Journal of Machine Learning Research 7 (2006) 167–189 Submitted 3/04; Revised 9/05; Published 2/06

MinReg: A Scalable Algorithm for Learning Parsimonious Regulatory Networks in Yeast and Mammals

Dana Pe'er DPEER@GENETICS.MED.HARVARD.EDU
Genetics Department
Harvard Medical School
Boston, MA 02115, USA

Amos Tanay AMOS@POST.TAU.AC.IL
Computer Science Department
Tel Aviv University
Tel Aviv, Israel

Aviv Regev AREGEV@CGR.HARVARD.EDU
Bauer Center for Genomics Research
Harvard University
Cambridge, MA 02138, USA

Editor: Tommi Jaakkola

Abstract

In recent years, there has been a growing interest in applying Bayesian networks and their extensions to reconstruct *regulatory networks* from gene expression data. Since the gene expression domain involves a large number of variables and a limited number of samples, it poses both computational and statistical challenges to Bayesian network learning algorithms. Here we define a constrained family of Bayesian network structures suitable for this domain and devise an efficient search algorithm that utilizes these structural constraints to find high scoring networks from data. Interestingly, under reasonable assumptions on the underlying probability distribution, we can provide performance guarantees on our algorithm. Evaluation on real data from yeast and mouse, demonstrates that our method cannot only reconstruct a high quality model of the yeast regulatory network, but is also the first method to scale to the complexity of mammalian networks and successfully reconstructs a reasonable model over thousands of variables.

Keywords: Bayesian networks, structure learning, gene networks, gene expression, approximation algorithms

1. Introduction

Learning Bayesian network structure from data (Cooper and Herskovits, 1992; Heckerman et al., 1994) and its application to reconstruct *gene regulatory networks* from biological data (Friedman et al., 2000; Pe'er et al., 2001; Hartemink et al., 2002; Ong et al., 2002; Imoto et al., 2002; Yoo et al., 2002) is a subject of current research.

Regulatory networks control the expression of thousands of genes in a living cell, modulating the expression levels of individual genes based on external and internal conditions. To regulate

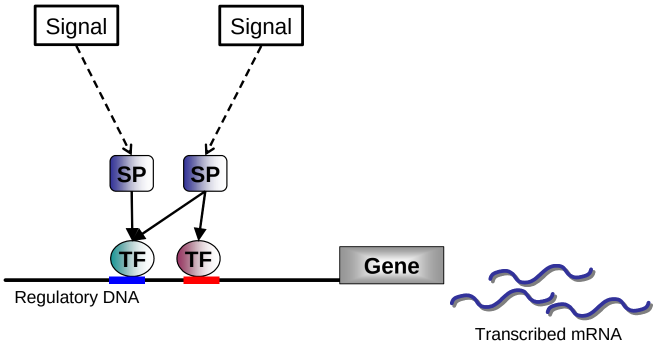


Figure 1: Biological regulation: Signals activate signaling molecules (SM), which in turn activate transcription factors (TF). When activated, these bind to DNA regulatory sequences. Combinations of such binding events control the levels of mRNA transcription in a combinatorial manner.

the expression of a gene, specialized proteins called *transcription factors (TFs)* bind to regulatory sequences on the DNA of *target genes* and work in a combinatorial fashion to ensure the correct amount is being transcribed (Figure 1). The behavior of those transcription factors is in turn controlled by the cell's environment through the action of *signaling proteins (SPs)*. The combined network of transcription factors and signaling proteins forms a regulatory program controlling the expression of individual genes directly (by regulator TFs) and indirectly (by regulator SPs). Since these networks serve as the information processing devices of cells, it is of great interest to uncover their structure and the regulation functions that they encode.

How can we learn such regulatory programs? An experimental technique, called *DNA microarrays* allows us to simultaneously measure the expression of thousands of genes under various conditions and perturbations, providing biologists with global observations of the workings of the cell. Importantly, microarrays measure not only the expression levels of target genes, but also of genes encoding regulators - TFs and SPs. As has been previously demonstrated (Pe'er et al., 2001, 2002; Segal et al., 2003), in many cases a TF's expression level is a good proxy to its activity, allowing us to construct a network that relates the gene expression of a target gene to the gene expression of its regulators. However, there are also numerous cases where a TF's activity is not determined by its expression level, but rather by other types of biochemical events, that that are unobserved in microarray data. Fortunately, in some of these cases, a change in the expression of indirect regulators (such as SPs that control the TF's activity) may be observed in microarray measurements, allowing us to detect an indirect regulatory relation in lieu of the direct event.

Following these biological considerations, it is expected that regulatory interactions between the genes would often result in corresponding statistical dependencies between random variables representing their expression. Thus, a Bayesian network approach to regulatory network reconstruction treats the expression level of each gene as a random variable and attempts to estimate the structural features of the dependencies in their joint probability distribution from data. Bayesian networks are particularly well suited for this domain, as has been demonstrated by early studies (Friedman et al., 2000; Pe'er et al., 2001; Hartemink et al., 2002). First, experimental evidence indicates that the regulatory network is sparse, such that only a few genes directly control the transcription of a given target (Martinez-Antonio and Collado-Vides, 2003; Shen-Orr et al., 2002; Lee et al., 2002). Second, microarray measurements are typically noisy, necessitating a probabilistic model. Finally, biological networks contains many important hidden variables (*e.g.* the actual activity level of the regulators) which can be handled well in a Bayesian networks framework.

Nevertheless, the biological domain raises several important challenges for learning Bayesian networks. The central difficulty is that contrary to previous applications, microarrays measure thousands of variables (genes) across at most a few hundred samples. Thus, even if a search for the optimal solution (over a prohibitively large space) was possible, statistical noise is likely to lead to spurious dependencies, resulting in models that significantly overfit the data.

This problem becomes even more pronounced when considering the complex regulatory networks of mammals. While Bayesian network approaches have been relatively successful in tackling networks of a unicellular model organism, the Baker's yeast *Saccharomyces cerevisiae*, they have yet to achieve similar success in mammalian systems, such as human or mouse cells. These organisms have considerably more complex regulatory systems, with a larger number of regulators and target genes, and much more complex combinatorial regulatory functions. Deciphering these networks can have significant implications to the understanding of animal development and common diseases. A central question toward these important applications is finding a parsimonious set of *major* regulators at the center of a given response, and distinguishing them from additional redundant regulators or by product effects.

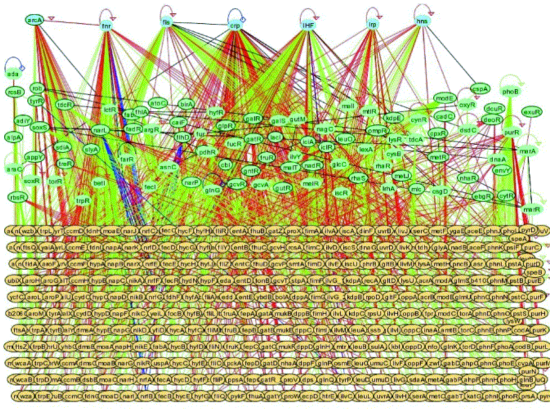
In this paper, we propose a novel approach to address these issues. We enforce biologically-motivated restrictions to limit the search to simple network structures that significantly reduce the space of possible networks, while highlighting the most relevant biological information. We devise a search algorithm that utilizes these structural constraints to efficiently find high scoring networks. Furthermore, under reasonable assumptions on the underlying probability distribution, we provide guarantees on our algorithm's performance, thus providing an approximation algorithm for a certain class of Bayesian networks. This is of particular interest, because approximation algorithms for learning Bayesian networks have only been developed for polytrees(Dasgupta, 1999).

We evaluate the performance of our algorithm on synthetic and real gene expression data sets for both yeast and mammals. Our results show good structure reconstruction on synthetic data and that the model learned from gene expression data generalizes well to unseen test data. Importantly, our results also illustrate the ability of the learned models to successfully reconstruct biologically correct regulatory relations in complex mammalian systems.

2. Regulation Model

Our gene regulation model is a Bayesian network that describes regulatory relations between genes. In this network, each random variable corresponds to the gene expression level of a specific gene. If gene Y is a parent of gene X in the Bayesian network, we interpret this as "Y *regulates* X". We denote by $\mathbf{Pa}_X$ the set of all regulators (parents) of the gene (variable) X. Any gene that "regulates" in our model is termed a *regulator*. The key point behind to our approach is that we enforce a number of biologically motivated constraints to limit these regulators and the graph structure.

Unlike a standard Bayesian network, we limit the possible regulators (parents) in the network to a set of candidate regulators $\mathcal{C}$. Our candidate set $\mathcal{C}$ is chosen based on prior biological knowledge, and contains known and putative regulators in the organism being studied. Note, that while finding



Current Opinion in Microbiology

Figure 2: A literature reconstruction of the Bacterial *E. Coli* regulation network from (Martinez-Antonio and Collado-Vides, 2003). Notice this includes small top layer of regulators and many targets for each.

which genes in a genome may function as regulators in general is often tractable, finding which regulators are *active* in a data set is difficult. Our inference will focus on this latter question.

In fact, previous biological studies indicate that only a small fraction of all potential regulators may be active in a given data set. Accordingly, we also constrain the structural properties of the graph, seeking a Bayesian network in which only a limited number of genes are regulators, *i.e.*, have an outdegree greater than zero. Moreover, extensive studies in both bacteria and yeast (Shen-Orr et al., 2002; Martinez-Antonio and Collado-Vides, 2003; Lee et al., 2002) (Figure 2) indicate that each such "master regulatory gene" may affect the transcription of many genes (indeed, only 3-6% of the yeast and human genes respectively encode TFs). Thus, we expect each regulator to have a high out-degree. These constraints result in a graph of small depth, in which layers containing a small number of regulators control a large bottom layer of target genes, consistent with current biological understanding.

In addition to its biological relevance, this network structure has an obvious statistical motivation: Only when a gene consistently scores high as a parent for many genes, do we believe it indicates a true signal. An occasional high score as a parent of a single gene is attributed to spurious chance. Since learning an accurate genetic network is not possible in the current data paucity in the gene expression domain, our restrictions represent a reasonable first order approximation of the network which preserves its biological relevance. In fact, for most biological applications, false positives are significantly more "costly" than false negatives, and finding a robust set of key regulators whom are most strongly supported by the data (as offered by our model) is a more important goal then discovering their complete set of targets.

We now provide a formal definition of our model: A *regulation graph* is a Bayesian network with the following restrictions on its structure.

Definition 1 *Given a set of random variables $\mathcal{X} = \{X_1, \ldots, X_n\}$, a set of* candidate regulators- $\mathcal{C}$ *and the constants d and k, we define a* regulation graph, $\mathcal{G}$ *to be a Bayesian network over* $\mathcal{X}$ *so that:*

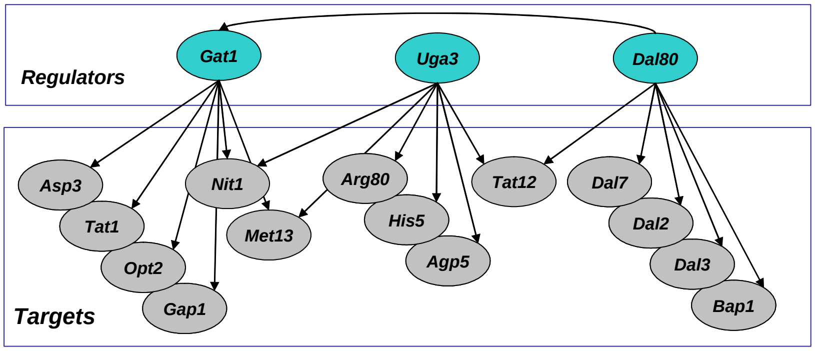


Figure 3: Regulation graph. The top layer is associated with the regulators and the bottom layer is associated with all other variables. The key concept behind the regulation graph is a small number of regulators, each with many targets. Note, the illustrated nitrogen catabolism response was automatically inferred by our algorithm from a gene expression data set.

- *All parents belong to the candidate set:* $\forall X, \mathbf{Pa}_X \subset \mathcal{C}$.
- *The number of parents for each variable (indegree) is bounded by* d*:* $\forall X, |\mathbf{Pa}_X| \leq d$.
- *The total number of parents in the model is bounded by k: We term the the union of all parent sets in the network to be the graph's* regulators, *denoted by* $\mathcal{R}$*, thus we constrain* $|\mathcal{R}| \leq k$.

■

The graph structure is best visualized as a graph with a shallow depth: in the top layers, a small set of regulators (chosen from a large set $\mathcal{C}$), possibly regulating each other and in the bottom layer, all other variables (see Figure 3).

2.1 Optimization Problem

Since a regulation graph is a Bayesian network, the straightforward approach to learning its structure would be to use the typical heuristic greedy hill-climbing search (Heckerman, 1998) used for this task. This involves traversing the space of legal models in a greedy fashion using local operators such as adding, removing or reversing a single edge. At each step, the operation that best improves the score is chosen.

Unfortunately, the standard approach is likely to fail as the limited number of regulators specified by the regulation graph could be quickly used up. For example, we may wish to search for a regulation graph over 2000 variables, limited to 30 regulators. We begin with the empty graph and in a greedy fashion add the optimal edge at each iteration. In many of these iterations, a new regulator is added to the regulator set $\mathcal{R}$. Therefore, after little over 30 iterations, no new regulators could be added to $\mathcal{R}$ and all subsequent legal steps would only involve adding edges from regulators already in $\mathcal{R}$. While these regulators might be the best parents for a small set of variables, thousands of other variables remain unexplained in the model.

In fact, contrary to learning regular Bayesian networks, the choice of parents for a variable X in our model is no longer independent of the parents chosen for other variables. Since the total number of parents in the model is limited to k, choosing a parent for one variable can limit the choice of parents for other variables. Thus, a regulator should be added to $\mathcal{R}$ only when it is a good parent for many variables concurrently. Any search algorithm we design must take this into account and add a regulator to a *set* of target genes in each greedy step.

Fortunately, once a regulator set $\mathcal{R}$ is given, finding the optimal regulation graph constrained to $\mathcal{R}$ is polynomial, and for most practical cases efficient. For any given variable, there is a small number, $\binom{k}{d}$, of possible parent sets (compared to $\binom{n}{d}$ possible parent sets for Bayesian networks bounded by an indegree of d).[1] Thus, it is quick to calculate the local score, denoted $Score(X;\mathbf{P})$, for all possible parents sets and choose the highest scoring parents.

$$\mathbf{Pa}_X = \text{argmax}_{\mathbf{P}\subset\mathcal{R},|\mathbf{P}|\leq d} Score(X;\mathbf{P}) \tag{1}$$

When $\mathcal{R}$ is not given, we can use use this property to efficiently evaluate the quality of any potential set of regulators.

Definition 2 *We define the* utility, $F(\mathbf{R})$ *of a regulator set* $\mathbf{R}$ *as:*

$$F(\mathbf{R}) = \sum_{i=1}^{n} \max_{\mathbf{P}\subset\mathbf{R},|\mathbf{P}|\leq d} Score(X_i;\mathbf{P}) \tag{2}$$

■

The utility of a regulator set $\mathbf{R}$ can be computed quickly and closely approximates the score of the optimal network constrained to $\mathbf{R}$, denoted SCORE$(\mathbf{R})$. $F(\mathbf{R})$ scores a graph structure resulting from independently choosing the optimal parents for each variable, and is therefore an upper bound on SCORE$(\mathbf{R})$. Note, that an independent choice of parents for each variable may lead to a structure containing cycles. Thus, a legal Bayesian network might have a some parent sets that score sub-optimally. However, since cycles can form only between the variables in $\mathbf{R}$, which is a relatively small part of the entire network ($k << n$), we expect that SCORE$(\mathbf{R})$ is usually only slightly less than $F(\mathbf{R})$. Furthermore, acyclicity can be resolved within a subgraph involving only k nodes (the complexity of solving this problem is constant in n, though exponential in k). In most practical cases, only a few short cycles form and the optimal solution can be easily found. In comparison, resolving cyclicity in a Bayesian network can be exponential in n ($k << n$), since cycles can form among any n variables.[2]

We treat $F(\mathbf{R})$ as a scoring function that measures the quality of regulator sets. This implies a new optimization problem to find a small set of regulators, $\mathcal{R}$, which maximize this score:

Definition 3 *The* Best Regulator Set *problem: Given a set of variables* $\mathcal{X}$, *a data set of samples* D, *a set of candidate parents* $\mathcal{C}$, *and the constants* d *and* k, *we wish to find*

$$\mathcal{R} = \text{argmax}_{\mathbf{R}\subset\mathcal{C},|\mathbf{R}|\leq k} F(\mathbf{R}) \tag{3}$$

■

[1] We typically use d ranging between 3 to 5 and k ranging between 30 to 70, whereas n is in the thousands and $|\mathcal{C}|$ is in the hundreds.

[2] In our typical setting where k ranges between 30 to 70 and $n \geq 2000$, the difference in score after breaking the cycles is negligible. With high probability this difference would not change our choice of regulating set $\mathcal{R}$. Therefore, for the remainder of this paper, we ignore the issue of cyclicity.

This problem is conceptually similar to the *Set Cover* problem, a classical hard problem. The challenge is that the regulator set $\mathcal{R}$ must be chosen from a much larger candidate set $\mathcal{C}$ and there are $\binom{|\mathcal{C}|}{k}$ possible regulator sets. While there does not seem to be any efficient algorithm to find an optimal solution, we next present an efficient algorithm that attempts to approximate it.

3. MinReg Learning Algorithm

We now turn to the task of learning a regulation model (specified by a set of regulators $\mathcal{R}$, and a parent structure on the variables in $\mathcal{X}$) from a training set ($D = \{\mathbf{x}[1], \ldots, \mathbf{x}[M]\}$, consisting of M instances drawn independently from an unknown distribution $P(\mathcal{X})$). Our goal is to choose the set of regulators $\mathcal{R}$ and learn the regulatory graph structure that best explains this distribution. We take a *score-based approach* to this learning task and we define a scoring function that measures how well each candidate model fits the observed data.

Given a scoring function, our task is to devise a search algorithm capable of efficiently finding a high scoring model. As discussed in Section 2.1, the hard part of the search is to find the optimal set of regulators, $\mathcal{R}$. Our novel greedy algorithm for this task, MinReg (sketched in Figure 4), begins with an empty set of regulators and an empty graph structure. At each iteration, for each possible candidate, we construct an increment regulator set by adding that candidate to the current regulator set. We calculate the score for each of the increment regulator sets and choose the one that gives the largest gain. Each time $\mathcal{R}$ is updated, we calculate the optimal regulation graph restricted to the current regulator set $\mathcal{R}$. We continue to iterate until some stopping criterion is reached.

A crucial point is to correctly define the gain of a given regulator at each iteration. We calculate a local score between a variable and its regulating *set*. When considering a new candidate regulator $c \in \mathcal{C}$ as a parent for a variable X, we measure not how well c scores for X, but how much additional gain c gives to X's local score. Thus, each of the regulating parents provide a distinct contribution to the score.

Definition 4 *We define the* marginal utility *of adding a regulator set* $\mathbf{C}$ *to an already chosen regulator set* $\mathbf{R}$ *as*

$$F(\mathbf{C} \mid \mathbf{R}) = F(\mathbf{C} \cup \mathbf{R}) - F(\mathbf{R}) \tag{4}$$

∎

Thus, at each iteration, we add the candidate regulator with the largest marginal utility.

3.1 α-modularity and Performance Guarantees

MinReg is a greedy algorithm, which at each iteration, adds to the model the best single regulator according to some local criterion This greedy approach does not necessarily lead to a global optimum. Can we characterize the situations in which the MinReg algorithm is lead astray? Consider the case where $Score(X;A) + Score(X;B)$ is significantly less than $Score(X;A \cup B)$. In this situation, neither A nor B would be attractive enough to get selected by themselves in any of the greedy steps, whereas their joint contribution may be significantly higher than any other combination of regulators. Thus, the greedy algorithm is misled to choose an inferior regulator set. Importantly, this is a biologically-plausible scenario, since synergy between regulators is a well-documented phenomenon.

```
MinReg Algorithm
Begin with an empty regulator set $\mathcal{R}$ and an empty graph
In each iteration find the candidate regulator with the highest marginal utility:
$c^* = \mathrm{argmax}_{c \in \mathcal{C}} F(c \mid \mathcal{R})$
    Iterate over each candidate regulator $c \in \mathcal{C}$
        calculate $F(c \mid \mathcal{R})$
        Iterate over each variable $X = X_1, \ldots, X_n$
            approximate its best parents restricted to $\mathcal{R} \cup c$
            $\max_{\mathbf{P} \subset \mathcal{R} \cup c, |\mathbf{P}| \leq d} Score(X_i; \mathbf{P})$
            Add local score of $X_i$ to utility of $c$
    Add best candidate regulator to $\mathcal{R}$ and update the regulation graph
until stopping criterion (3.3.1)
```

Figure 4: Overview of the MinReg algorithm. The algorithm consists of two nested greedy loops. The external loop finds the optimal set $\mathcal{R}$ of k regulators. For each $X \in \mathcal{X}$ an internal loop finds an optimal set of parents $\mathbf{Pa}_X$.

We argue that this characterizes the only situation in which our algorithm fails. We show that if we can bound the severity of such effects, we can derive a worst case error bound on the algorithm's performance: in this case, MinReg is an *approximation algorithm*, guaranteed to find a solution which is not too far from optimal. To formally prove this guarantee, we introduce the notation of α-*modular* functions.

Definition 5 *Let f be a function defined over subsets of $\mathcal{C}$. f is monotone increasing if forall subsets* $\mathbf{A}, \mathbf{B}$ *s.t.* $\mathbf{A} \subseteq \mathbf{B}$, *the following holds*

$$f(\mathbf{A}) \leq f(\mathbf{B})$$

▮

Definition 6 *(Lehmann et al., 2001) Let f be a function defined over subsets of $\mathcal{C}$. f is α-modular ($\alpha \geq 1$) if and only if for all subsets* $\mathbf{A}, \mathbf{R}$ *and for all singletons Z, the following holds:*

$$f(\mathbf{A} \cup Z|\mathbf{R}) \leq f(\mathbf{A}|\mathbf{R}) + \alpha f(Z|\mathbf{R})$$

▮

Note that this is a generalization of sub-modular functions, f is sub-modular for $\alpha = 1$. One might consider α as some measure on the convexity of f over the space of subsets from $\mathcal{C}$. For larger α, more "synergy" can be gained by joining sets together. We will show that if we can bound the amount of "synergy" between regulators, we can bound the error of our greedy algorithm accordingly.

Lemma 7 *The following are equivalent definitions of* α*-modular functions: ((Lehmann et al., 2001))*

1. *For any subsets* $\mathbf{S} \subseteq \mathbf{T}$ *and singleton* $Z \notin \mathbf{T}$ *we have* $f(Z|\mathbf{S}) \geq \alpha f(Z|\mathbf{T})$.
2. *For any subsets* $\mathbf{S} \subseteq \mathbf{T}$ *and subset* $\mathbf{V}$ *we have* $f(\mathbf{V}|\mathbf{S}) \geq \alpha f(\mathbf{V}|\mathbf{T})$.
3. *For any subsets* $\mathbf{A}, \mathbf{B}$, *we have* $f(\mathbf{A}) + \alpha f(\mathbf{B}) \geq f(\mathbf{A} \cup \mathbf{B}) + \alpha f(\mathbf{A} \cap \mathbf{B})$

These equivalent formulations offer us another perspective: the marginal utilities of α-modular functions are "almost" (up to a factor of α) monotone decreasing. This fits our intuition that as $\mathcal{R}$ grows larger, the utility of adding new regulators diminishes.

Theorem 3.1: If F is an α-modular and monotone increasing function,[3] then the MinReg algorithm (presented in Figure 4) is a polynomial time approximation algorithm for the Best Regulator Set: Denote by OPT_k the optimal k regulator set (i.e. that maximizes F) and by MINREG_k the regulator set found by the MinReg algorithm, then

$$(\alpha+1)F(\mathrm{MINREG}_k) \geq F(\mathrm{OPT}_k) \tag{5}$$

This theorem provides assurance that while MinReg is a very quick algorithm that greedily takes locally optimal steps, the score of the regulator set found by MinReg is not too far away (at most a factor of $1+\alpha$) from the optimal solution reached by exhaustively enumerating all possible regulator sets.

Proof: Our proof is by induction. For $k = 1$, the optimal solution is the best single regulator and this is exactly the regulator found by the MinReg algorithm therefore $\mathrm{MINREG}_1 = \mathrm{OPT}_1$. We assume that $(\alpha+1)F(\mathrm{MINREG}_{k-1}) \geq F(\mathrm{OPT}_{k-1})$ and prove it for k.

Set $J = \mathrm{argmax}_{I \in \mathcal{C}} F(I)$, the best single regulator in $\mathcal{C}$ and $\hat{J} = \mathrm{argmax}_{I \in \mathrm{OPT}_k} F(I)$, the best single regulator in OPT_k. Note, J is the first regulator chosen by the MinReg algorithm.

We define the following sub-problem imitating the behavior of the greedy algorithm. Let $\hat{F}(\mathbf{Y}) = F(\mathbf{Y} \cup \{J\}) - F(J)$, our goal is to find a set of $k-1$ regulators that optimize $\hat{F}$ on $\mathcal{C} \setminus \{J\}$. This is exactly what MinReg does after it chooses J in the first iteration. We denote by $\widehat{\mathrm{OPT}}_{k-1}$ and $\widehat{\mathrm{MINREG}}_{k-1}$ the optimal and greedy solutions respectively to this new sub problem. It is easy to see that $\hat{F}$ is a α-modular function and that our induction holds for $\hat{F}$ as well, that is, $(\alpha+1)\hat{F}(\widehat{\mathrm{MINREG}}_{k-1}) \geq \hat{F}(\widehat{\mathrm{OPT}}_{k-1})$.

By the inductive hypothesis, it suffices to show that the increment is α-modular, i.e.:

$$\begin{aligned} F(\mathrm{OPT}_k) - \hat{F}(\widehat{\mathrm{OPT}}_{k-1}) &\leq (\alpha+1)F(\mathrm{MINREG}_k) - \hat{F}(\widehat{\mathrm{OPT}}_{k-1}) \Rightarrow \\ F(\mathrm{OPT}_k) &\leq (\alpha+1)F(\mathrm{MINREG}_k) \end{aligned}$$

simply by subtraction of the same value on both sides. By the induction hypothesis on $\hat{F}$ we have that

$$\begin{aligned} F(\mathrm{OPT}_k) - \hat{F}(\widehat{\mathrm{OPT}}_{k-1}) &\leq (\alpha+1)(F(\mathrm{MINREG}_k) - \hat{F}(\widehat{\mathrm{MINREG}}_{k-1})) \\ &\leq (\alpha+1)F(\mathrm{MINREG}_k) - \hat{F}(\widehat{\mathrm{OPT}}_{k-1}) \end{aligned}$$

[3] While the marginal utilities should be almost monotone decreasing, we want the function itself to be monotone increasing.

because $(\alpha+1)\hat{F}(\widehat{\text{MINREG}}_{k-1}) \geq \hat{F}(\widehat{\text{OPT}}_{k-1})$. Note that

$$\begin{gathered} F(\text{MINREG}_k) - \hat{F}(\widehat{\text{MINREG}}_{k-1}) = \\ F(\text{MINREG}_k) - F(\widehat{\text{MINREG}}_{k-1} \cup \{J\}) + F(J) = F(J) \end{gathered}$$

since $\widehat{\text{MINREG}}_{k-1} \cup \{J\} = \text{MINREG}_k$ by the very way in which the MinReg algorithm works. Thus to prove the induction for k it is enough to show that:

$$F(\text{OPT}_k) - \hat{F}(\widehat{\text{OPT}}_{k-1}) \leq (\alpha+1)F(J) \tag{6}$$

Since $\widehat{\text{OPT}}_{k-1}$ is at least as good as any solution of size $k-1$, by definition:

$$\hat{F}(\widehat{\text{OPT}}_{k-1}) \geq \hat{F}(\text{OPT}_k \setminus \{\hat{J}\}) = F(\text{OPT}_k \setminus \{J'\} \cup \{J\}) - F(J) \tag{7}$$

By α-modularity of F:

$$F(\text{OPT}_k) \leq F(\text{OPT}_k \setminus \{\hat{J}\}) + \alpha F(\hat{J}) \tag{8}$$

Subtracting (7) from (8) gives:

$$F(\text{OPT}_k) - \hat{F}(\widehat{\text{OPT}}_{k-1}) \leq [F(\text{OPT}_k \setminus \{\hat{J}\}) - F(\text{OPT}_k \setminus \{\hat{J}\} \cup \{J\})] + [\alpha F(\hat{J}) + F(J)] \tag{9}$$

Monotonicity of F implies that $F(\text{OPT}_k \setminus \{\hat{J}\}) \leq F(\text{OPT}_k \setminus \{\hat{J}\} \cup \{J\})$, therefore, the first bracket gives a negative contribution. Maximality of J implies that $F(J) \geq F(\hat{J})$, therefore the second bracket is $\leq (\alpha+1)F(J)$. ∎

It remains to show that F is both monotone and α-modular. Recall, F is a sum of maximizations of local scoring functions: $F = \sum_{i=1}^{n} \max_{\mathbf{P} \subset \mathbf{R}, |\mathbf{P}| \leq d} Score(X_i; \mathbf{P})$. The local maximizations are clearly monotone, if $\mathbf{S} \subset \mathbf{T}$, then $\forall X, \max_{\mathbf{P} \subset \mathbf{T}} Score(X; \mathbf{P}) \geq \max_{\mathbf{P} \subset \mathbf{S}} Score(X; \mathbf{P})$. Thus F, being the sum of monotone functions, is monotone as well.

Empirically we observe that F is α-modular, usually for relatively small α (see 4.1.1). At first this might sound surprising, since as mentioned above, synergy is known to play an important role in biological regulation, and we do not expect $Score(X; \mathbf{P})$ to be α-modular in the gene expression domain. Fortunately, while regulators are synergistic for specific targets, F is a sum over thousands of variables. Even if the synergy between two regulators is very high, this synergy would need to hold for many targets, otherwise it would average out when summing over all of $\mathcal{X}$. We empirically tested the synergy between regulators and groups of regulators in both yeast and mammalian data sets, the worst factor we encountered was 1.2. Therefore, we make the assumption of α-modularity of F with $\alpha = 2$ in the gene expression domain.

3.2 Scoring Function

To define the local score $Score(X; \mathbf{P})$, we adopt the Bayesian paradigm and use the Bayesian BDe scoring function (Heckerman et al., 1994; Heckerman, 1998) commonly used for learning Bayesian networks. The Bayesian score evaluates the posterior probability of the graph given the data:

$$\text{score}_{\mathcal{B}}(\mathcal{G} : \mathcal{D}) = \log P(\mathcal{D} \mid \mathcal{G}) + \log P(\mathcal{G})$$

where $P(\mathcal{D} \mid \mathcal{G})$ takes into consideration our uncertainty over the parameters and averages the probability of the data over all possible parameter assignments to $\mathcal{G}$.

$$P(\mathcal{D} \mid \mathcal{G}) = \int P(\mathcal{D} \mid \mathcal{G}, \theta) P(\theta \mid \mathcal{G}) d\theta$$

The particular choice of the priors $P(\mathcal{G})$ and $P(\theta \mid \mathcal{G})$ determines the exact Bayesian score.

The BDe score refers to a certain class of priors with several desirable properties (as detailed in (Heckerman, 1998))In particular, the BDe score of an entire regulation graph $\mathcal{G}$, decomposes into sum over the local scores for each variable.

$$\text{score}(\mathcal{G} \,:\, D) = \sum_i Score(X_i; \mathbf{Pa}_{X_i}) \tag{10}$$

We use these local decomposed scores as the local score for our algorithm.

3.3 MinReg Implementation

A general overview of the MinReg algorithm was presented in Section 3. Several details in MinReg's implementation lead to substantial speed-up of the naïve algorithm, such that our implementation can generate a model over thousands of genes within few minutes.

First, we define a function f_X for each variable X, $f_X(\mathcal{R}) = \max_{\mathbf{P} \subset \mathcal{R}, |\mathbf{P}| \leq d} Score(X; \mathbf{P})$. This is the optimal contribution of X to F, restricted to a regulator set $\mathcal{R}$. Thus, we have 3 levels of scoring functions: *Score* - for a particular variable and its parents, f_X - the optimal score of a single variable X, and $F(\mathcal{R}) = \sum_{X \in \mathcal{X}} f_X(\mathcal{R})$.

The naïve greedy algorithm has k iterations. In each iteration, $F(c|\mathcal{R})$ is calculated for all $c \in \mathcal{C}$. Since each calculation of $F(c|\mathcal{R})$ requires calculating $f_X(c|\mathcal{R})$ for all $X \in \mathcal{X}$, f_X is calculated $k|\mathcal{C}||\mathcal{X}|$ times. Calculation of f_X requires calculating *Score* for each of the $\binom{k}{d}$ possible sets of parents. While this is constant in n, in practice k^d could be very large. We devise a number of heuristics based on α-modularity to reduce the number of times we need to calculate each of the 3 functions F, f_X, and *Score*.

We employ a branch and bound approach to $F(c|\mathcal{R})$, using the α-modularity of F to filter out candidates with little potential. In the first iteration, for all $c \in \mathcal{C}$ we calculate $\text{Util}(c) = F(c)$. We store the candidate regulators in a heap sorted by $\text{Util}(c)$. At any given time, $\text{Util}(c) = F(c|\mathbf{A})$, for some $\mathbf{A} \subseteq \mathcal{R}$. The α-modularity of F ensures that $\alpha\text{Util}(c) \geq F(c|\mathcal{R})$ (see Lemma 7). In most cases, we expect the regulator with the highest marginal utility to be toward the top of the heap.

Once a new regulator is added to $\mathcal{R}$, the marginal utilities change and need to be recalculated. In each subsequent iteration, we traverse down the heap and only re-evaluate candidates for whom $\alpha * \text{Util}(c)$ is greater than the best marginal valuation found thus far, denoted c^*. Each time $F(c|\mathcal{R})$ is calculated, we use this value to update $\text{Util}(c)$ in the heap. Once we reach a candidate such that $\alpha * \text{Util}(c) < F(c^*|\mathcal{R})$ we stop traversing the heap, as α-modularity ensures that none of the candidates beyond this point can be better than c^*. While this branch and bound does not change the worst case complexity, for most practical cases, only the few topmost candidates are examined in each iteration.

While the previous speed-up came at no loss in the quality of the final solution, the next two heuristics reduce accuracy. These heuristics are based on the assumption that *Score* is α-modular in most cases (though probably by a larger factor). This assumption is reasonable (albeit not always accurate). While regulation is sometimes synergistic, functions such as XOR are rare in biology and even when synergy exists, it is bounded by a reasonable constant. More importantly, we expect the synergistic pairs are themselves uncommon.

Similarly to how $\text{Util}(c)$ approximates $F(c|\mathcal{R})$, we cache $\text{Util}_X(c)$ as an approximation of $f_X(c|\mathcal{R})$. Whenever $F(c|\mathcal{R})$ is calculated, we do so only approximately: $F(c|\mathcal{R}) = \sum_{X \in \mathcal{X}} \text{Util}_X(c)$.

In the first iteration we initialize $\text{Util}_X(c) = f_X(c)$, for each $c \in \mathcal{C}$ and $X \in \mathcal{X}$. In subsequent iterations, we only recalculate $f_X(c|\mathcal{R})$ (and update $\text{Util}_X(c)$) for those X's whose parent set $\mathbf{Pa}_X$ changed in the previous iteration. This is especially effective in later iterations where $\mathbf{Pa}_X$ rarely changes.

Finally, instead of calculating f_X exactly, we approximate it using a greedy algorithm similar to Figure 4. We start with no parents and at each iteration add best parent, a$rgmax_{c \in \mathcal{R}} f_X(c|\mathbf{Pa}_X)$ to $\mathbf{Pa}_X$. This only requires $d|\mathcal{R}|$ calculations of *Score* instead of $\binom{|\mathcal{R}|}{d}$.

3.3.1 Stopping Criterion

Formally, the best regulator set problem requires a predefined constant k, specifying the number of regulators in the model. However, there are no obvious biological grounds for choosing a particular "good" k, as there is a trade-off between fine resolution (offered by a larger number of regulators) and statistical robustness (from a small number of regulators).

We address this fine balance by taking an adaptive approach. We devise a *stopping criterion* for the addition of new regulators to our model. We continue to add regulators as long as their contribution to the score is significantly better than random regulators. We generate a set of m random regulators with similar properties to the real candidate regulators in our data. We construct these by sampling regulators with replacement from the original candidate regulator set. For each sampled regulator, we randomly permute the order of its samples. Thus, the random regulators have the same distribution over their values, but these are independent of the target variables. We calculate the score for these random regulators in a manner similar to the real candidate set and store these in a heap. This provides us with an empirical distribution for the score of a random regulator.

We continue to add regulators to our model as long as they score greater than the random candidates. We update the scores in the candidate heap in a similar manner to the real candidates, pruning the heap using α-modularity. We stop once a random regulator scores better than any real regulator.

4. Experimental Results

We evaluated our algorithm on two data sets, a compendium of yeast expression profiles and a data set of mouse B-lymphocyte expression profiles. The distinct two data sets provide us each with a different realistic evaluation context. The yeast *S. cerevisiae* is the most extensively studied organism on a genomic scale, and has the most extensively characterized regulatory system among eukaryotes. In addition to an extensive amount of microarray data, identifying cis-regulatory elements (Bussemaker et al., 2001) and TF binding events (ChIP-chip experiments) (Lee et al., 2002; Harbison et al., 2004) is tractable on a genomics scale. Finally, decades of careful studies on individual gene functions are documented in a genome database (Cherry et al., 2001). Together, these data sources will allow us to carefully evaluate the success of our method in light of current biological knowledge.

Mammalian regulatory systems, such as those of the laboratory mouse, are notoriously difficult to elucidate, both experimentally and computationally. First, these networks are significantly more complex, involving a larger number of regulators, binding to long promoter and enhancer sequences. In particular, different cell types and cell states employ different regulatory networks to process signals. Furthermore, this complexity renders both genomics studies of regulatory events (such as ChIP-chip experiments) and computational ones (such as discovery of *cis*-elements) difficult or intractable. In fact, the sole available source of relevant data in mammalian systems is typically microarray measurements of expression profiles. Importantly, no successful method was demonstrated to date for reconstructing regulatory networks from mammalian expression profiles. Even partial success of MinReg in reconstructing mammalian regulation would constitute a significant scientific advance.

The yeast data set contained 358 samples combined from the Compendium (Hughes et al., 2000) and stress (Gasch et al., 2000) data sets.[4] We compiled a set $\mathcal{C}$ of 466 candidate regulators for yeast, which includes any gene with a potential regulatory role based on annotation or sequence homology. The expression profiles were discretized into 3 values: *down-regulated*, *no change* and *up-regulated*.[5] We included only 3755 genes with significant change in gene expression in at least 15 samples. We set the maximal indegree, $d = 3$, a reasonable estimate for the regulation of most yeast genes (in particular under a limited number of condition).[6] We conservatively set $\alpha = 2$ based on empirical evaluation of the data (see Section 4.1.1 below). We applied our MinReg algorithm to this data set resulting in a yeast regulation model with 44 key regulators.

The mouse data set consisted of 204 samples from purified spleenic B-lymphocytes (Sambrano et al., 2002), subjected to a number of stimuli (ligands) and combinations of these stimuli. We compiled a list of 684 candidate regulators using criterion similar to the yeast candidate regulator set.[7] We discretized the data as in yeast, except that the data was discretized into 5 levels: strongly down regulated, weakly down regulated, no change, weakly upregulated and strongly upregulated. We included only the 4373 genes that significantly changed in at least 18 samples. We ran the MinReg algorithm on this data and inferred a regulation model with 75 key regulators.

We employed both statistical and biological criteria to evaluate the performance of the algorithm. We examine our assumptions of α-modularity, the ability of our algorithm to generalize to unseen data, and the accuracy of the reconstruction on synthetic data. To demonstrate the accuracy of our algorithm in reconstructing the real yeast and mammalian regulatory network, we devise an approach to infer regulator function from our model, and compare that to the known central regulators in the relevant biological processes.

4.1 Statistical Evaluation

In this section we well evaluate the statististical robustness of the MinReg algorithm. We focus on two issues, is the assumption of alpha-modularity a reasonable one for our gene expression domain, and how well does our learned model generalize to unseen test data.

[4] The Compendium (Hughes et al., 2000) contains 276 deletion mutants from various functional classes and the Stress data set (Gasch et al., 2000) contains 82 samples of responses to 12 different stress conditions.

[5] The Bayesian score is based on a multinomial distribution. Exact continuous measurement is a very noisy estimate of the actual gene expression and in our experience, discrete states better represent gene activity. We used a soft discretization based on a linear piecewise step function for each level of activity.

[6] While some genes might have more regulators, there is not enough data to learn such complex regulatory function from so few samples. Importantly, our goal is to robustly learn the key regulatory relations, not the full detailed network.

[7] Mouse has many more known regulators, but only 1/3 the mouse genome was printed on the microarray and only these genes were included in the analysis.

4.1.1 ALPHA MODULARITY

MinReg employs a greedy approach, evaluating only the addition of single regulators to the model at each iteration, potentially missing a better combination of regulators.[8] Based on the assumption of α-modularity of F, Theorem 3.1 ensures that the score of the greedy solution is not much worse than the score of the optimal solution. Furthermore, to improve the speed performance, the α-modularity of F is used strongly by the implementation to bound the number of the regulators that are evaluated from the heap at each iteration.[9]

To empirically evaluate the α-modularity of F in the two data sets used, we calculated the pairwise gain in score for all pairs of regulators in the candidate set $\mathcal{C}$. Thus, for each pair of candidates c_1 and c_2, we calculated the worst α using $F(c_1 \cup c_2)$, $F(c_1)$ and $F(c_2)$. In addition, we calculated the worst α for 10,000 pairs of random subsets, $\mathbf{C}_1, \mathbf{C}_2 \subset \mathcal{C}$, ranging between 2 to 8 regulators each, using $F(\mathbf{C}_1 \cup \mathbf{C}_2)$,$F(\mathbf{C}_1)$ and $F(\mathbf{C}_2)$. For the yeast and B-lymphocyte data, the worst α empirically encountered were 1.184 and 1.229, respectively. We used $\alpha = 2$ as a conservative overestimation to determine when to stop evaluating candidates in the heap at each iteration.

To further boost speed, we make a weak (but inaccurate) assumption that *Score* is close to α-modular, allowing MinReg to reconstruct a large network over thousands of genes in a few minutes, rather than overnight. We evaluated the effect of this additional modification on MinReg's performance by comparing its affect on the likelihood of test data in cross validation, as well on the enrichment of GO annotations in target sets (see sections 4.1.2 and 4.2.1) Indeed, based on these two criteria, assuming α-modularity of *Score* does not hurt the algorithm's performance.

4.1.2 CROSS VALIDATION

To evaluate the statistical robustness of our learned model and its ability to generalize to unseen data, we tested MinReg's performance in 5-fold cross validation. We randomly split the data into 5 equal parts, and ran MinReg 5 times. Each time using 4/5 of the samples as training samples to to learn both the structure and parameters of the regulation model, and withholding 1/5th of the data samples as a testing set. We then used the inferred model and gene expression of inferred regulators in the test data to predict the expression levels of all 3755 variables in each test sample. That is, given the expression of regulators in the mth sample, we use $P(X \mid \mathbf{Pa}_X[m])$ to predict the value of X in that sample.

We compared the likelihood of test data in several different models. As a baseline, we used the marginal probability of each variable to predict its value. Since most of the variables had a high frequency of the value 0 (their corresponding gene's expression remained unchanged most of the time), even this simple predictor scored well (Figure 5, crosses). As competition to our MinReg algorithm, we generated 44 clusters using standard k-means clustering (Duda and Hart, 1973; Tavazoie et al., 1999) and randomly chose from within each cluster a gene $r \in \mathcal{C}$ as its "regulator". For each cluster we used $P(X \mid r)$ as our predictor. While cluster representatives somewhat improved the prediction over the baseline (0.06 log-loss/instance, Figure 5, circles), our MinReg algorithm clearly provided the best predictions (0.11 log-loss/instance, Figure 5, triangles). In conclusion, our cross-validation demonstrates that the model generated by the MinReg algorithm performs well on

[8] This assumption is made implicitly by the classic and widely used greedy Bayesian network learning algorithm (Heckerman, 1998), that considers greedy moves of adding, removing and reversing a single edge at each step.

[9] Typically, after the first few iterations, only a few regulators are evaluated at each iteration.

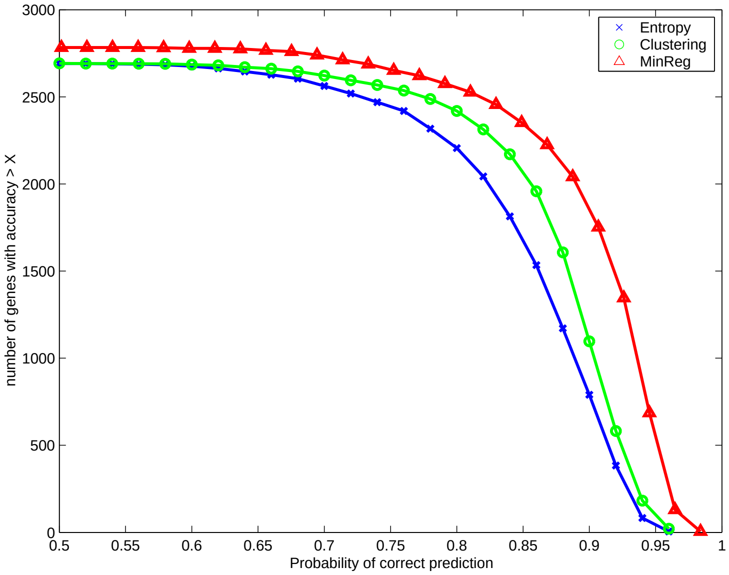


Figure 5: Cross validation of the predictive capabilities of our model on test data. The graph measures the number of variables correctly predicted at each probability. We compare our model (triangles) to the null model (crosses) that uses the marginal distribution of each variable and to a model based on cluster representatives (circles)

test data and that most of the information in an entire microarray can be captured by a small set of key regulators.

4.1.3 SYNTHETIC DATA

To evaluate the accuracy of the MinReg algorithm in a controlled setting, we generated synthetic data from a known regulation network. This gives a known ground truth to which we can compare the learned models. To make the data realistic, we generated synthetic data from the regulation model inferred from the yeast gene expression data above. While the inferred network is less complex than a true biological network, both share the same underlying probability distribution of the discretized data. We randomly sampled 10 data sets from this regulation model, each set consisting of 358 samples (same number of samples as the original data set). We tested MinReg's ability to reconstruct the correct network independently on each of these 10 synthetic data sets.

Our first test evaluated MinReg's choice of regulators. On average (over 10 repeats), MinReg correctly reconstructed 84% (39) of the generating 44 regulators. The worst case was 80% (35) and best case 91% (40). As for false positives, on average 74% of the reconstructed regulators were correct (worst case 71% and best case 77%).

Next we evaluated the detailed model itself. The generating model contained 6616 edges and we checked how many of those were correctly recovered. On average 70% of these were recovered (worst case 69% and best case 72%). In each model the percent of correct edges was a bit higher, with an average of 74% (worst case 72% and best case 77%). Even when we did not limit to candidate regulators (setting $\mathcal{C} = \mathcal{X}$) our reconstruction of regulators was surprisingly good, 42/47 of the regulators were correct and 76% of the individual edges were correctly reconstructed.

In summary, using only a small number of samples, MinReg is capable of learning a model over thousands of variables, reconstructing most of the relationships correctly.

4.2 Biological Evaluation

The crucial test for the success of our approach is in reconstructing the main aspects of a real regulatory program. The true underlying biology is vastly more complex than the simple regulation model that generated the synthetic data. In a real biological system, there are more regulators working together in more complex functions, feedback loops, and unobserved events. Furthermore, the expression data probing these is noisy. Unfortunately, since our knowledge on the principles and specifics of real regulatory networks is limited, so is our ability to test our model based on "realistic" simulated data. Due to this lack of biological knowledge we also do not have a gold standard network in any organism.

Fortunately, while evaluating each specific connection is impossible at the moment, we can estimate whether our model has correctly captured the overall biological regulatory events in the system. To do this, we rely on the functional annotations available for many genes, which describe (using the controlled vocabulary of the Gene Ontology (Consortium, 2000)) the molecular function, biological process and cellular location of individual genes. Thus, as described below, we will evaluate our reconstructed model by the ability to use it to correctly deduce the functional annotation of regulators, and by the fit of these annotations to the relevant biological system. Importantly, such functional characterization of key regulators is a critical biological task in its own right.

4.2.1 ANNOTATING REGULATORS

Our approach is based on the understanding that the biological function of a regulator is mediated by its set of targets. Therefore, the common shared function of its set of targets (*e.g.* enzymes involved in amino acid (AA) metabolism) characterizes the overall biological process it regulates (*e.g.* amino acid metabolism). Continuing with the AA metabolism example, if our reconstructed model is good, we expect a regulator of AA metabolism to have many inferred targets involved in AA metabolism. More generally, we expect that the function associated with a regulator based on its set of targets in the model (as reflected by significant enrichment for a particular annotation) will fit with the known function of this regulator (as reflected by its own annotation).

More formally, we denote by $\mathcal{X}_r$ the set of targets of a regulator r in our network structure. For each annotation term A, we calculate the fraction of genes in $\mathcal{X}_r$ associated with A and use the hyper-geometric distribution to calculate a p-value for this fraction. We report for each regulator, all the significant annotation terms with which it was associated and compare them to the known annotations for that regulator, and to the main functions expected in the biological systems we examine.

4.2.2 EVALUATING THE YEAST REGULATORY NETWORK

Based on this test, our reconstructed yeast network corresponds well to previous findings. Specifically, the model derived functions for 8 of the top 10 regulators (sorted by p-value) coincided with their known biological roles. Of the remaining two regulators, we were able to assign a putative role to one previously uncharacterized gene, but failed to identify the correct role of the other.[10] Furthermore, we examined numerous individual edges underlying these derived associations demonstrating that they are indeed supported by previously described regulator-target relations, lending support to our global analysis.

Additionally, the sequence motif representing binding site preference, is known for many yeast transcription factors. As an additional source of validation for MinReg's target sets, we use a putative map that uses these motif models to predict the gene targets of these transcription factors (Harbison et al., 2004). Similarlly to testing enrichment for GO annotations, we tested for enrichment of motifs in the promoter regions in each target set. In the case of signaling proteins, we tested for enrichment of the known transcription factor target activated by the signaling protein (see figure 1). A correct match between regulator and motif was found for 6 of the top 8 regulators,[11] Sst2 (Ste12), Met28, Uga3, Slt2 (Rlm1), Tpk2 (Msn2/4), and Tec1. Together the correct functional annotation and the occurance of the known motif in the regulatory regions, strongly support the quality of our reconstructed network. A more detailed and biologically oriented analysis of a simplified version of the proposed algorithm has been published in (Pe'er et al., 2002).

4.2.3 THE IMPORTANCE OF CANDIDATE REGULATORS

The pre-defined set of candidate regulators, C, is the only source of prior biological knowledge to our algorithm. In addition to focusing the model on regulatory relations, it narrows the search space and significantly reduces the running time of the algorithm (which is quadratic in the size of candidate set).

To assess the impact of this prior knowledge on MinReg's success, we examined MinReg's biological accuracy when run on the yeast data set, in the absence of a pre-defined candidate regulator set (*i.e.* $\mathcal{C} = \mathcal{X}$, such that any gene can be chosen as a regulator).[12] MinReg chose 35 regulators, only 6 of which were in the original candidate regulator set. This lack of "true" regulators suggests the expression of co-regulated genes is often at least as predictive (and sometimes even more) than that of the true regulating gene. While this model might be highly predictive and even generalizes well to new data, it does not reconstruct biological regulation, and is difficult to interpret.

Nevertheless, while $\frac{6}{35}$ is only a small fraction of the chosen regulators, this is a significant enrichment compared to their fraction in the candidate set (pvalue 0.003). The fact that our procedure results in a statistically significant enhancement of regulators is encouraging. We speculate that in complex organisms, where combinatorial regulation is expected to play a bigger role, this approach will be even more successful in detecting genes with a regulatory function.

4.3 Analysis of Mouse Data

In contrast to the significant success of several methods in reconstructing yeast regulatory networks, no algorithm has so far successfully reconstructed a mammalian regulatory network. To examine

[10] The results for the top 20 regulators are of similar quality: the associations for 13 regulators correspond their known function,four regulators were previously uncharacterized and the associations for three regulators are unsupported.

[11] These are 8/10 regulators we evaluated for gene function above, excluding 2 regulators for which no motif is currently known.

[12] Since the algorithm is quadratic in $\mathcal{C}$, we reduced $\mathcal{X}$ by including only genes whose expression significantly changed in ≥ 18 samples (versus 15 samples). This resulted in a set of 2828 genes for both $\mathcal{X}$ and $\mathcal{C}$. It is important to note that in our data set, the expression of many candidate regulators remains almost constant. Only 148 genes from our original candidate set of known and putative regulators were included in the 2828 genes.

whether MinReg can scale up to the challenge of a mammalian system, we evaluated the biological accuracy of its reconstruction of a B-lymphocyte regulatory network.

We first examine whether the key regulators identified by the algorithm are known to participate in the main biological process taking place in B lymphocytes under the tested stimuli- the decision between cell proliferation and cell death. Indeed, the inferred regulator set $\mathcal{R}$ includes the top five genes - Trp53, Nfkb1, Jun, Fos and Bak1 - known to play a pivotal role in this decision. 16 additional inferred regulators are known to be directly involved in the regulation of proliferation and cell death[13] and seven others are involved in the regulation of the cell division cycle.[14] Overall, 28/75 regulators are known to participate in regulation of the central process occurring in these cells. Importantly, in multi-cellular organisms such as mouse, each cell type is characterized by a distinct regulatory network (although some of the sub-systems may be used in different types of cells). Indeed, 28 of the 75 inferred regulators are known to be involved in lymphocyte regulation: 7 genes (Nfkb1, Jun, Fos, Daxx, Syk, Gnai2, Csf1r) are known central regulators in B-lymphocytes, 15 genes are known to be active in the regulation of lymphocytes in general, and 6 others encode cytokines and their receptors (important in the regulation of immune cells, including lymphocytes). Taken together, this analysis indicates that 44 of the 75 inferred regulators are known regulators of lymphocytes, cell proliferation and death or both.[15] This suggests, that when applied to a complex mammalian data set, MinReg is able to identify the key regulatory genes active in this system. Finding such central regulatory genes is still a major biological task in most systems.

We next examined the quality of our model structure, based on its ability to predict the detailed function of individual regulators (as described above for yeast). For each regulator, we compared the 3 top significant annotation terms ($P < 0.05$) based on its predicted targets with its known annotation terms (typically 5-6 per regulator). We defined 5 different categories[16] and evaluated the significance of our results by comparing to the null model of randomly assigning each regulator with 3 GO annotations (out of 2694 annotations tested). Based on these criteria the predicted function for over half (45/75) the regulators had at least some support in prior biological knowledge. Specifically, the predicted functional annotation of 6/75 regulators was "very good" ($P < 10^{-18}$), "good" for 28/75 additional regulators ($P < 10^{-35}$), and "weak" for 11/75 genes. 12/75 genes had "no match" to any annotations, but many of them were genes encoding relatively uncharacterized regulators with little or no known annotations. Importantly, only 16 of 75 regulators were assigned no significant annotation, indicating the biological coherence of our reconstructed model, where regulators are associated with functionally related targets.

To illustrate the quality of our findings, we highlight several specific examples. First, we note that some of the "very good" annotations demonstrate that MinReg can provide an extensive biological characterization of the regulatory function of genes. For example, our model indicates that the protein Map3k1 functions in the MAP Kinase Signaling Pathway has a signal transducer activity and works in the Growth factor signaling pathway. Importantly, the model also identified several of Map3k1's targets, including Fos and Nfkb1. This is a highly accurate characterization of the

[13] They are Aaft, Daxx, Foxo1, Gadd45g, Gnai2, Hipl2, Igbp1, Il2rh, Jund1, Itgb4, Map3k1, Rgs15, Rras2, Rsu1, Socs1, and Zmynd11.

[14] They are Ax1, Camk2b, Csfir, Elk3, Maf, Tbl1, Rgs2, and Tbl1.

[15] We expect that many of the other inferred regulators may be just as correct, and are simply not characterized by current biological knowledge. They suggest therefore novel biological hypotheses for experimental validation.

[16] Our categories are: "Very good" (more than one exact match); "good" (1 exact match), "weak" (1 approximate match to a related term), "no match" (significant annotation were associated with the regulator but none match any known annotations) and "no p-value" (no significant annotations were associated with the regulator).

molecular function of this protein, and of its biological regulatory role. Since Map3k1 is a signaling protein (rather than a transcription factor) this is a particularly important achievement, since direct assays of regulatory function (based on cis-elements and protein-DNA binding) cannot help in this task. Indeed, our method detects and correctly associates a whole range of regulators - including transcription factors, kinases and phosphotases. For example, Aatf, the apoptosis antagonizing transcription factor, is correctly associated with the apoptotic pathway and the mitotic cell cycle. Dusp4, the dual specificity phosphatase 4, is correctly associated with protein-tyrosine-phosphatase activity and MAPK signaling pathway. Finally, we emphasize that the main benefit of our approach is in suggesting novel hypotheses for further research. Thus, “weak” and “no match” associations may present the most important biological leads emanating from MinReg’s results. For example, the Jun oncogene (assigned to the “Weak” category), was predicted by our method to be involved in the Oncogene associated pathway and Cell proliferation and differentiation. While multiple abstracts in the published literature clearly and strongly support these two associations, Jun’s current GO annotation includes neither.

4.3.1 COMPARISON TO MODULE NETWORKS

Does MinReg have significant benefits in reconstructing mammalian regulatory networks over other (related) reconstruction approaches? To address this question, we compared MinReg’s performance on the B-lymphocyte data set to that of the Module Networks algorithm (Segal et al., 2003, 2005). Similar to MinReg, Module Networks associates a regulator to its targets based solely on dependencies in gene expression. However, while MinReg considers each target gene separately, Module Networks groups targets into sets (“modules”), such that all module genes share exactly the same regulatory program. In previous work (Segal et al., 2003), Module Networks was shown to be highly successful in reconstructing the yeast stress regulatory network. Here, we applied Module networks to the B-lymphocyte data and learn 75 modules and their associated regulation programs, involving 216 regulators overall. While many of the regulators overlapped those chosen by the MinReg algorithm,[17] these did not include 3 of the 5 known central regulators of cell proliferation and death (Nfkb1, Fos, nor Bak1) identified by MinReg.

For comparison, we can evaluate the Module Networks model by annotating each of its 206 regulators based on its associated targets (compiled across all 75 modules), resulting in 196 significantly annotated regulators. When evaluating the annotation quality of the top 75 regulators (sorted by p-value) by the same scale described above, we did not receive similarly significant results. In fact, only 13/75 genes had any support in prior biological knowledge (1/75 scored “very good”, 7/75 scored “good”, and 5/75 scored “weak”). Furthermore, when examining only regulators identified by both algorithms, MinReg’s associations outperform Module networks on 23 regulators, while Module networks only found a better association for one regulator (Gnai13). Thus, in the specific task of characterizing the molecular function and biological process controlled by a regulator, MinReg overwhelmingly outperforms Module Networks in this mammalian data set. This suggests that the detailed network and regulatory targets identified by MinReg are more accurate than those discovered by Module Networks.

What may be the underlying reason for MinReg’s success over Module Networks? The central goal of Module networks is to decompose the space of all genes into functionally coherent co-

[17] Module networks learned on the real valued data, rather than the discretized expression values, further supporting the robustness of the overlapping set of regulators.

regulated modules, at the "cost" of constraining them to share exactly the same set of regulators. While this constraint increases the statistical robustness and biological coherence (leading to a major success on yeast data), it may be less suited to complex mammalian regulatory systems. In contrast, MinReg focuses on finding the most dominant regulators and their targets in the data. A regulator is only assigned to a target if that specific edge is sufficiently supported by the data [18] and each gene chooses its unique set of regulators. We believe these two reasons combined led to MinReg's superior performance in regulatory reconstruction.

5. Discussion and Conclusions

We have introduced the MinReg framework, a constrained Bayesian network for the reconstruction of regulatory networks. The framework limits the total number of parents in the model, thus focusing on only a small parsimonious set of key regulators. We exploit these constraints to devise a novel efficient approximation algorithm to search for a high scoring network from expression data. Under reasonable assumptions on the underlying probability distribution, we can prove guarantees on our algorithm's performance. To derive these guarantees, we introduce the notion of α-modularity, a convexity measure of the scoring function over the space of possible parent sets. Approximation algorithms with a performance guarantee rarely exist for Bayesian networks (Dasgupta, 1999) and we hope this measure can be used to derive addition performance bounds for other sub-classes of Bayesian networks.

Machine learning in the gene expression domain is especially challenging as it requires learning structures over thousands of variables using at most hundreds of samples. Our extensive experimental results on real expression data demonstrate that our framework is up to this challenge: we successfully infer regulatory relations over thousands of genes within minutes. Our results are validated by statistical criteria (synthetic data, cross-validation) and biological ones (our ability to correctly infer a correct set of key regulators and their detailed regulatory functions). Importantly, unlike previous approaches, our method scales well to complex mammalian systems, discovering key mammalian regulators (both signaling proteins and transcription factors) solely from expression data.

While constraining the number of regulators carries obvious statistical and computational advantages, what does it cost us in biological accuracy? We claim that the focus on a small and parsimonious regulatory set is as motivated biologically as it is statistically. Most importantly, any complex biological network involves a multitude of genes and proteins, but biologists' primary goal is most typically to find the central genes, that play the most important functional role in the system. In fact, a full and accurate model of the exact network at a given point, may fail to highlight those central genes. Rather, by focusing on a small set of key regulators, MinReg can provide clear critical leads for further research. Indeed, our analysis of the B lymphocyte data set indicates that MinReg is able to focus on the very key regulators of a complex process (cell proliferation and death) as well as on a significant number of cell specific regulators. Using an established "guilt-by-association" approach (Wu et al., 2002; Ihmels et al., 2002), we further capitalize on the learned structure, and identify the accurate functional roles of these proteins in regulating cellular processes. This is a major feat, never before accomplished by a computational algorithm for a mammalian system. Importantly, MinReg is not only superior to standard clustering, but it overwhelmingly outperforms in this task the recently published Module Network algorithm (Segal et al., 2005).

[18] Many genes in the final model do not have any regulator, as none scored well enough.

Our method relies on the assumption that regulatory interactions between genes often result in corresponding statistical dependencies between random variables representing their expression. Recently, there have been a number of successful attempts to use other data sources - such as *cis*-elements (Bussemaker et al., 2001; Segal et al., 2002) and transcription factor binding events (Bar-Joseph et al., 2003) to infer regulatory relations in yeast. However, these successes cannot scale well to mammalian systems, in which computational detection of *cis*-elements is far less tractable (due to long and ill-defined promoters), and experimental detection of binding events is currently very limited (due to the genome size and the difficulty in carrying such experiments *in vivo*). In contrast, the collection of mammalian expression data is growing at an exponential rate, and methods such as MinReg that rely solely on gene expression for network reconstruction are direly needed.

MinReg lies between two graphical model based approaches for learning regulatory networks: unconstrained Bayesian networks and Module Networks Segal et al. (2005). While unconstrained Bayesian networks allow for a reconstruction of finer structure, they have only been successful at reconstructing small networks or subnetworks consisting of only a few variables Pe'er et al. (2001); Hartemink et al. (2002); Imoto et al. (2002). In contrast, MinReg and Module Networks can reconstruct a network over thousands of variables, based on the assumption that a small number of regulators can be chosen from a pre-defined candidate set. All three approaches, assume that regulator expression can be a proxy for its activity. Bioinformatics validation (of all approaches), and experimental validation (of Module Networks (Segal et al., 2003)) indicates that they can be at least partly successful in this task. This success is somewhat surprising, since actual protein activity depends on many biochemical events in addition to mRNA transcription.

What accounts for the significant success of MinReg compared to Module networks in mammalian network reconstruction? In the Module Network approach, genes are grouped into modules, thus losing their individual identity and distinction. MinReg provides a finer structure, allowing each gene an individual set of parents and regulatory function. Many recent biological papers stress the importance of modularity in biological networks (Hartwell et al., 1999; Ihmels et al., 2002; Segal et al., 2003; Bar-Joseph et al., 2003). Such organization facilitates orchestrating coordinated responses to external and internal signals by co-regulating genes that participate in a common function or task. While modularity may be a general organizing principle of regulatory networks, it may be too coarse grained on it own to represent the complex coordination between multiple genes and biological process. Rather, complex mammalian regulation is probably orchestrated by few key regulators, which combine together to regulate the genome, one target at a time through its unique regulatory program.

References

Z. Bar-Joseph, G. K. Gerber, T. I. Lee, N. J. Rinaldi, J. Y. Yoo, F. Robert, D. B. Gordon, E. Fraenkel, T. S. Jaakkola, R. A. Young, and D. K. Gifford. Computational discovery of gene modules and regulatory networks. *Nature Biotechnology*, 21:1337–42, Nov 2003.

H. Bussemaker, H. Li, and E. Siggia. Regulatory element detection using correlation with expression. *Nature Genetics*, 27(2):167–171, 2001.

J. M. Cherry, C. Ball, K. Dolinski, S. Dwight, M. Harris, J. C. Matese, G. Sherlock, G. Binkley, H. Jin, S. Weng, and D. Botstein. *Saccharomyces* genome database. http://genome-www.stanford.edu/Saccharomyces/, 2001.

The Gene Ontology Consortium. Gene ontology: tool for the unification of biology. *Nature Genetics*, 25(1):25–29, 2000.

G. F. Cooper and E. Herskovits. A Bayesian method for the induction of probabilistic networks from data. *Machine Learning*, 9:309–347, 1992.

S. Dasgupta. Learning polytrees. In *Proceedings of the Fifteenth Annual Conference on Uncertainty in Artificial Intelligence (UAI)*. 1999.

R. O. Duda and P. E. Hart. *Pattern Classification and Scene Analysis*. John Wiley & Sons, New York, 1973.

N. Friedman, M. Linial, I. Nachman, and D. Pe’er. Using Bayesian networks to analyze expression data. *Journal of Computational Biology*, 7:601–620, 2000.

A. P. Gasch, P. T. Spellman, C. M. Kao, O. Carmel-Harel, M. B. Eisen, G. Storz, D. Botstein, and P. O. Brown. Genomic expression program in the response of yeast cells to environmental changes. *Mol. Bio. Cell*, 11:4241–4257, 2000.

C. T. Harbison, D. B. Gordon, T. I. Lee, N. J. Rinaldi, Macisaac K. D., T. W. Danford, N. M. Hannett, J. B. Tagne, D. B. Reynolds, J. Yoo, E. G. Jennings, J. Zeitlinger, D. K. Pokholok, M. Kellis, P. A. Rolfe, K. T. Takusagawa, E. S. Lander, D. K. Gifford, E. Fraenkel, and R. A. Young. Transcriptional regulatory code of a eukaryotic genome. *Nature*, 431:99–104, 2004.

A. J. Hartemink, D. K. Gifford, T. S. Jaakkola, and R. A. Young. Combining location and expression data for principled discovery of genetic regulatory networks. In *Pacific Symposium on Biocomputing*, pages 437–449, 2002.

L. H. Hartwell, J. J. Hopfield, S. Leibler, and Murray A. W. From molecular to modular cell biology. *Nature*, 2, Dec 1999.

D. Heckerman. A tutorial on learning with Bayesian networks. In M. I. Jordan, editor, *Learning in Graphical Models*. Kluwer, Dordrecht, Netherlands, 1998.

D. Heckerman, D. Geiger, and D. M. Chickering. Learning Bayesian networks: The combination of knowledge and statistical data. In *Proceedings of the Tenth Annual Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 293–301. 1994.

T. R. Hughes, M. J. Marton, A. R. Jones, C. J. Roberts, R. Stoughton, C. D. Armour, H. A. Bennett, E. Coffey, H. Dai, Y. D. He, M. J. Kidd, A. M. King, M. R. Meyer, D. Slade, P. Y. Lum, S. B. Stepaniants, D. D. Shoemaker, D. Gachotte, K. Chakraburtty, J. Simon, M. Bard, and S. H. Friend. Functional discovery via a compendium of expression profiles. *Cell*, 102(1):109–26, 2000.

J. Ihmels, G. Friedlander, S. Bergmann, O. Sarig, Y. Ziv, and N. Barkai. Revealing modular organization in the yeast transcriptional network. *Nature Genetics*, 31:370–7, Aug 2002.

S. Imoto, T. Goto, and S. Miyano. Estimation of genetic networks and functional structures between genes by using bayesian networks and nonparametric regression. In *Pacific Symposium on Biocomputing*, pages 185–186, 2002.

T. I. Lee, N. J. Rinaldi, F. Robert, D. T. Odom, Z. Bar-Joseph, G. K. Gerber, N. M. Hannett, C. T. Harbison, C. M. Thompson, I. Simon, J. Zeitlinger, E. G. Jennings, H. L. Murray, D. B. Gordon, B. Ren, J. J. Wyrick, J. B. Tagne, T. L. Volkert, E. Fraenkel, D. K. Gifford, and R. A. Young. Transcriptional regulatory networks in *saccharomyces cerevisiae*. *Science*, 298:799–804, 2002.

B. Lehmann, D. Lehmann, and N. Nisan. Combinatorial auctions with decreasing marginal utilities. In *ACM Conference on Electronic Commerce*, pages 18–28, 2001.

A. Martinez-Antonio and J. Collado-Vides. Identifying global regulators in transcriptional regulatory networks in bacteria. *Current Opinion Microbioly*, 6(5):482–9, 2003.

I. M. Ong, J. D. Glasner, and D. Page. Modelling regulatory pathways in *e. coli* from time series expression profiles. *Bioinformatics*, 18 Suppl 1:S241–S248, 2002.

D. Pe'er, A. Regev, G. Elidan, and N. Friedman. Inferring subnetworks from perturbed expression profiles. *Bioinformatics*, 17 Suppl 1:S215–S224, 2001.

D. Pe'er, A. Regev, and A. Tanay. Minreg: Inferring an active regulator set. *Bioinformatics*, 18 Suppl 1:S258–S267, 2002.

G. R. Sambrano, G. Chandy, S. Choi, D. Decamp, R. Hsueh, K. M. Lin, D. Mock, N. O'Rourke, T. Roach, H. Shu, B. Sinkovits, M. Verghese, and H. Bourne. Unravelling the signal-transduction network in b lymphocytes. *Nature*, 420:708–710, Dec 2002.

E. Segal, Y. Barash, I. Simon, N. Friedman, and D. Koller. From promoter sequence to expression: A probabilistic framekwork. In *RECOMB*, pages 263–272. 2002.

E. Segal, D. Pe'er, A. Regev, D. Koller, and N. Friedman. Learning module networks. *Journal of Machine Learning Research*, 6:557–588, April 2005.

E. Segal, M. Shapira, A. Regev, D. Pe'er, D. Botstein, D. Koller, and N. Friedman. Module networks: identifying regulatory modules and their condition specific regulators from gene expression data. *Nature Genetics*, 34:166 – 176, 2003.

S. S. Shen-Orr, R. Milo, S. Mangan, and U. Alon. Network motifs in the transcriptional regulation network of escherichia coli. *Nature Genetics*, 31(1):64–8, 2002.

S. Tavazoie, J. D. Hughes, M. J. Campbell, R. J. Cho, and G. M. Church. Systematic determination of genetic network architecture. *Nat Genet*, 22(3):281–5, 1999.

L. F. Wu, T. R. Hughes, A. P. Davierwala, M. D. Robinson, R. Stoughton, and S. J. Altschuler. Large-scale prediction of *saccharomyces cerevisiae* gene function using overlapping transcriptional clusters. *Nature Genetics*, 31:255–265, 2002.

C. Yoo, V. Thorsson, and G. F. Cooper. Discovery of causal relationships in a gene-regulation pathway from a mixture of experimental and observational dna microarray data. In *Pacific Symposium on Biocomputing*, pages 498–509, 2002.