BIOINFORMATICS

Vol. 20 no. 18 2004, pages 3594–3603
doi:10.1093/bioinformatics/bth448

Advances to Bayesian network inference for generating causal networks from observational biological data

Jing Yu[1,2], V. Anne Smith[1], Paul P. Wang[2], Alexander J. Hartemink[3,] and Erich D. Jarvis[1,*]*

[1]Department of Neurobiology, Duke University Medical Center, Box 3209, Durham, NC 27710, USA, [2]Department of Electrical Engineering and [3]Department of Computer Science, Duke University, Durham, NC 27708, USA

Received on March 4, 2004; revised on June 18, 2004; accepted on July 13, 2004
Advance Access publication July 29, 2004

ABSTRACT

Motivation: Network inference algorithms are powerful computational tools for identifying putative causal interactions among variables from observational data. Bayesian network inference algorithms hold particular promise in that they can capture linear, non-linear, combinatorial, stochastic and other types of relationships among variables across multiple levels of biological organization. However, challenges remain when applying these algorithms to limited quantities of experimental data collected from biological systems. Here, we use a simulation approach to make advances in our dynamic Bayesian network (DBN) inference algorithm, especially in the context of limited quantities of biological data.

Results: We test a range of scoring metrics and search heuristics to find an effective algorithm configuration for evaluating our methodological advances. We also identify sampling intervals and levels of data discretization that allow the best recovery of the simulated networks. We develop a novel *influence score* for DBNs that attempts to estimate both the sign (activation or repression) and relative magnitude of interactions among variables. When faced with limited quantities of observational data, combining our influence score with moderate data interpolation reduces a significant portion of false positive interactions in the recovered networks. Together, our advances allow DBN inference algorithms to be more effective in recovering biological networks from experimentally collected data.

Availability: Source code and simulated data are available upon request.

Contact: yu@ee.duke.edu; amink@cs.duke.edu; jarvis@neuro.duke.edu

Supplementary information: http://www.jarvislab.net/Bioinformatics/BNAdvances/

*To whom correspondence should be addressed.

1 INTRODUCTION

A variety of network inference algorithms have recently been used to identify gene regulatory networks from observational gene expression data (Akutsu *et al*., 2000; Arkin *et al*., 1997; D'haeseleer *et al*., 1999; Friedman *et al*., 2000; Gardner *et al*., 2003; Hartemink *et al*., 2001; Liang *et al*., 1998; Weaver *et al*., 1999; Xu *et al*., 2002). Bayesian network (BN) inference algorithms have shown particular promise (Hartemink *et al*., 2002; Husmeier, 2003; Smith *et al*., 2002, 2003), because unlike most other modeling frameworks, they can capture many types of relationships between variables. Owing to their probabilistic nature, BN algorithms are also capable of handling noisy data as found in biological experiments. They can effectively handle hundreds of variables (Friedman *et al*., 2000; Smith *et al*., 2002). While a static BN is restricted to be acyclic, a dynamic Bayesian network (DBN) can be used to infer cyclic phenomena such as feedback loops that are prevalent in biological systems. DBN algorithms can also infer direction of causality because they incorporate temporal information (Friedman *et al*., 1998; Smith *et al*., 2002, 2003).

However, there remain limitations in using discrete BN inference algorithms to analyze gene expression data. First, it is well known that inference algorithms perform better with larger quantities of data—and BN algorithms are no exception (Heckerman, 1996)—but in molecular biology the quantity of data that can be collected is often limited. Second, discrete BNs typically use combinatorial interaction models, making it difficult to determine the sign (+/−) and relative magnitude of interactions between variables. Although continuous BNs (Imoto *et al*., 2002) typically use additive interaction models, making it easy to deduce the sign and relative magnitude of interactions between variables, interactions between transcriptional regulators are known to often be combinatorial rather than additive.

Recently, we developed an approach to evaluate and advance the performance of inference algorithms

Bioinformatics vol. 20 issue 18 © Oxford University Press 2004; all rights reserved.

(Jarvis *et al*., 2002; Smith *et al*., 2002; also see Wessels *et al*., 2001; Zak *et al*., 2001). This approach involves applying network inference algorithms to data sampled from a biologically plausible simulator, and evaluating accuracy of the algorithms by comparing the recovered networks to the original networks used by the simulator. The simulator does not need to be an exact match to, or model all features of, a real transcriptional regulatory network, so long as it captures many of the important biological features. As with any model, the simulator is an approximation of a real system, making certain simplifying assumptions; these assumptions still allow for sufficient complexity in the simulator so as to exhibit the qualitative phenomena that are present in real systems. A simulation approach is necessary to assess the accuracy of our algorithms, given how little is known about real transcriptional regulatory systems. Our approach has been successfully used to evaluate algorithm performance on different network topologies and data sampling schemes (Husmeier, 2003; Smith *et al*., 2003).

Here, we apply this simulation framework to evaluate and improve the ability of DBN inference algorithms to recover networks when faced with limited data. We create an influence score for BN interactions that attempts to estimate regulatory signs and magnitudes in the recovered networks. We find that the influence score can also be used to prune away false positive links in the recovered networks. We determine that moderate data interpolation is successful at further reducing false positives when the data are limited. These advances work well on multiple datasets from a variety of different network topologies, including those with feedback loops and independent regulatory pathways, and thus offer promise for recovering meaningful models of biological systems from limited data.

2 METHODS

2.1 The simulator: GeneSim

GeneSim, written in Matlab, models genetic regulatory pathways of arbitrary network structure and produces values for gene expression levels at discrete time-steps. Values are produced by a combination of two processes. First, values at each time step are updated by a simple stochastic process:

$$Y_{t+1} - Y_t = f(Y_t) = A(Y_t - T) + \varepsilon,$$

where Y_t is a vector representing the expression levels of all genes at time t. Second, expression levels are restricted by a floor and ceiling function to range from 0 to 100 (arbitrary units). Expression levels are initialized to random values uniformly sampled from this range. The matrix A represents the regulatory interactions in the simulated network. The magnitude of each entry of A describes the strength of regulation that a regulator gene exerts upon a target gene; the sign indicates the type of regulation, with positive values indicating activation and negative values indicating repression. The vector T represents *constitutive expression values* for each gene; a regulator gene exerts an influence on its target gene only to the extent that it differs from its constitutive value (in this study, all constitutive values are set to 50, the median value between the maximum and minimum). If the regulator gene is present at a level above its constitutive value, then the regulatory effect on its target genes occurs as specified in A; the higher the regulator's level, the stronger the specified effect on its target genes. In contrast, if the regulator gene is present at a level below its constitutive value, then its effect is in the opposite direction of that specified in A; the lower the regulator's level, the stronger the opposite effect on its target genes. This latter property acts a constant rate (AT), which is intended to capture basic processes that are not explicitly modeled, such as mRNA degradation or release from repression, that act to return the target gene to its constitutive expression level. Our framework also allows for concentration-dependent degradation or autoregulatory effects to be modeled as elements along the diagonal in A; however, we did not do so here. The ε term models inherent biological noise and is drawn uniformly at random from the range -10 to 10. If a gene has no regulator (the corresponding row in A is all zeros), then it will move in a random walk, with steps taken according to the values of ε. This entire process of updates is bounded by the floor and ceiling values to prevent the concentration of an mRNA species from becoming negative or growing unbounded. The floor and ceiling values also introduce non-linearity near the boundaries of the expression range, resulting in expression value response curves that are approximately sigmoid, as is commonly found in concentration–response profiles of mRNA synthesis in real biological systems (Broccardo *et al*., 2004; Ferrari *et al*., 2004).

As the simulation runs, the data are sampled in pre-specified intervals, and the samples are exported to a text file. For example, if we collect data every five time-steps, the sampling interval is five and the sampled output is the series of expression level vectors ($Y_0, Y_5, Y_{10}, \ldots$), analogous to data gathered in a microarray time course experiment.

We use GeneSim to simulate 10 different randomly generated genetic regulatory networks. Each of the networks has 20 genes; 8–12 of these genes have regulatory interactions with at least one other gene; the remainder move in a random walk and thus serve as distracters for the inference algorithm (Fig. 1 and Supplemental Figure 1). For every link (also known as an edge or arc), we randomly assign the regulation strength to have one of four possible values, 0.05, 0.1, 0.15 or 0.2, with a randomly assigned sign ($+/-$). A total of 100 links are present across all 10 networks, 60 of which are one-parent links (they point to a gene only having one regulator), 34 are two-parent links, and six are three-parent links (Supplemental Table 1). For each of the experiments in Section 3, data from each of the 10 networks are sampled in 10 independent runs of the simulator, creating 10 independent datasets per network.

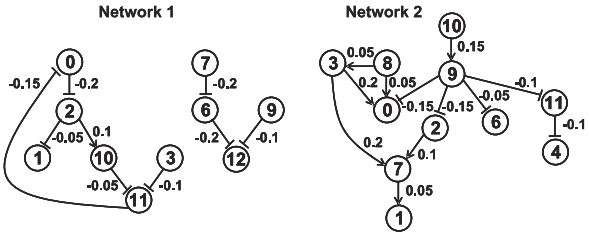

Fig. 1. Two examples from the set of ten networks simulated by GeneSim to produce data sets. In Network 1, there are two independent regulatory pathways, one of which includes a large feedback structure, while in Network 2, there are three parents for one gene. Numbers next to links specify regulation strengths; arrows: activation; flat heads: repression. All ten networks are shown in Supplemental Figure 1.

The DBN inference algorithm is applied separately to each of these datasets (10 networks, 10 datasets each) to obtain an average recovery performance for each experiment.

2.2 The network inference algorithm

Our DBN inference algorithm is written in C++ and is designed to search for high-scoring networks that describe probabilistic relationships between discrete variables. There are no additional assumptions added to the algorithm, such as linearity or non-linearity of the data. Every node in the DBN network represents a single variable, here one gene. Every directed link between two nodes represents a conditional statistical dependence of the child node on the parent node, here a regulatory relationship in which the parent gene regulates the child gene at a later time. We use a first-order Markov DBN where every variable at a given time point is influenced by itself and its parents in the immediate previous time point. Markov equivalence class ambiguity does not arise because DBNs use temporal information to unambiguously determine the direction of links between nodes. We compute a score for each network G, using a scoring metric that evaluates how probable it is that the network explains relationships in the observed data D. We use search heuristics to identify the network with highest score (top network).

Bayesian scoring metrics. We compare two scoring metrics: the BDe (Bayesian Dirichlet equivalence) and the BIC (Bayesian information criterion), as described by Heckerman (1996). Both scoring metrics incorporate a penalty for complexity to guard against over-fitting of data. The BDe score is based on the full Bayesian posterior probability $P(G|D)$ and has an inherent penalty for complexity since it computes the marginal likelihood $P(D|G)$ by integrating the probability of the data over all possible parameter assignments to G. The BIC score is an asymptotic approximation to the BDe score that uses an explicitly penalized estimate of the likelihood.

Search heuristics. We chose to compare three search methods that are distinct in their underlying principles: (1) greedy search with many random restarts, (2) simulated annealing and (3) a genetic algorithm. Because greedy search can easily become trapped in local optima, it has no theoretical convergence guarantees, but with many random restarts it has been observed to work well. Simulated annealing is a variant of Metropolis Hastings that has a convergence guarantee, but only under strict conditions. Genetic algorithms do not have a similar theoretical underpinning, but are inspired by an evolutionary perspective and have been observed in certain instances to find good solutions in large spaces. For greedy search, in each step we consider every possible local change and choose the one that improves the score the most; we use 100 random restarts to escape from local score maxima. Thus, the greedy search method we use does not have problems due to order of link selection. For simulated annealing and the genetic algorithm, we start with empty networks (i.e. no links).

The greedy search and simulated annealing algorithm frameworks are described by Heckerman (1996). We are not aware of a genetic algorithm being described for BN search, and thus explain the operations we chose to implement. A genetic algorithm (GA) (Goldberg, 1989) is a search heuristic that modifies a population of candidate networks using three operators: *mutation*, which produces an isolated change in one network, helping to escape local maxima; *crossover*, which swaps parts of two networks with one another, potentially combining well-scoring sub-networks; and *reproduction*, which promotes the best networks to the next generation. We mutate candidate networks by introducing a single change to a link in each network (addition, deletion or reversal). Crossover is performed by swapping sets of parents for each node between two networks. In particular, if we specify a network as the set of parents for each node $X_i(Pa(X_i))$, two candidate networks i and j can be denoted $\{Pa_i(X_1), \ldots, Pa_i(X_n)\}$ and $\{Pa_j(X_1), \ldots, Pa_j(X_n)\}$, respectively; then a randomly chosen variable X_k serves as a swap point, leading to two networks $\{Pa_i(X_1), \ldots, Pa_i(X_k), Pa_j(X_{k+1}), \ldots, Pa_j(X_n)\}$ and $\{Pa_j(X_1), \ldots, Pa_j(X_k), Pa_i(X_{k+1}), \ldots, Pa_i(X_n)\}$. In each iteration of the GA, either a mutation or a crossover operation is chosen at random. The newly created networks are reproduced in the next generation if they have higher scores than the current networks in the current generation, replacing the lowest scoring networks.

Influence score. We develop a novel *influence score* as part of our DBN algorithm in an attempt to predict the sign ($+$ or $-$) and relative magnitude of regulatory influences. This score is computed from the parameter estimates of the conditional probability values $\theta_{ijk} = P(x_i = k|pa(X_i) = j)$ for the top network. The parameter θ_{ijk} represents the probability that a node X_i is in state k when its parent set $Pa(X_i)$ is in state j. Although the values θ_{ijk} are posterior mean estimates of the conditional probabilities, the actual influence score is not; it is a summarization of these conditional probability estimates into a single number to approximate the sign and magnitude of the interactions between a child variable and each of its

parents. The conditional probability values are the same as those used to calculate the BDe and BIC scores. Intuitively, if there is a high probability for a child to be high when one parent is high and for the child to be low when that parent is low, then that parent is presumably an activator. Conversely, if there is a high probability for a child to be low when one parent is high and for the child to be high when that parent is low, then that parent is presumably a repressor. Motivated by this intuition, we apply a four-step process to compute the influence score for each link in the top network:

(1) We build a table of cumulative distribution function (CDF) values c_{ijk} from the values θ_{ijk} by summing over all lower states of k:

$$c_{ijk} = \sum_{k'=0}^{k} \theta_{ijk'},$$

where j and k represent an *ordered* set of states, and c_{ijk} represents the probability that child node X_i is in state k or lower when its parent set is in state j. Table 1 is an example of a CDF table for one node with two parents (denoted $P1$ and $P2$).

(2) If a parent is an activator, the CDF should shift in the positive direction (right) as the parent's value increases (when the parent is low, there is high probability for the child being low; when the parent is high, there is high probability for the child being high). Conversely, if a parent is a repressor, the CDF should shift in the negative direction (left) as the parent's value increases (when the parent is low, there is high probability for the child being high; when the parent is high, there is high probability for the child being low). Thus, to determine the sign of regulation from the c_{ijk} values, we formulate a voting system from shifts in the CDF tables. Three types of vote choices are made: positive, negative or neutral. If multiple parents are present, we consider each parent separately, with the other parents fixed in all possible instantiations. For example, in Table 1, we consider $P1$ with $P2$ fixed at value 0 (rows $j = 0$, 1 and 2), then $P2$ fixed at value 1 (rows $j = 3$, 4 and 5), and so forth. Then for each corresponding j and k, if all $c_{ijk} \leq c_{i(j-1)k}$ (CDF shift positive), then we consider parent $P1$ to possibly have a positive regulatory sign and increase the positive vote tally by one; however, if all $c_{ijk} \geq c_{i(j-1)k}$ (CDF shift negative), we increase the negative vote tally by one; otherwise we increase the neutral vote tally by one. We perform this voting for each parent with the others in all fixed instantiations. In the example of Table 1, there would be three votes per parent.

(3) We then designate the sign of regulation based on the total vote tally for each parent. If all votes are in the positive, or positive and neutral categories, we designate the sign of regulation to be positive (+); vice versa for negative sign (−). If votes exist in both the positive and negative categories, the sign is indeterminate and the influence score is set to 0.

(4) For links with a positive or negative sign after Step 3, we calculate the relative magnitude of the influence. The magnitude of the influence score is calculated from the difference between c_{ijk} and $c_{i(j-1)k}$ for all j: the larger the difference, presumably the stronger the magnitude of the influence. To avoid distortions from the number of categories used for discretization, we only consider the difference between $c_{ij'k}$ and $c_{ij''k}$, where j' and j'' are the states where the parent is at its lowest and highest values, respectively, with other parents at a fixed value. For instance, in the example of Table 1 when $P2$ is fixed at 0, for $P1$, $j' = 0$, $j'' = 2$, and $k = 0$, 1 and 2. If the link has a positive sign after Step 3, then each positive vote increases the magnitude by $(c_{ij'k} - c_{ij''k})$. If the link has a negative sign, then each negative vote decreases the magnitude by $(c_{ij'k} - c_{ij''k})$. The magnitude of the influence score is not changed by neutral votes. Finally, the magnitude is divided by the total number of votes to scale the resultant score into the range from −1 to 1. The more positive the influence score is, the stronger the activation; the more negative, the stronger the repression. When the influence score is near 0, it is difficult to infer the type of regulation.

Table 1. An example of a CDF table for a node i with two parents labeled $P1$ and $P2$

Parent configuration j	Parents $(P2, P1)$	CDFs $k = 0$	$k = 1$	$k = 2$
$j = 0$	(0, 0)	C_{i00}	C_{i01}	C_{i02}
$j = 1$	(0, 1)	C_{i10}	C_{i11}	C_{i12}
$j = 2$	(0, 2)	C_{i20}	C_{i21}	C_{i22}
$j = 3$	(1, 0)	C_{i30}	C_{i31}	C_{i32}
$j = 4$	(1, 1)	C_{i40}	C_{i41}	C_{i42}
$j = 5$	(1, 2)	C_{i50}	C_{i51}	C_{i52}
$j = 6$	(2, 0)	C_{i60}	C_{i61}	C_{i62}
$j = 7$	(2, 1)	C_{i70}	C_{i71}	C_{i72}
$j = 8$	(2, 2)	C_{i80}	C_{i81}	C_{i82}

Expression levels are discretized into three levels (0, 1 and 2). The c_{ijk} values in the table are used to calculate influence scores from $P1$ and $P2$ to node i.

2.3 Data collection and processing

Discretization. Before being passed to our DBN inference algorithm, the continuous data values we collect need to be discretized. In this study, we discretize the expression levels generated by GeneSim into different numbers of categories (with equal bin sizes) to determine if finer or coarser discretization improves recovery accuracy.

Sampling interval. We evaluate several different sampling intervals to determine the interval at which the inference algorithm recovers networks most accurately.

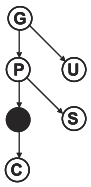

Fig. 2. Cartoon depiction of relative relationships of a target gene. The solid node is the target gene. P denotes a parent node, G denotes a grandparent, C denotes a child, U denotes an uncle and S denotes a sibling.

Quantity of data. We collect various numbers of sampled data points (25–5000) from GeneSim in order to analyze how different quantities of data affect the recovery performance. We compare the accuracy of our recovered networks across these different numbers of data points. The lowest quantities collected, 25–100 data points, represent biologically realistic quantities of data in the context of gene expression experiments.

Data interpolation. We test the effects of data interpolation on network recovery when only small quantities of sampled data are available. By interpolating the data, we are effectively adding an assumption to the data about the smoothness of the curves connecting any two sampled data points. While more complex strategies could be considered, we only apply linear interpolation here. For some experiments, we lengthen the sampling interval to allow room for interpolating points between samples.

2.4 Network recovery quantification

Classification of links. Based on their existence and non-existence in the recovered network and the true underlying network, we classified the recovered links into four categories: true positive (TP, a link that exists in both networks), true negative (TN, a link that does not exist in either network), false positive (FP, a link that exists only in the recovered network) and false negative (FN, a link that exists only in the true network). To evaluate the accuracy of a recovered network, we use two measures defined as follows: (1) recall = TP/(TP + FN), the percentage of links in the true network that also exist in the recovered network; and (2) imprecision = FP/(FP + TP), the percentage of links in the recovered network that do not exist in the true network.

Sub-classification of false positive links. Some false positives are more informative than others, in that they link genes that are not in a direct parent–child relationship but are still nearby in the pathway. Thus, we categorize false positives as being either from relatives (informative) or from strangers (uninformative). The most informative false positive links of a node are from its grandparents, since they are upstream in the pathway and only one step removed from the true parent; other relatives are siblings, uncles and children (Fig. 2); strangers consist of all nodes that are not relatives.

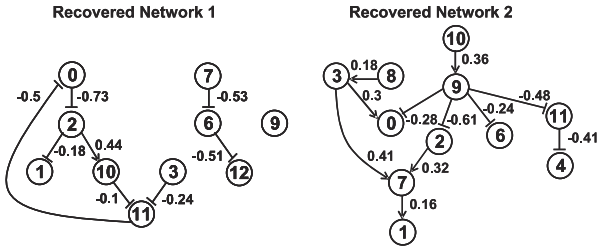

Fig. 3. Example of recovered networks for the underlying structures shown in Fig. 1 with 2000 data points and the BDe scoring metric. For network 1, one link (from node 9 to 12) is missing. For network 2, one link (from node 8 to 0) is missing. Numbers next to links specify computed influence scores, which correlate roughly with regulation strengths.

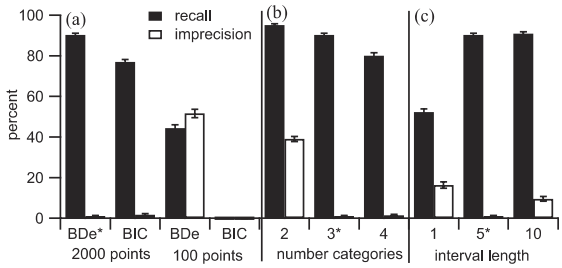

Fig. 4. Evaluating (**a**) the scoring metric, (**b**) discretization and (**c**) the sampling interval. Error bars, standard error of the mean (SEM). Asterisks, indicate the basic configuration (BDe scoring metric, 2000 data points, 3-category discretization and sampling interval of 5) that was varied along individual dimensions to produce the three panels (a), (b) and (c).

3 RESULTS

3.1 Advancing the DBN algorithm

Bayesian scoring metrics. To compare the two scoring metrics, for each of the 100 datasets from 10 networks described in Section 2.1, we use both 2000 and 100 data points sampled at an interval of five, a three-category discretization and greedy search with random restarts. With 2000 data points, we find that our DBN inference algorithm performs well, recovering most (85–100%) of the links in the networks (Figs. 3 and 4a). However the BDe scoring metric outperforms the BIC, having higher recall (Fig. 4a). With only 100 data points, BDe recovers some information while BIC finds no links at all (Fig. 4a). These results underscore the fact that the BIC scoring metric over-penalizes complexity relative to the BDe with small quantities of data (Heckerman, 1996). As we are especially interested in using microarray data in limited quantities to recover networks of molecular interactions, the BDe scoring metric is used in the remaining experiments.

Search heuristics. We find that all three search methods—greedy search with random restarts, simulated annealing and genetic algorithm—return nearly identical networks in our

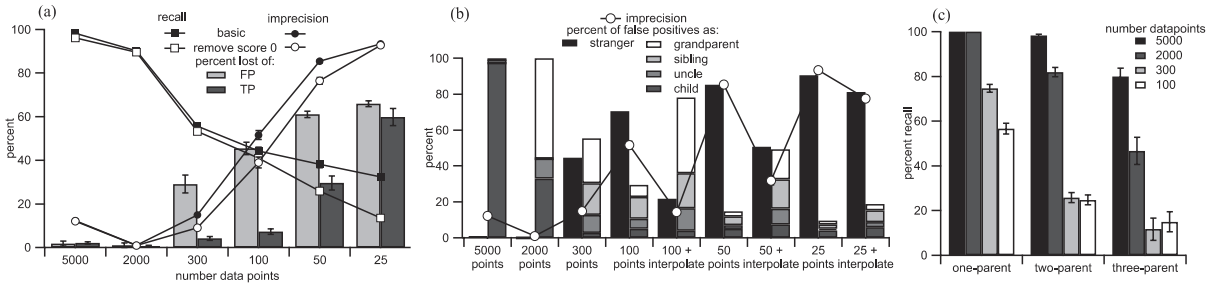

Fig. 5. Recovery results with different quantities of data. (**a**) Recall (squares) and imprecision (circles) for different quantities of data and effect of removing links with influence scores <0.001 (open symbols). Bar graph shows percentage lost of false positive (light grey) and true positive (dark grey) links from removing links with influence scores <0.001. (**b**) Percentage of false positive links from strangers (black bars on the left for each quantity of data) versus relatives (grandparents, siblings, uncles or children; stacked bars on the right for each quantity of data). Each set of bars sums to 100% of the false positives found for each quantity of data. Open circles represent the imprecision as in (a). The '+interpolate' indicates increased effective quantity of data due to interpolation, using a sampling interval of 15 and interpolation of two points in between. (**c**) Effect of quantity of data and the number of parents on the percentage recall. Error bars, SEM.

tests, each with high recall and low imprecision when allowed to run long enough (as seen in Fig. 3). However, each method requires a different amount of time to find the top network. Using a Dell PC with a 2.26 GHz CPU and 1 GB RAM, greedy search with random restarts is fastest (minutes), simulated annealing is second (tens of minutes) and the genetic algorithm is slowest (hours). Owing to the longer running times of simulated annealing and the genetic algorithm, these were tested on only three datasets. To save time, greedy search with random restarts is used in the remaining experiments.

3.2 Advancing data collection and processing

Discretization. We compare DBN inference algorithm performance when each of the 100 datasets of 2000 sampled data points is discretized into two, three or four categories. Only when the data are discretized into three categories does the algorithm find nearly the same regulatory network as the true network (Fig. 4b). For two-category discretization, the network has high imprecision (Fig. 4b). We believe this is attributable to loss of information due to the overly coarse discretization. For four-category discretization, the network has lower recall (Fig. 4b), due to difficulty in recovering links to nodes with multiple parents (data not shown). We believe this occurs because the finer discretization spreads the data across larger numbers of states for conditional probability values, making it more difficult to find dependencies between nodes (especially with multiple parents). Three-category discretization is thus used in the remaining experiments.

Sampling interval. Previously, Smith *et al*. (2003), using another simulated system and different network topologies, showed that there is an optimal range of sampling intervals for DBN inference algorithms. As a result, we seek to locate a sampling interval in the optimal range with our simulated system and network topologies. Using 100 datasets of 2000 data points sampled at a variety of different intervals, we find that the DBN recovers a more accurate network when the simulation is sampled at an interval of five (Fig. 4c); sampling at an interval of one yields lower recall and higher imprecision (Fig. 4c); sampling at an interval of 10 yields higher imprecision (Fig. 4c). We conclude that the sampling interval of five is within the optimal range.

Quantity of data. Since the quantity of data is a critical factor in the DBN network recovery, we perform a systematic evaluation of recovery results with 10 datasets each consisting of 25–5000 data points from each of the 10 simulated networks, using a sampling interval of five. Across all networks, our DBN inference algorithm works relatively well in recovering the true network with large quantities of data (2000 and 5000 data points; Fig. 5a, solid squares and circles). As the size of the dataset decreases (300 and less) the inference algorithm accuracy also decreases—recall drops and imprecision increases (Fig. 5a, solid squares and circles), with the relative percentages between the two crossing at 100 data points. Interestingly, when the dataset contains 5000 points, although the recall is very high (98.3%), the imprecision (12.2%) is also relatively high, compared with 2000 data points (Fig. 5a). To better understand the nature of these false positive links, we classify false positives as either strangers or one of four types of relatives (Fig. 2). Almost all of the false positives in the 5000 data point case are from child nodes back to their parents (Fig. 5b) to form loops. As the quantity of data decreases, the proportion from strangers (uninformative) in the false positive links increases, while the proportion from relatives (informative) decreases, especially from grandparents. We additionally evaluate DBN recall performance according to the number of parents of the child node in the true networks. There is lower recall of links to children with more than one parent; this effect is more pronounced with smaller quantities of data (Fig. 5c).

3.3 Main advances

Influence score. The influence score we compute accurately reflects the sign of regulation in the simulated networks (compare Figs 1 and 3). Not surprisingly, the magnitudes are

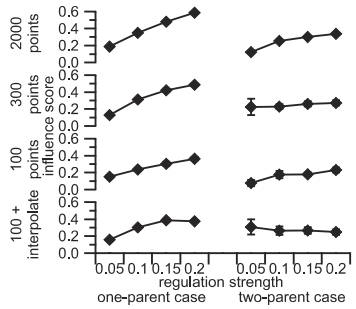

Fig. 6. Influence score. Average influence scores are shown for links in one-parent and two-parent cases. Influence scores in the recovered networks correlate well with the regulation strengths of the true network as there is an increasing linear relationship between the two. The '100+interpolate' analysis uses a sampling interval of 15 with interpolation of two points in between. Error bars, SEM.

different from the magnitudes of the regulatory strength in matrix *A* of GeneSim. To test whether the magnitude of the influence score is correlated with the underlying strength of regulation, we group the true positive links according to regulatory strengths across the 10 simulated network topologies. With large quantities of data (2000), the relative magnitudes of the influence scores in the recovered networks match the relative regulatory strengths in the true networks for both the one- and two-parent cases (Fig. 6; too few three-parent cases exist for proper analysis). However, as the quantity of sampled data is decreased, there is a decrease in influence score representation, particularly in the two-parent case. Specifically, with multiple parents and little data (300 data points or fewer), the influence score magnitude of one parent is partially obscured by the magnitude of the other parent. Regardless of the quantity of data, when the influence score indicates a sign for the interaction ($+/-$; Fig. 3), it is correct 100% of the time.

An influence score of 0 or very close to 0 means either that the sign of regulation is difficult to determine, or the regulation strength is very weak. We hypothesize that this information might be useful to eliminate false positive links. To test this hypothesis, we eliminate all links with an influence score whose magnitude is below 0.001 and find that this preferentially eliminates false positive links as opposed to true positive links for datasets with 300, 100 and 50 data points; with 25 data points, this preference is not as great (Fig. 5a, open squares and circles, and bar charts).

Data interpolation. We test whether using moderate data interpolation improves the accuracy of recovered networks in the context of small quantities of data. We find that simple linear interpolation has little effect on recall, but dramatically reduces imprecision when using only 100 (Fig. 7a) or 50 (Fig. 7b) data points.

Imprecision decreases most when a single data point is interpolated between each pair of real data points. When two data points are interpolated and the original sampling interval is 15 or 20, imprecision continues to decrease, but not as dramatically. However, as more interpolated data are added, the imprecision changes little, suggesting that the benefit of interpolation may plateau, as one might expect. Interpolation with only 25 original data points results in small comparable decreases for both imprecision and recall (Fig. 7c). With these smaller quantities of data (25–100), we note that the optimal sampling interval without interpolation, which results in higher recall, shifts from five to more spaced sampling (Fig. 7, solid symbols).

To determine if interpolation affects the influence score, we sample 100 data points at an interval of 15, with two data points interpolated between each. We find that the overall performance is similar to that without interpolation, except that interpolation obscures the relationship between the regulatory strength and the magnitude of the influence score for links with high regulatory strengths (Fig. 6). This is probably due to the fact that nodes receiving the largest regulatory strengths in the simulation tend to reach their maximal or minimal values more quickly than others, perhaps within the interval between sampled data. This would occur in a biological system when the transcription rates for different genes are such that they reach their corresponding maximal levels of expression at different times.

To better understand the effects of interpolation on reducing imprecision, we examine the identity of false positives in comparison to non-interpolated data. Using 100, 50 and 25 data points, we find that interpolation both reduces the proportion of strangers and increases the proportion of grandparents, most dramatically in the case of 100 data points (Fig. 5b). Thus, the reduction in imprecision due to interpolation is preferentially from strangers, as desired.

4 DISCUSSION

In this study, we evaluate and improve DBN inference algorithms for recovering networks from simulated biological systems. We find that the best configuration for evaluating our DBN inference algorithm is a greedy search method with random restarts employing the BDe scoring metric. The greedy search method works faster because it only allows changes that improve the score. This might be useful when the number of nodes in the network is large or processing time is critical. However, with more complex networks, the search may become more difficult if the surface of the landscape induced by the scoring metric is jagged and marked by large numbers of local maxima. In such cases, the asymptotic correctness of simulated annealing might overcome the speed advantage of greedy search (see Hartemink *et al*., 2002). We believe that the poor performance of the BIC scoring metric in the presence of small quantities of sampled data is due to the fact that its penalty for model complexity is accurate only asymptotically and is overly stringent relative to the BDe with finite quantities of data.

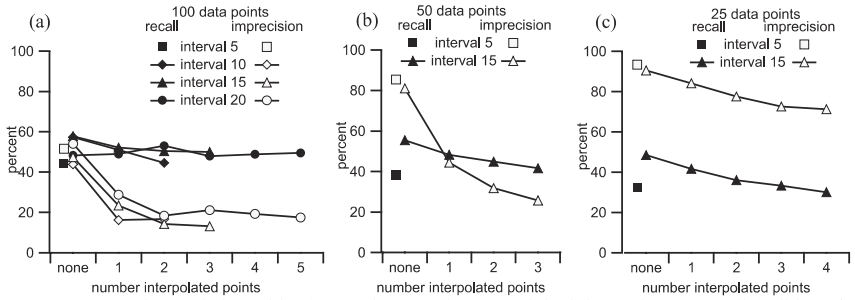

Fig. 7. Effect of interpolation with (**a**) 100, (**b**) 50 and (**c**) 25 original data points on recall (solid symbols) and imprecision (open symbols). Values for a sampling interval of five, which did not undergo interpolation, are offset to the left for clarity. Original sampling intervals are listed in the key at the top of each graph; SEM error bars are not shown, as all were smaller than the symbols being plotted.

We find the level of data discretization to be critical for network inference. In our simulated networks, three categories seem to best balance the tradeoff between information loss when too few categories are used and insufficient data for estimation when too many categories are used. We also find, as in Smith *et al*. (2003), that there is an optimal interval for sampling data from the system. However, we further find that the optimal interval varies with the number of data points sampled. Perhaps with very small datasets, the total time covered by sampling is small, and may not capture the flow of interaction through the full length of a pathway. Thus, small numbers of data points may require larger intervals to increase total coverage. Optimal sampling intervals are presumably determined by the internal dynamics of a system; to recover the most accurate network, educated assumptions about the system's dynamics need to be made.

The most critical advances we make are: (1) developing an influence score to recover more meaningful, more interpretable and more accurate networks; and (2) demonstrating the value of moderate data interpolation to assist in the recovery of more accurate networks from only small quantities of data, quantities that are biologically reasonable in the context of gene expression experiments. Our influence score performs well at predicting both the signs of interactions between nodes and the relative magnitudes of the regulatory strengths. Activation and repression interactions are known to be important in biological systems and we are not aware of previous methods for identifying these in a discrete Bayesian network.

Our results suggest that imprecision is one of the most significant problems for biological applications of DBN inference algorithms. We find that this concern is somewhat mitigated by the fact that many false positives are informative in that they link a node to a relative, if not the correct parent. Nevertheless, being able to reduce the overall number of false positives and shift false positives from strangers to relatives remains a critical goal in improving the effectiveness of network inference algorithms. Promisingly, the influence score we develop is useful in pruning away low-scoring false positive links. Furthermore, moderate data interpolation with good time coverage further reduces the overall number of false positive links. Interpolation also improves accuracy by shifting false positive links from strangers to relatives such as grandparents, which at least places genes only one gene removed from their direct regulator. With 25 time points, interpolation leads to small improvements, presumably because full coverage of the pathway is not obtained with so little data. In summary, we believe that interpolation will be a critical component for applying DBN inference algorithms to gene expression data, where time course measurements are often limited to less than 100 time points, since an animal usually has to be sacrificed at each time point. We note that interpolation does not eliminate the benefits of collecting larger quantities of real data; for any particular quantity of real data, extensive interpolation seems to provide no benefit over only moderate interpolation.

Bayesian networks can also be used to analyze continuous data directly, either with linear or non-linear regression (as in Imoto *et al*., 2002). Although it might seem that a continuous network would be preferred to one that discretizes data, discrete networks have their own advantages. In particular, continuous models typically assume additive influence of multiple parent variables on a child, whereas our discrete models can capture the types of combinatorial relationships among multiple parents that are commonly observed in gene regulation. Moreover, discrete networks need not adopt an assumption of Gaussian noise and are computationally less intensive to learn. Thus the choice of using a continuous or discrete network is largely a question of whether the assumptions each makes are suitable for the domain to which they are being applied.

The question of which search heuristic would be best suited to a specific BN search problem is an open one. An exact solution to the problem can be formulated using a dynamic programming algorithm (Ott *et al*., 2004), but since this algorithm remains exponential in its running time, it is only practical for small networks of 20–30 genes. Such an algorithm might be usefully exploited in conjunction with one of the heuristics tested here to produce more sophisticated heuristics scaling to large networks with hundreds to thousands of genes.

Our results are useful for designing experiments for later analysis with DBN inference algorithms in that they suggest minimum quantities of data that must be collected and provide

guidance in choosing good sampling intervals in order to recover networks with a desirable level of accuracy. We recommend that when designing experiments for use with a DBN inference algorithm, at least 50–100 real data points be collected at a sampling interval that matches the expected dynamics of the system and moderate interpolation (one to three data points interpolated between every real point) be performed.

Work remains to be done. Although we reduced false positives when using limited quantities of data, true positives did not increase. This is especially the case when a node had more than one parent (multiple regulators), which is a serious problem for genetic pathway recovery, as combinatorial regulatory control is a basic property of genetic pathways. Unfortunately, this is a limitation of statistical inference methods: the more complex the relationship to be learned, the more data will be required to learn it accurately. Since gene expression data are likely to remain in short supply, it is prudent to not attempt to recover complex pathways from gene expression data alone. Multiple types of data, such as transcriptional binding sites and protein levels, can significantly enhance the ability to accurately recover regulatory network structures when used in conjunction with expression data (Hartemink *et al*., 2002).

We also find that with large quantities of data (5000 data points) the top networks have incorrect links, particularly from child nodes to parent nodes, that are not there with smaller quantities of data (2000 data points). This seems to contradict the fact that BN inference algorithms are statistically consistent and, thus, known to perform better as more data becomes available (Heckerman, 1996). However, the consistency of BN inference is based on the assumption that the underlying system generating the data is a Bayesian network. This is presumably not true for real biological systems and is not true for our simulated system. Thus, due to this mismatch between the BN and the system being studied, more data may actually result in over-fitting the network to the data, producing false links between closely related variables. In fact, the BDe score for the true structures without these spurious links was lower than the false structures with them. In our experiments, the influence scores for these spurious links (mean $\pm$ SE: 0.086 ± 0.05) are much smaller than those for links with correct orientations (0.427 ± 0.008), and thus the influence score should be useful in selecting against these spurious links.

The performance of our inference algorithms depends in part on how the underlying networks are simulated; some meaningful biological information is always lost in mathematical modeling. However, while we do not claim that our simulation exactly matches the corresponding biology, we believe the advances we tested on this simulation are likely to lead to similar advances when applied to real biological data.

Our results suggest caution in the interpretation of networks, even those with a relatively small number of genes, reconstructed from what are today still considerable quantities of expression data (between 50 and 100 data points). However, we must keep in mind that when DBN inference algorithms are applied to real data, the results are not intended to be 100% correct or serve as a substitute for gene intervention experiments (e.g. gene manipulation). Rather, the networks found by DBN algorithms provide a rough but extremely useful sketch of the underlying biological pathways, generating hypotheses to be tested and offering significant guidance for future manipulation experiments.

ACKNOWLEDGEMENTS

We thank Kurt Grandis and Derek Scott for assistance in the beginning stages of this project, and Kazuhiro Wada, Tom Smulders and other members of the Jarvis laboratory for critical comments. This research was supported in part by an NSF CAREER award to A.J.H., and Whitehall and Packard Foundation grants to E.D.J.

REFERENCES

Akutsu,T., Miyano,S. and Kuhara,S. (2000) Inferring qualitative relations in genetic networks and metabolic pathways. *Bioinformatics*, **16**, 727–734.

Arkin,A., Shen,P. and Ross,J. (1997) A test case of correlation metric construction of a reaction pathway from measurements. *Science*, **277**, 1275–1279.

Broccardo,C.J., Billings,R.E., Chubb,L.S., Andersen,M.E. and Hanneman,W.H. (2004) Single cell analysis of switch-like induction of CYP1A1 in liver cell lines. *Toxicol. Sci.*, **78**, 287–294.

D'haeseleer,P., Wen,X., Fuhrman,S. and Somogyi,R. (1999) Linear modeling of mRNA expression levels during CNS development and injury. *Pac. Symp. Biocomput.*, **4**, 41–52.

Ferrari,M., Cosentino,M., Marino,F., Bombelli,R., Rasini, E., Lecchini,S. and Frigo,G. (2004) Dopaminergic D1-like receptor-dependent inhibition of tyrosine hydroxylase mRNA expression and catecholamine production in human lymphocytes. *Biochem. Pharmacol.*, **67**, 865–873.

Friedman,N., Murphy,K. and Russell,S. (1998) Learning the structure of dynamic probabilistic networks. *Proceedings of the 14th Conference on Uncertainty in Artificial Intelligence*, pp. 139–147.

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

Gardner, T.S., di Bernardo,D., Lorenz,D. and Collins,J.J. (2003) Inferring genetic networks and identifying compound mode of action via expression profiling. *Science*, **301**, 102–105.

Goldberg,D.E. (1989) Genetic Algorithms in Search, Optimization, and Machine Learning. Wesley, MA.

Hartemink,A.J., Gifford,D., Jaakkola,T. and Young,R. (2001) Using graphical models and genomic expression data to statistically validate models of genetic regulatory networks. *Pac. Symp. Biocomput.*, **6**, 422–433.

Hartemink,A.J., Gifford,D., Jaakkola,T. and Young,R. (2002) Combining location and expression data for principled discovery of genetic regulatory network models. *Pac. Symp. Biocomput.*, **7**, 437–449.

Heckerman,D. (1996) A tutorial on learning with Bayesian networks. *Technical Report MSR-TR-95–06*, Microsoft Research (March 1995; revised November 1996).

Husmeier,D. (2003) Sensitivity and specificity of inferring genetic regulatory interactions from microarray experiments with dynamic Bayesian networks. *Bioinformatics*, **19**, 2271–2282.

Imoto,S., Goto,T. and Miyano,S. (2002) Estimation of genetic networks and functional structures between genes by using Bayesian network and nonparametric regression. *Pac. Symp. Biocomput.*, **7**, 175–186.

Jarvis,E.D., Smith,V.A., Wada,K., Rivas,M.V., McElroy,M., Smulders,T.V., Carnici,P., Hayashisaki,Y., Dietrich,F., Wu,X. *et al*. (2002) A framework for integrating the songbird brain. *J. Comp. Physiol. A*, **188**, 961–980.

Liang,S., Fuhrman,S. and Somogyi,R. (1998) REVEAL, a general reverse engineering algorithm for inference of genetic network architectures. *Pac. Symp. Biocomput.*, **3**, 18–29.

Ott,S., Imoto,S. and Miyano,S. (2004) Finding optimal models for small gene networks. *Pac. Symp. Biocomput.*, **9**, 557–567.

Smith,V.A., Jarvis,E.D. and Hartemink,A.J. (2002) Evaluating functional network inference using simulations of complex biological systems. *Bioinformatics*, **18**, S216–S224.

Smith,V.A., Jarvis,E.D. and Hartemink,A.J. (2003) Influence of network topology and data collection on functional network influence. *Pac. Symp. Biocomput.*, **8**, 164–175.

Weaver,D.C., Workman,C.T. and Stormo,G.D. (1999) Modeling regulatory networks with weight matrices. *Pac. Symp. Biocomput.*, **4**, 112–123.

Wessels,L.F.A., van Someren,E.P. and Reinders,M.J.T. (2001) A comparison of genetic network models. *Pac. Symp. Biocomput.*, **6**, 508–519.

Xu,H., Wu,P., Wu,C., Tidwell,C. and Wang,Y. (2002) A smooth response surface algorithm for constructing a gene regulatory network. *Physiol. Genomics*. **11**, 11–20.

Zak,E.D., Doyle,F.J., III, Gonye,G.E. and Schwaber,J.S. (2001) Simulation studies for the identification of genetic networks from cDNA array and regulatory activity data. *Proc. Int. Conf. Intell. Comput. Symp. Biol.*, **2**, 231–238.