undirected, binary distance measure $d^B$ is plotted for partial Pearson and partial Spearman correlations. Colors indicate the different correlation measures.

specificity is always very large, but with a very low sensitivity. Only very few regulations were identified and the performance does not improve with larger datasets. It is known that Banjo requires large datasets for better performances [13]. This may be a reason for the observations. A similar behavior shows the other Bayesian network approach, the state space model. It is slightly better than DBN, but SSM has as well very low sensitivity. The predictive power of such a stochastic approach could not be shown under the conditions in this study.

The neural network approach shows the best results among all methods tested. It has a balance between true positives and true zeros. This is due to the appropriately chosen threshold for the postprocess discretization. Nevertheless, NN predicts many regulations and many of them are incorrect, that is, it has many false regulations. Even with a

large number of datasets, a complete reconstruction is not possible.

Schäfer and Strimmer [7], the authors of the GeneNet package including GGM, pointed out that their method is intended for analysis of small sample sizes. It requires a large number of genes in the dataset to estimate the null distribution from the data, which is used for detecting statistical significant interactions. This behavior is shown in Figure 2. Increasing number of genes in the dataset decrease the distance measure $d^B$ resulting in a better performance.

The assumption of statistical independence of each time point measurement is not satisfied, although ARACNE performs not all that bad. With larger datasets the performance increases, but decreases, as expected, with noisy data.

The Spearman correlation is a nonparametric measure for correlation. It does not make any assumption for the probability distribution of the data. In this study it outperforms the Pearson correlation, which can only detect linear relationships. It seems that the rank correlation is more appropriate for analyzing time series data because of its robustness against noisy data.

A crucial point in the determination of the distance measures is the chosen thresholds for discretization of the resulted matrices of continuous real values. An optimal threshold as shown in Table 2 was determined for the methods NN, GGM, all PCs, and all SCs with regard to an optimized distance measure. The other methods do not need a discretization, nice the methods provided a ternary matrix (DBN and SSM) or have already removed all nonsignificant links form the matrix (MI). A measure, independent of an artificial postprocessing of the results, is the area under the curve (AUC) score which we calculated as well. The advantage of using AUC is to obviate the need for choosing a discretization threshold. In Table 3 it is shown that a similar classification of method performances is obtained with regard to AUC and the distance measure.

Some aspects have not been addressed in this study and can be investigated further. It would be interesting to see the performances for larger networks sizes (more than 50 nodes). Some methods, such as GGM, should then show increased performances. Further, many applications are not suitable for analyzing such large datasets. For these methods a reduction of the dimension of the data has to be performed in order to obtain datasets of appropriate sizes. Different reduction methods can be investigated for that. Moreover, it would be interesting to see the impact of missing data on the reconstruction results, since in real experiments there are often not all genes included in the dataset.

It is shown that the reliable reconstruction of the whole GRN is anymore an ambitious intention and needs further progress. For that, the quality and quantity of gene expression measurements have to be improved as well as the performance of current or new algorithms. Benchmarks with realistic artificial data has to identify those methods which show the best results under different conditions.

## Acknowledgments

This work was funded by the Max Planck Society and by the European Commission within its FP6 Programme under the thematic area Life sciences, genomics and biotechnology for health with the EMBRACE Project (Contract no. LSHG-CT-2004-512092) and 7th Framework Programme with the Project APO-SYS (HEALTH-F4-2007-200767).

## References

[1] C. Wierling, R. Herwig, and H. Lehrach, "Resources, standards and tools for systems biology," *Briefings in Functional Genomics and Proteomics*, vol. 6, no. 3, pp. 240–251, 2007.

[2] G. Ruvkun, "Molecular biology. Glimpses of a tiny RNA world," *Science*, vol. 294, no. 5543, pp. 797–799, 2001.

[3] S. Liang, S. Fuhrman, and R. Somogyi, "Reveal, a general reverse engineering algorithm for inference of genetic network architectures," in *Proceedings of the Pacific Symposium on Biocomputing (PSB '98)*, pp. 18–29, Maui, Hawaii, USA, January 1998.

[4] P. D'haeseleer, X. Wen, S. Fuhrman, and R. Somogyi, "Linear modeling of mRNA expression levels during CNS development and injury," in *Proceedings of the 4th Pacific Symposium on Biocomputing (PSB '99)*, pp. 41–52, Big Island of Hawaii, Hawaii, USA, January 1999.

[5] T. Chen, H. L. He, and G. M. Church, "Modeling gene expression with differential equations," in *Proceedings of the 4th Pacific Symposium on Biocomputing (PSB '99)*, pp. 29–40, Big Island of Hawaii, Hawaii, USA, January 1999.

[6] K. Basso, A. A. Margolin, G. Stolovitzky, U. Klein, R. Dalla-Favera, and A. Califano, "Reverse engineering of regulatory networks in human B cells," *Nature Genetics*, vol. 37, no. 4, pp. 382–390, 2005.

[7] J. Schäfer and K. Strimmer, "An empirical Bayes approach to inferring large-scale gene association networks," *Bioinformatics*, vol. 21, no. 6, pp. 754–764, 2005.

[8] N. Friedman, M. Linial, I. Nachman, and D. Pe'er, "Using Bayesian networks to analyze expression data," *Journal of Computational Biology*, vol. 7, no. 3-4, pp. 601–620, 2000.

[9] H. Hache, C. Wierling, H. Lehrach, and R. Herwig, "Reconstruction and validation of gene regulatory networks with neural networks," in *Proceedings of the 2nd Foundations of Systems Biology in Engineering Conference (FOSBE '07)*, pp. 319–324, Stuttgart, Germany, September 2007.

[10] C. Rangel, J. Angus, Z. Ghahramani, et al., "Modeling T-cell activation using gene expression profiling and state-space models," *Bioinformatics*, vol. 20, no. 9, pp. 1361–1372, 2004.

[11] M. J. Beal, F. Falciani, Z. Ghahramani, C. Rangel, and D. L. Wild, "A Bayesian approach to reconstructing genetic regulatory networks with hidden factors," *Bioinformatics*, vol. 21, no. 3, pp. 349–356, 2005.

[12] N. Friedman, K. Murphy, and S. Russell, "Learning the structure of dynamic probabilistic networks," in *Proceedings of the 14th Annual Conference on Uncertainty in Artificial Intelligence (UAI '98)*, G. Cooper and S. Moral, Eds., pp. 139–147, Morgan Kaufmann, Madison, Wis, USA, July 1998.

[13] J. Yu, V. A. Smith, P. P. Wang, A. J. Hartemink, and E. D. Jarvis, "Advances to Bayesian network inference for generating causal networks from observational biological data," *Bioinformatics*, vol. 20, no. 18, pp. 3594–3603, 2004.

[14] A. V. Werhli, M. Grzegorczyk, and D. Husmeier, "Comparative evaluation of reverse engineering gene regulatory networks

with relevance networks, graphical Gaussian models and Bayesian networks," *Bioinformatics*, vol. 22, no. 20, pp. 2523–2531, 2006.

[15] E. P. van Someren, L. F. A. Wessels, E. Backer, and M. J. T. Reinders, "Genetic network modeling," *Pharmacogenomics*, vol. 3, no. 4, pp. 507–525, 2002.

[16] H. de Jong, "Modeling and simulation of genetic regulatory systems: a literature review," *Journal of Computational Biology*, vol. 9, no. 1, pp. 67–103, 2002.

[17] T. S. Gardner and J. J. Faith, "Reverse-engineering transcription control networks," *Physics of Life Reviews*, vol. 2, no. 1, pp. 65–88, 2005.

[18] A. de la Fuente, N. Bing, I. Hoeschele, and P. Mendes, "Discovery of meaningful associations in genomic data using partial correlation coefficients," *Bioinformatics*, vol. 20, no. 18, pp. 3565–3574, 2004.

[19] J. Quackenbush, "Computational analysis of microarray data," *Nature Reviews Genetics*, vol. 2, no. 6, pp. 418–427, 2001.

[20] P. Mendes, W. Sha, and K. Ye, "Artificial gene networks for objective comparison of analysis algorithms," *Bioinformatics*, vol. 19, supplement 2, pp. ii122–ii129, 2003.

[21] D. Repsilber and J. T. Kim, "Developing and testing methods for microarray data analysis using an artificial life framework," in *Proceedings of the 7th European Conference on Advances in Artificial Life (ECAL '03)*, vol. 2801 of *Lecture Notes in Computer Science*, pp. 686–695, Dortmund, Germany, September 2003.

[22] H. Hache, C. Wierling, H. Lehrach, and R. Herwig, "GeNGe: systematic generation of gene regulatory networks," *Bioinformatics*, vol. 25, no. 9, pp. 1205–1207, 2009.

[23] P. J. Werbos, "Backpropagation through time: what it does and how to do it," *Proceedings of the IEEE*, vol. 78, no. 10, pp. 1550–1560, 1990.

[24] D. Heckerman, D. Geiger, and D. M. Chickering, "Learning Bayesian networks: the combination of knowledge and statistical data," *Machine Learning*, vol. 20, no. 3, pp. 197–243, 1995.

[25] B. Bollobás, C. Borgs, J. Chayes, and O. Riordan, "Directed scale-free graphs," in *Proceedings of the 14th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA '03)*, pp. 132–139, Baltimore, Md, USA, November 2003.

[26] M. J. Schilstra and C. L. Nehaniv, "Bio-logic: gene expression and the laws of combinatorial logic," *Artificial Life*, vol. 14, no. 1, pp. 121–133, 2008.

[27] E. H. Davidson, J. P. Rast, P. Oliveri, et al., "A genomic regulatory network for development," *Science*, vol. 295, no. 5560, pp. 1669–1678, 2002.