A Method For Reconstructing Transcriptional Regulatory Networks

Thesis for the M.Sc. Degree

Submitted to the Scientific Council of
The Weizmann Institute of Science
Rehovot 76100, Israel

By

Shiri Margel

Carried Out Under the Supervision of
Professor Eytan Domany
July 4, 2005

Acknowledgments

I would like to start my thesis by thanking the people who have assisted me along the road. To my supervisor Professor Eytan Domany for his scientific guidance and fascinating ideas. His thoughts and remarks always aided, and captured my convictions in a more scientific manner. Eytan, with his charismatic personality has managed to combine in his lab both scientific excellence and a welcoming, warm environment. This distinct atmosphere is inspiring, and stimulates achievement.

I want to thank my friends, the members of the 'Domany's group'. I especially want to thank Or Zuk, for his fruitful thoughts and ideas; Yuval Tabach for biological insights; Liat Ein-Dor, Tal Shay, Hila Benjamin, Michal Sheffer, Noam Shental, Assif Yitzhaky, Dafna Tsafrir, Libi Herzberg, Hila Gal and Yael Garten for helping me throughout this time.

I want to thank Dr. Eyal Shimony and Dr. Michael Milyavsky for devoting the time to help me with my work.

My deep gratitude goes to my family for their love and support. A special thanks goes to my parents for always being there for me, to my husband, David, for his encouragement and understanding, and to my daughter, Hadar, for making everything worthwhile.

Abstract

In order to understand processes in life, in particular processes in biology, it is essential to uncover the dependencies between the elements of the process. Meaning, to understand which elements 'control' other elements and/or interact with them. Often, this knowledge is not known and we can gain information about the process from data, i.e. from measurements of the values of the elements. One of the popular ways of representing these dependencies (and independencies) between elements is using Bayesian Networks. A Bayesian Network is a statistical model that represents dependencies and independencies between a group of random variables. In recent years this model has been used in many fields to learn processes from data. In particular, this model was extensively used to learn biological processes.

In the first part of my work I show that it is possible to learn the structure of a Bayesian Network from data, with high probability, provided the data is large enough. Moreover, I derived upper bounds on this sample complexity. These bounds are represented as a function of the Bayesian Network parameters and as a function of the probability with which we want to learn the correct structure. The upper bounds I received suggest that in order to learn with high probability the correct network structure from data, one needs a very large number of realizations (examples, measurements), which rarely exist in biological data. Thus, it is very problematic to learn the exact structure of a Bayesian Network that represents a biological process.

Due to the limitations of learning the exact structure of Bayesian Networks from data, I looked for a different method to learn networks from data. This method is described in the second part of my thesis: I developed a heuristic algorithm for reconstructing transcriptional networks from both gene expression data and sequence data. In the center of this algorithm is a classifier that uses gene expression data in order to predict whether a certain transcription factor regulates a certain gene. Then, if the classifier confirmed that the transcription factor regulates the gene, the algorithm continues by verifying that the transcription factor binding motif appears in the promoter of the gene. I applied this algorithm on a transcriptional network related to Smooth Muscle Cells differentiation and discovered both previously well known transcriptional connections, as well as new ones.

Contents

Part I

On the sample Complexity of Learning the structure of Bayesian Networks from Data

1 Introduction

Bayesian Networks (BNs) are models for representing knowledge under uncertainty. Their compact representation and modularity have made them very popular, and today they are used in various fields such as AI, Expert systems, Economics and Computational Biology. The key idea of BNs is the explicit representation of (in)dependencies in the distribution of measured data. These independencies are exploited to compactly represent numerical parameters and for efficient inference.

1.1 What is a Bayesian Network ?

A BN is a model that represents the dependencies and independencies between a group of random variables. It is composed of two components. The first component, called the *structure* of the BN, is a directed acyclic graph (DAG) that represents the (in)dependencies among the random variables. If X and Y belong to the group of the random variables, we will say that X is a *parent* of Y if there is an edge from X to Y in the DAG. We would say that X is independent of all variables that are not its parents, given his parents. The second component, called the *parametrization* of the BN, consists of numerical parameters, which encode the conditional probability of each variable, given the values of its parents in the network. An example of a BN can be seen in figure 1 ([1]).

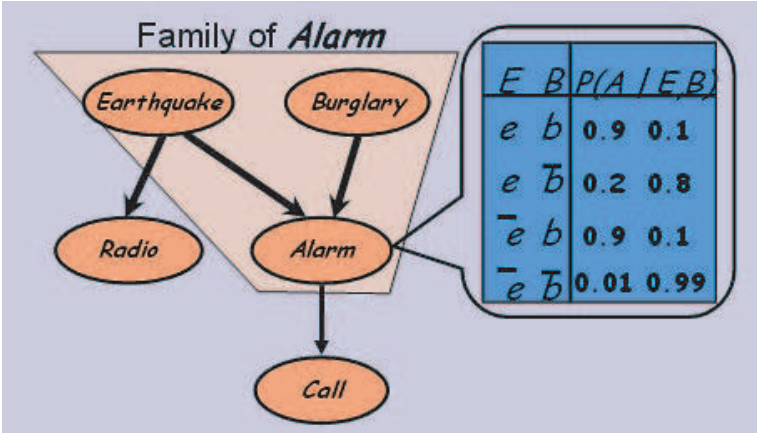

Figure 1: An example of a simple BN. There are fiveboolean variables in this BN: Earthquake, Burglary, Radio, Alarm and Call. We can see that both Earthquake and Burglary are the parents of Alarm, Earthquake is also the parent of Radio and Alarm is the parent of Call. Since Alarm has two parents that can take 2 values each, and since Alarm itself can also take two values, it has four parameters, indicating the probability that it will take the value one, given each of its parents configurations. In each case the probability that its value is zero is one minus the probability that its value is one, given the specific parents configuration.

1.2 Bayesian Networks in Biology

In recent years, many works in bioinformatics use BNs in order to learn various biological processes from various data types.

Friedman *et. al.* [2] used BNs in order to learn interactions between pairs of genes in *Saccharomyces Cerevisiae* from gene expression data. In this work the BN represents connections between genes. The variables of the BN are the genes from the gene expression data chip. The space of the values that each variable gets is specified using two approaches. The first one is the discrete approach, in which each variable gets three values: under-expressed, normal and over expressed (compared to a control, that could define the normal value as the average expression value of each gene over all experiments or the average expression value of all gene expression measurements in each experiment). The second one is the linear gaussian approach, in which they learned a linear regression model for each variable given its parents. Since the data contained only a few dozen samples and thousands of variables it is not possible to learn the exact structure of the network from it (in order to learn the exact structure of a BN with thousands of variables one needs much more samples - described in details in my work), thus, they chose to learn only small features of the network. Each feature is a pair of genes that are related in one of two ways. The first relation, called Markov relation, contains pairs of genes which have an edge between them in the BN or which are parents of a third gene in the BN. The second relation, called order relation, contains pairs of genes with a

path between them in the BN. In order to understand to what extent the data supports a certain feature they used the *bootstrap* method, in which they generated "pertubed" versions of their data (smaller data sets that contain part of the samples of the full data, chosen randomly) and learn the BN structure from them. The confidence of the feature is the percent of times it appeared in the BN structures learned from the "pertubed" data versions. In order to learn the BN structure from data they developed an algorithm that reduces the graphs search space, by identifying a relatively small number of candidate parents for each gene, and restricting the search space to networks with these parents. The search algorithm they developed is an iterative one, where at each stage one adds more candidate parents to each gene, if they contribute to the score of the gene.

Peer *et. al.* [3] have extended the previous work in a few directions. First of all, they added a third feature type, called separator, which contains three genes, such that one of them (the separator) separates the other two in the BN (or: the two genes are independent given the separator). Second, they tried to reconstruct significant subnetworks of the BN, built of several Markov pairs. In order to do so, they constructed a graph, where 2 variables (genes) in the graph are connected if they are related with a Markov relation (with confidence higher than a certain threshold). They took each non trivial component from this graph (contains more than three variables), called it a subnetwork, and tried to expand this subnetwork by adding variables that are Markov related to the variables in the subnetwork, with a confidence that passed a certain threshold, lower than the previous one. They also used another way for building these subnetworks, in which they measured the probability that a certain subnetwork gets a better confidence levels, than the ones it got. If this probability is low, they assumed that this group of variables is indeed a subnetwork. They also used this method for finding features and subnetworks of related genes in *Saccharomyces Cerevisiae* data.

Segal *et. al.* [4] learned transcriptional modules of *Saccharomyces Cerevisiae*, using gene expression data and promoter sequences of the genes in the data chip. A transcriptional module contains a set of genes that are co-regulated in a subset of the experiments in the data, and share a common motif profile (the same transcription factors binding sites in their promoters). In order to do so they used BNs in a different way than Friedman *et. al.* or Peer *et. al.*. Each gene in the data chip had its own BN, with a known structure. Each such BN contained four groups of variables. The first group contained a variable for each nucleotide in the promoter of the gene; the second group contained a variable for each motif (they work with a fixed number of transcription factor binding

site motifs); the third group contained one variable, representing the transcription module that the gene belongs to; the fourth group contained a variable for each expression value of each experiment in the data. The variables in the first group were the parents of the variables in the second group. The variables in the second group were the parents of the variable in the third group. The variable in the third group was the parent of the variables in the fourth group. There were no other direct connections in the network. The values of the variables in the first and fourth groups are observed from the data. The variable in the third group contains the module to which the gene belongs. The variables in the second group are boolean variables, indicating wether each motif belongs to the gene module. These variables are not observed from the data, and were learned using the *Expectation Maximization (EM)* algorithm.

Another work that used BNs in order to learn a biological network is the work of Sachs *et. al.* [5]. They took 11 key phosphorylated proteins in human T-cell signalling, and tried to learn the signalling causality map of them. In order to do so they generated multivariate flow cytometry data (for more information about this method see [6], [7]). This data was collected after a series of stimulatory cues and inhibitory interventions. They made flow cytometry measurements of the 11 proteins. Because these measurements were on a single cell, they managed to have thousands of data points. Now, they used the data in order to learn the structure of the BN with the 11 genes as variables. They used the same learning algorithm as Peer *et. al.* and discovered most of the known connections between these 11 proteins and 2 new connections. In this case there were much more data realizations and not so many variables as in the BN of Friedman *et. al.* and Peer *et. al.*, thus the chance to learn a structure that is close to the real one is higher.

1.3 Sample Complexity of Learning Bayesian Networks - Background

Often there is no way of building BNs according to expert knowledge. In these cases one needs to learn a BN from data, i.e. from a sample of N realizations (values taken by the network's variables). It is very important to assess the sample size one needs in order to learn a BN that approximates the true network, with the desired precision. The learning task can be described in the following way: given a database of independently drawn instances, construct a BN that best describes the joint distribution in the database. There are several ways to measure the quality of the learning procedure. One way

is evaluating the difference between the learned distribution and the real distribution (the distribution from which the samples are taken, called also the target or correct distribution); this difference can be evaluated by calculating, for example, the *Kullblack-Leibler* distance between the distributions (explained later). Another way is evaluating the difference between the learned structure and the correct structure (the structure from which the samples are taken, called also the target or correct structure). One estimate of this difference is the number of different edges between the real and learned models.

The subject of sample complexity (how many samples suffice in order to learn the right target) of BNs has drawn some attention in the past decade.

Some of the works in this field used the *PAC (probably approximately correct) framework*: consider a concept class C defined over a set of instances X of length n and a learner L using hypothesis space H. C is PAC learnable by L using H if for all $c \in C$, distributions D over X, $\varepsilon > 0$, $\delta \in (0, 1)$, learner L will, with probability at least $(1 - \delta)$, output a hypothesis $h \in H$, s.t. $error_D(h) < \varepsilon$, in time that is polynomial in $1/\delta$, $1/\varepsilon$, c and $size(c)$ [8].

Dasgupta *et. al.*[9] used the PAC framework in order to obtain sample complexity bounds for learning the conditional probability functions of BNs with known structure, with and without hidden variables (variables whose existence we are not aware of). He showed that for a Boolean BN (each variable can get 0 or 1) with n observed variables, and with at most k parents to each variable, one can find a distribution function with distance at most ε from the target distribution with probability $> 1 - \delta$, provided the learning set contains at least

$$O(\frac{288n^2 2^k}{\varepsilon^2} \log^2 (1 + \frac{3n}{\varepsilon}) \log \frac{18n^2 2^k \log (1 + \frac{3n}{\varepsilon})}{\varepsilon\delta})$$

samples.

Friedman & Yakhini [10] have studied the problem of learning the distribution of a discrete BN (each variable gets a finite number of values) with an unknown structure, and without hidden variables. They showed that the number of samples needed to learn a distribution which is not more then ε far from the target distribution, with probability $> 1 - \delta$, is

$$O((\frac{1}{\varepsilon})^{\frac{4}{3}} \log \frac{1}{\varepsilon} \log \frac{1}{\delta} \log \log \frac{1}{\delta}) \quad (1)$$

They used the MDL score with the BIC penalty (se eq. 6) to score the model selection (the different BNs).

Both works used the *Kullblack-Leibler* (*KL*) distance [11] (also termed relative entropy) between the original and learned models to measure the quality of the approximation. This distance is defined by

$$KL(P||Q) = \sum_x P(x) \log \frac{P(x)}{Q(x)}$$

where P and Q are probability distributions.

It can be seen that when comparing the dependency of the last two bounds on δ and ε, Friedman & Yakhini result is better than the result of Dasgupta *et. al.*; moreover, in Friedman & Yakhini work the structure was not known and there was no restriction on the parents' number. However, Dasgupta *et. al.* worked with hidden variables and they specified the dependency of the bound on the BN parameters (k and n).

A different criterion for estimating the distance between BNs was presented by Greiner *et. al.* [12], whose main argument is that the learned BNs are typically used for answering queries, therefore, in this context, a more appropriate measure is the performance of the learned model (compared to the real one) when answering queries. They assumed that the structure is known and presented a way of learning a distribution with the optimal performance, given examples of both the underlying distribution and of the queries that will be posed.

Haughton [13], [14] showed that the problem of learning the correct structure of a curve exponential model (see section 4) is *asymptotically consistent*, using the MDL score with the BIC penalty (see eq. 6). Simply stated, an (asymptotically) consistent scoring criterion is one that - in the limit, as the number of samples grows to infinity - prefers (almost surely) the model containing the fewest number of parameters that can represent the generative distribution exactly [15]. Moreover, Haughton gave bounds on the probability (δ) to learn a wrong model structure. A model is considered to be wrong if it doesn't contain the probability of the real model (models that underfit the data), or if it contains the probability of the real model, but it contains more parameters then the real model (models that overfit the data, see section 2.1). She has shown that if a certain model doesn't contain the probability of the real model then the probability to learn this wrong's model structure is $O(e^{-N\alpha})$, where $N \to \infty$ is the number of samples and $\alpha > 0$ is some constant. If a certain model does contain the probability of the real model (but is not minimal) then the probability to learn this wrong model structure is $O(N^{-k})$, where $N \to \infty$ and $k > 0$ is some constant. In the second case the dependency of k on the model was also specified. Using the results of the work of Geiger *et. al.* [16] that showed that discrete BNs without hidden variables are curved

exponential models, we can conclude that Haughton's findings are true for discrete BNs without hidden variables.

1.4 Our work outline and main results

In our study, we prove that boolean BNs without hidden variables are asymptotically consistent, using the MDL score (see eq. 6). We use a different, more directed, approach to do so, compared to Haughton work. Moreover, we derive an upper bound on the sample complexity of learning the correct structure of a boolean BN, without hidden variables.

We were motivated by the experimental observation that often, two BNs with extremely close distributions (in KL or L_p sense) have totally different structures. Thus, having samples that exceed the bound that was obtained by Yakhini *et. al.* ([10]) is not enough when one wants to learn the correct structure.

To illustrate this, we present in figure 2 an example demonstrating the difficulty of learning the correct structure of a small BN. On the basis of such examples it seems that learning the correct structure is a much harder task (and hence requires a larger number of samples) than approximating the underlying distribution.

For simplicity, we assume throughout our work that the *ordering* of the variables is known, and w.l.o.g. is given by the identity permutation, so $(i,j) \in E \Rightarrow i < j$.

Moreover, we first show that when the samples do not contain noise, i.e., the samples follow the exact distribution of the real BN (a detailed explanation of this concept is given at the beginning of section 3), the MDL score is consistent for each penalty $\Psi(N) = o(N)$. Next, we showed that the bound on the sample complexity (N) that is needed in order to learn the real network structure using the MDL score with the BIC penalty is

$$N \geq \frac{(|G^*| - n)}{\gamma^3} \cdot \frac{1}{\varepsilon^2} \log(\frac{(|G^*| - n)}{\gamma^3} \cdot \frac{1}{\varepsilon^2}) =$$

$$O(2^n (\frac{1}{\gamma})^3 (\frac{1}{\varepsilon})^2 (\log \frac{1}{\varepsilon} + \log \frac{1}{\gamma} + n))$$

where G^* is the real graph, $|G^*|$ is the number of parameters in G^*, n is the number of variables in G^*, γ is a lower bound on the probability of the BNs and ε is a lower bound on the distance between parameters in the BNs (see section 3.1).

Next, we looked at noisy samples, since it is clear that in reality the sampling process introduces noise. We first showed that learning the correct structure is asymptotically consistent, using the MDL score with any penalty $\Psi(N) = o(N)$, when looking only on

BNs that don't contain the probability of the real BN. Moreover, we showed that we will learn the right structure with probability $> 1-\delta$ using the MDL score with the BIC penalty for sample number

$$N \geq \max\{\frac{2(|G^*|-n)}{\gamma^3}\cdot\frac{1}{\varepsilon^2}\log(\frac{2(|G^*|-n)}{\gamma^3}\cdot\frac{1}{\varepsilon^2}), 512[\log(\frac{1}{\delta})+\log(6)+n\log(2)+\log\binom{n}{2}](\frac{1}{\gamma})^{n+6}(\frac{1}{\varepsilon})^4\} =$$

$$O((\frac{1}{\gamma})^n(\frac{1}{\varepsilon})^4(n+\log(\frac{1}{\delta})))$$

(see section 3.2.1).

Then, we showed that learning the correct structure is asymptotically consistent, using the MDL score with penalty $\Psi(N) = N^{-\eta}$, $\eta \in (\frac{1}{2}, 1)$, when considering only BNs that contain the probability of the real BN, and are larger (in the sense of parameters number) than the real BN. Moreover, we showed that using the MDL score with the above penalty function we will learn the right structure with probability $> 1-\delta$ for sample number

$$N \geq [162n^2(\frac{1}{\gamma})^n\{\log(\frac{1}{\delta})+\binom{n}{2}\log(2)+2\log(n)+\log(6)\}]^{\frac{1}{2\eta-1}} =$$

$$O([n^2(\frac{1}{\gamma})^n(\log(\frac{1}{\delta})+n^2)]^{\frac{1}{2\eta-1}}) \Rightarrow$$

(see section 3.2.2).

In this work, we address to the simplest and strictest problem of learning the correct BN structure, and do not consider the case of learning a structure that approximates the original one. Clearly, a relaxation of this requirement (for example allowing a certain fraction of mis-learned edges), will lead to lower sample complexity. On the other hand, when measuring the distance between the real and learned structures we restrict ourselves to situations where the learner knows in advance the ordering of the variables, which determines the directionality of every possible edge. We assume also that the learner is computationally unbounded and learns by scoring all possible models exhaustively and selecting the best scoring one. Obviously, due to the two above assumptions, the number of samples needed by real-life algorithms for structure learning could be higher.

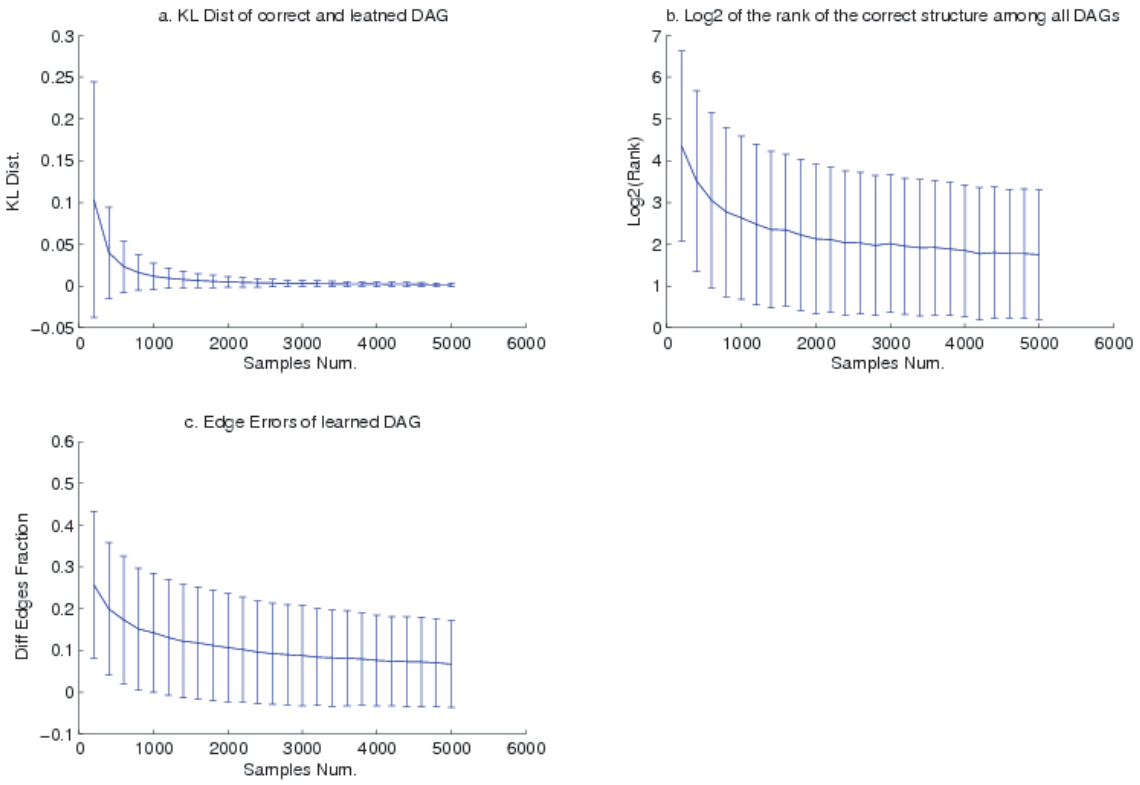

Figure 2: Learning a small Bayesian Network. We generated randomly 1000 4-node BNs, by choosing uniformly one of the 543 4-nodes labelled DAGs, and for each such DAG, choosing uniformly the value for each parameter. For each BN we generated 5000 realizations. The x axis represents the sample size N, i.e. the number of realizations, selected at random out of the 5000, that were used for learning. For each network and sample we learned a model from the data by exhaustively scoring all 543 labelled DAGs of size 4, and selected the best scoring model with the MDL score with BIC penalty (see eq. 6). All three plots show mean $\pm$ one standard deviation of the measured quantity. The y-axes contains: (a) the KL distance between the original and learned model. It nicely approaches zero. (b) the log2 of the rank of the correct structure among all 543 DAGs. We can see that the rank doesn't converge to zero after 5000 realizations, and that its convergence is much slower than the KL distance. If the correct BN structure receives the same score as other structures, it will get the lowest rank among them (the rank is according to score - the highest score graph is ranked 1 ($\log 2 = 0$), the lowest score graph is ranked 543 ($\log 2 = 9.08$)). (c) # edges that appear in the correct graph and not in the learned graph or vice versa / # total edges in the four node clique (6). In this case we don't take into account the direction of the edges (meaning, we only check if during the learning procedure the skeleton of the correct structure was learned, which is easier than learning the correct structure). We can see that the average of this fraction is not zero after the 5000 realizations.

2 Preliminaries and Definitions

2.1 Bayesian Network Preliminaries and Definitions

Let $X_1, \ldots, X_n$ be random binary variables, with joint probability distribution P. In general, uppercase will denote random variables, and lowercase will denote their realizations. We will sometimes skip the latter, so for example $Pr(x_1, \ldots, x_n) = Pr(X_1 = x_1, \ldots X_n = x_n)$. Formally, a BN $B = (G, \Theta)$ is defined by the graph G and the parameters Θ. Here

$G = (V, E)$ is an n-vertex DAG (the structure) which represents (in)dependencies between the X_i's, such that a variable is independent of its non-descenders , given its parents [17]. $\Theta = \{\Theta_i\}_{i=1}^n$ (the parameterization) specifies conditional probabilities, such that $\Theta_{i|pa_G(i)} = Pr(X_i = 1|Pa_G(i) = pa_G(i))$, where $Pa_G(i)$ are the parents of X_i in the graph G. The set of parents of X_j excluding X_i is denoted $Pa_G(j \setminus i)$, and its values by $pa_G(j \setminus i)$. Keeping the other parent's values and setting $X_i = k$ will be denoted by $Pa_G(j \setminus i : k)$, for $k = 0, 1$.

The joint probability distribution associated with B is denoted P_B, and satisfies :

$$P_B(X_1, \ldots, X_n) = \prod_{i=1}^{n} Pr(X_i|X_1, \ldots, X_{i-1}) =$$

$$\prod_{i=1}^{n} Pr(X_i|Pa_G(i)) = \prod_{i=1}^{n} \Theta_{i|pa_G(i)}^{x_i}(1 - \Theta_{i|pa_G(i)})^{1-x_i} \tag{2}$$

We assume our BN is "non-deterministic", which simply means that $\Theta_{i|pa_G(i)} \in (0, 1)\ \forall i, pa_G(i)$, and we define the 'determinism' of B as the smallest deviation of its parameters from 0 or 1:

$$\gamma = \gamma(B) = \min_{i,pa_G(i)} \{\Theta_{i|pa_G(i)}, (1 - \Theta_{i|pa_G(i)})\} \tag{3}$$

We mark the space of all (labelled !) *DAGs* on n vertices by Λ_n. The graph dimension $|G|$, is defined as the number of parameters needed to specify P_B when the structure is G.

G is called *minimal* (with respect to P_B) if there is no $B' = (G', \Theta')$, with $|G'| < |G|$, and $P_{B'} = P_B$ (see figure 3). We assume data is generated from a *minimal* BN denoted $B^* = (G^*, \Theta^*)$ with $G^* = (E^*, V^*)$. We refer to G^* as the 'correct' structure, and our purpose is to recover it from the data.

Two BNs are *equivalent* if the set of distributions that can be represented using the structure of one of them is identical to the set of distributions that can be represented using the structure of the other [18]. An *equivalence class* of G is the group of DAGs that are equivalent to G. There is a known criterion for two graphs to be in the same equivalence class given by Pearl & Verma [17]: the graphs should have the same *skeleton* (the skeleton of a DAG is its structure without the direction on the edges) and the same *V-structure* (the set of all ordered triplets of vertices in G = (V, E) that form a v-shape, meaning, two of these vertices are the parents of the third vertex, and neither of the parents is a parent of the other) in order to be equivalent.

The clique satisfying the ordering $\{1, .., n\}$ is denoted C^*. A BN associated with C^* is denoted by $B_{C^*} = (C^*, \Theta^{C^*})$. The clique with the edge (i, j) removed is denoted $C^*_{i,j} = C^* \setminus \{(i, j)\}$.

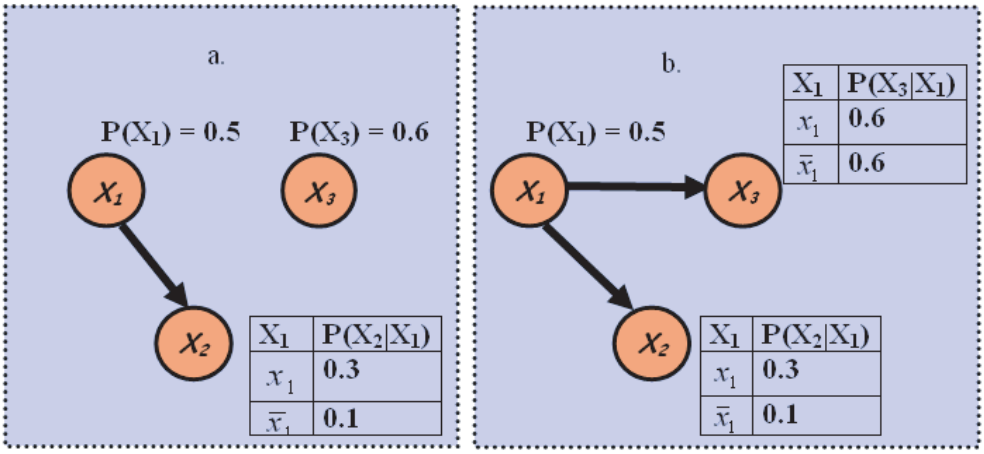

Figure 3: An example of a minimal BN. In (a) we can see a minimal BN. In (b) we can see a BN that represents the same distribution as the one in (a), but has more parameters than the BN in (a), and thus is not minimal.

We denote by N the number of realizations (also referred to as samples) in the learning set. The learning set will be denoted by D_N. The samples will be denoted by $x^{(i)}, i = 1, .., N$, with $x^{(i)} = (x_1^{(i)}, .., x_n^{(i)})$, so $x_j^{(i)}$ is the value of the r.v. X_j in the i-th sample. We assume the samples are identically distributed (i.i.d.), with $x^{(i)} \sim P_{B^*}$. The sample distribution $\widehat{P}_N$ is simply the frequency function of the sample $x^{(i)}$, given by

$$\hat{P}_N(x) = \frac{1}{N} \sum_{i=1}^{N} 1_{x^{(i)}=x}$$

2.1.1 Learning the structure of a Bayesian Network from Data

In order to learn the correct structure of a BN ($B = (G, \Theta)$) from data (D_N) one needs to calculate $P(G|D_N)$ for all possible n-vertex DAGs G (where n is the variables number of the correct BN), and the graph G that maximizes $P(G|D_N)$ is the learned structure. Using *Bayes rule* we get that:

$$P(G|D_N) = \frac{P(D_N|G)P(G)}{P(D_N)}$$

Since $P(D_N)$ doesn't depend on G it is enough to calculate $P(D_N|G)$ and know $P(G)$ (the *prior* on each G) in order to learn which G maximizes $P(G|D_N)$. In the Bayesian approach to model selection one starts with a *prior* distribution over all possible Bayesian networks. Given the learning set D_N, one can evaluate the *posterior* probability of each network structure. The score received in this way is called Bayesian score and is calculated in the following way:

$$Bayesian\_Score(G|D_N) = \log P(D_N|G) + \log P(G)$$

where $P(D_N|G)$, called the *marginal likelihood*, is calculated by:

$$P(D_N|G) = \int_\theta P(D_N|G,\theta)P(\theta|G)d\theta$$

where $P(D_N|G,\theta) = \sum_{i=1}^{N} P(x^{(i)}|G,\theta)$ is the likelihood function (see eq. 4), $P(\theta|G)$ is the prior over the parameters and $P(G)$ is the prior over the structures.

Although, in principle, this approach is appealing, in practice it is often impossible to evaluate the integral over all possible parameters, when the number of parameters is large.

Thus, we chose to use a different approach that approximates the marginal likelihood.

For any BN $B = (G, \Theta)$, the log-likelihood of the data has the form :

$$LL(D_N|G,\Theta) = \sum_{i=1}^{N}\sum_{j=1}^{n} logPr(x_j^{(i)}|pa_G^{(i)}(j)) =$$

$$\sum_{j=1}^{n}\sum_{i=1}^{N}(x_j^{(i)} log\Theta_{j|pa_G^{(i)}(j)} + (1-x_j^{(i)})log(1-\Theta_{j|pa_G^{(i)}(j)})) \tag{4}$$

Given G, the maximal likelihood parameterization $\hat{\Theta} = \hat{\Theta}(G)$, is simply given by sample probability, so $\hat{\Theta}_{i|pa_G(i)} = \hat{P}_N(X_i = 1|pa_G(i))$. Therefore, the maximum likelihood of the data given G, denoted by $LL_N(G)$, is given by :

$$LL_N(G) := LL(D_N|G,\hat{\Theta}) =$$

$$-N\sum_{j=1}^{n}\sum_{pa_G(j)} \hat{P}_N(pa_G(j))h_b(\hat{\Theta}_{j|pa_G(j)}) =$$

$$-N\sum_{j=1}^{n} H_{\hat{P}_N}(X_j|Pa_G(j)) \tag{5}$$

The definitions of h_b and H_P can be seen in eq. 7 and in eq. 9.

Clearly, the log-likelihood is not a useful score for comparing structures, since adding edges to the graph always improves the likelihood, which makes the complete graph the highest scoring graph. A common strategy to cope with this problem of overfitting, is based on the *Minimal Description Length (MDL)* principle, introduced in [19]. The MDL score 'penalizes' complex models, thus giving a trade-off between data-fitting and model complexity. A fairly general MDL score is described in [10], and is given by :

$$S_N(G) = LL_N(G) - |G|\Psi(N) \tag{6}$$

where $\Psi(N)$ is some *penalty function*.

So, in order to learn the correct structure, we scan all n-vertex DAGs G and calculate the score for each G. The graph that maximizes this score is the learned graph.

The penalty function $\Psi(N) = \frac{1}{2}\log N$, also termed the *Bayesian Information Criterion (BIC)*, is particulary interesting because it turns out that $S_N(G)$ with $\Psi(N) = \frac{1}{2}\log N$ is an $O(1)$ approximation of the Bayesian score $Bayesian\_Score(G|D_N)$ when there are no hidden variables [20], [16].

2.2 Information Theory

We briefly recall some Information Theory formulas, which were used in this work [21].

Let X, Y, Z, T be discrete random variables and P be a probability function.

The binary entropy function is:

$$h_b(p) = -p\log p - (1-p)\log(1-p) \tag{7}$$

The entropy of a variable X with respect to P is:

$$H_P(X) = -\sum_{X=x} P(x)\log P(x) \tag{8}$$

The conditional entropy of a variable X given Y with respect to P is:

$$H_P(X|Y) = -\sum_{X=x}\sum_{Y=y} P(x,y)\log P(x|y) \tag{9}$$

The mutual information between X and Y is:

$$I_P(X,Y) = H_P(Y) - H_P(Y|X) \tag{10}$$

and the conditional mutual information of two variables is:

$$I_P(X,Y|Z) = H_P(Y|Z) - H_P(Y|X,Z) \tag{11}$$

In the context of BNs we say that

$$I_{P_B}(X,Y|Z) = H_{P_{B_{(X,Y)}}}(Y|Z) - H_{P_B}(Y|X,Z) \tag{12}$$

where X and Y are variables in the BN, Z is a set of variables in the BN and the difference between $B_{(X,Y)}$ and B is that the edge (X,Y) is in the DAG of B and not in the DAG of $B_{(X,Y)}$.

3 Learning the correct structure

Following the framework of Friedman & Yakhini ([10]), we derive bounds on the sample complexity of learning the correct structure in two steps. First, we assume that the observed distribution is *ideal*, that is $\hat{P}_N = P_{B^*}$. In order to produce an ideal learning set of size N, we calculate the expected number of 1s (0s) for each variable, based on P_{B^*}, and then make sure that this will be the exact number of 1s (0s) in the learning set (or if not possible, a value which is closest to that number). We start with the variables with no parents and make sure that the number of 1s (0s) in the learning set represents their probability according to P_{B^*}. For example, if X_1 has no parents, $N = 100$ and $P_{B^*}(X_1 = 1) = 0.4$, then we will prepare 40 samples with $X_1 = 1$ and 60 samples with $X_1 = 0$. Next, we continue to variables with parents from the previous group only, meaning, variables with no grandparents. For each such variable we calculate the number of 1s (0s) given each configuration of its parents, that will exactly represent P_{B^*}, and make sure that the learning set contains exactly this number of 1s (0s) (or if not possible, a value which is closest to that number). For example, if the only parent of X_2 is X_1, and $P_{B^*}(X_2 = 1|X_1 = 1) = 0.25$, then in 10 if the 40 samples with $X_1 = 1$, X_2 will also be 1, in the rest of the 30 samples X_2 will be 0. In the same way we continue to the group of variables with parents from the 2 previous groups only, and so on.

Here the number of samples needed is a result of a trade-off between the likelihood term and the penalty term in S_N (see eq. 6), since the number of samples affects both the likelihood term, in the N that multiplies the conditional entropy, and the penalty term. The conditional entropy in the likelihood term is the same for any N in the ideal case (eq. 5).

Next, we study the effect of sampling noise on the sample complexity. Using Chernoff bounds and the union bound, we show that with high probability $\hat{P}_N$ is close to P_{B^*}, allowing us to bound the probability that the correct structure does not get the maximal score.

In order to simplify our discussion, we assume throughout our work that the *ordering* of the variables is known. Note that for a given ordering there is at most one DAG in each equivalence class [17].

Before deriving our results, we will look deeper into the notion of minimality. A BN is *minimal* (see figure 3), if every edge (i, j) gives extra information on X_j given all its other parents. More formally, let $B = (G, \Theta)$ a BN with $G = (V, E)$, then G is *minimal*

with respect to P_B if and only if

$$I_{P_B}(X_i, X_j | Pa_G(j \setminus i)) > 0,\ \forall (i,j) \in E. \tag{13}$$

The next lemma shows that the previous conditional mutual information stays positive when we add edges to G.

Lemma 1 *Let $B = (G, \Theta)$ a BN with $G = (V, E)$. If G is minimal and $\Theta \in (0,1)$, then $\forall (i,j) \in E$, X_i gives extra information on X_j given all its other parents in the clique C^*. Meaning,*

$$I_{P_{B_{C^*}}}(X_i, X_j | Pa_{C^*_{i,j}}(j)) > 0,\ \forall (i,j) \in E$$

Proof Let $(i,j) \in E$. From the minimality of G, we have some parents configuration $pa_G(j)$ such that $\Theta_{j|pa_G(j\setminus i:0)} \neq \Theta_{j|pa_G(j\setminus i:1)}$, otherwise, eq. 13 is not satisfied because X_i doesn't add any new information on X_j. We know that X_j is conditionally independent of $\{X_1, \ldots, X_{j-1}\} \setminus Pa_G(j)$ given $Pa_G(j)$, so , for $k = 0, 1$:

$$P_{B_{C^*}}(X_j = 1 | Pa_{C^*}(j \setminus i : k)) = \Theta_{j|pa_G(j\setminus i:k)} \tag{14}$$

and $P_{B_{C^*}}(pa_{C^*}(j \setminus i)) \geq \min_x P_{B_{C^*}}(x) \geq \gamma(B^*)^n > 0$.

We know that

$$I_{P_{B_{C^*}}}(X_i, X_j | Pa_{C^*_{i,j}}(j)) =$$

$$\sum_{pa_{C^*_{i,j}}(j)} \{P_{B_{C^*}}(pa_{C^*_{i,j}}(j)) h_b(\Theta_{j|pa_{C^*_{i,j}}(j)}) - \sum_{k=0}^{1} P_{B_{C^*}}(pa_{C^*}(j \setminus i : k)) h_b(\Theta_{j|pa_{C^*}(j\setminus i:k)})\} \tag{15}$$

Due to the concavity of h_b and to the fact that

$$P_{B_{C^*}}(pa_{C^*_{i,j}}(j)) \Theta_{j|pa_{C^*_{i,j}}(j)} = \sum_{k=0}^{1} P_{B_{C^*}}(pa_{C^*}(j \setminus i : k)) \Theta_{j|pa_{C^*}(j\setminus i:k)} \tag{16}$$

we can say that each element in the sum of eq. 15 is non negative. Since we know that we have some parents configuration $pa_{C^*}(j)$ such that $P_{B_{C^*}}(X_j = 1 | Pa_{C^*}(j \setminus i : 0)) \neq P_{B_{C^*}}(X_j = 1 | Pa_{C^*}(j \setminus i : 1))$ and the probabilities in the last inequality are both positive, we can be sure that at least the element with these probabilities in eq. 15 sum is positive, and so $I_{P_{B_{C^*}}}(X_i, X_j | Pa_{C^*_{i,j}}(j)) > 0$ ■

3.1 Deriving Bounds in the Ideal Case

As a first step we will show that the MDL score is consistent in the ideal case. Meaning, that we can find N_0, s.t. if $N > N_0$ we will surely learn the correct structure of G^*. In this case we will learn the right structure with probability 1 for large enough N, since the sample is ideal. Thus, we show here that the MDL score is consistent, which is a stronger requirement than asymptotical consistency (see proposition 1). Then, we will show the dependency of this upper bound on the BN of G^* (see proposition 2).

3.1.1 Consistency of the MDL Score with respect to BN structures

The following lemmas are true for the ideal case.

Lemma 2 *If $S_N(G) > S_N(G^*)$, then $|G| < |G^*|$.*

Proof Since $\hat{P}_N = P_{B^*}$, we get $LL_N(G^*) = LL_N(C^*)$. But clearly, $LL_N(G) \leq LL_N(C^*), \forall G$, therefore :

$$S_N(G^*) = LL_N(G^*) - |G^*|\Psi(N) < S_N(G) = LL_N(G) - |G|\Psi(N) \leq LL_N(C^*) - |G|\Psi(N) =$$

$$LL_N(G^*) - |G|\Psi(N) \tag{17}$$

So $|G| < |G^*|$. ■

Lemma 3

$$S_N(G^*) \geq \max_{(i,j)\in E^*} LL_N(C^*_{i,j}) - n\Psi(N) \Rightarrow$$
$$S_N(G^*) = \max_G S_N(G) \tag{18}$$

Proof Assume, negatively that $\exists G \in \Lambda_n$, $S_N(G^*) < S_N(G)$. From lemma 2 we get $|G| < |G^*|$, therefore we have some edge $(i,j) \in E^* \setminus E$. Thus, $G \subseteq C^*_{i,j}$ and $LL_N(G) \leq LL_N(C^*_{i,j})$. From the lemma's condition we know that

$$S_N(G^*) \geq LL_N(C^*_{i,j}) - n\Psi(N)$$

and since $|G| \geq n$ we get

$$LL_N(C^*_{i,j}) - n\Psi(N) \geq LL_N(G) - n\Psi(N) \geq LL_N(G) - |G|\Psi(N) = S_N(G)$$

Meaning, we got $S_N(G^*) \geq S_N(G)$, which is a contradiction. ■

The likelihood in the *ideal* case is given by :

$$LL_N(C^*) = LL_N(G^*) = -N \sum_{i=1}^{n} H_{P_{B_{G^*}}}(X_i|Pa_{G^*}(i)) \tag{19}$$

Removing the edge (i, j), the likelihood of $C^*_{i,j}$ is :

$$LL_N(C^*_{i,j}) = -N[\sum_{i=1, i\neq j}^{n} H_{P_{B_{C^*}}}(X_i|Pa_{C^*}(i)) +$$

$$H_{P_{B_{C^*_{i,j}}}}(X_j|Pa_{C^*_{i,j}}(i))] \tag{20}$$

Using lemma 3 we will bound the number of samples needed to learn G^* correctly by comparing its score to the log likelihood of all the $C^*_{i,j}$ minus the penalty function multiplied with n. We demand that $\forall(i, j) \in E^*$:

$$S_N(G^*) = LL_N(G^*) - |G^*|\Psi(N) =$$

$$LL_N(C^*) - |G^*|\Psi(N) \geq LL_N(C^*_{i,j}) - n\Psi(N)$$

Or $\forall(i, j) \in E^*$:

$$LL_N(C^*) - LL_N(C^*_{i,j}) \geq (|G^*| - n)\Psi(N) \tag{21}$$

We will now compute the difference:

$$LL_N(C^*) - LL_N(C^*_{i,j}) =$$

$$N[H_{P_{B_{C^*_{i,j}}}}(X_j|Pa_{C^*_{i,j}}(j)) - H_{P_{B_{C^*}}}(X_j|Pa_{C^*}(j))] =$$

$$NI_{P_{C^*}}(X_i, X_j|Pa_{C^*_{i,j}}(i)) \tag{22}$$

Using eq. 21 and eq. 22, we get:

Proposition 1 *In the ideal case :*

$$\frac{\Psi(N)}{N} \leq \frac{1}{|G^*| - n} \min_{(i,j)\in E^*} I_{P_{B_{C^*}}}(X_i, X_j|Pa_{C^*_{i,j}}(j))$$

$$\Rightarrow S_N(G^*) = \max_G S_N(G) \tag{23}$$

From lemma 1 we know that $\min_{(i,j)\in E^*} I_{P_{C^*}}(X_i, X_j|Pa_{C^*_{i,j}}(j))$ is positive. So the right hand side is a positive constant depending on B^*. Thus, for any reasonable penalty $\Psi(N) = o(N)$, there is some constant $N_0 = N_0(B^*)$ such that if there is no sampling noise, the correct structure is learned for $N \geq N_0 \Rightarrow$ the MDL score is consistent.

3.1.2 Dependence of the Sample Complexity on the correct BN

We showed in proposition 1 that there exists an upper bound $N_0 = N_0(B^*)$ on the sample size needed to learn the right BN structure from data. We now want to investigate the dependency of the sample complexity upper bound on B^*. Note that the upper bound (N_0^*) we specify in this section is not tight, meaning, $N_0^* > N_0$.

We assume that $\exists \varepsilon > 0$ s.t. $\forall j, pa_{C^*}(j)$

$$|\Theta_{j|pa_{C^*}(j\backslash i:1)} - \Theta_{j|pa_{C^*}(j\backslash i:0)}| > \varepsilon \tag{24}$$

In order to find this N_0^* we want to find a lower bound L on $\min_{(i,j)\in E^*} I_{P_{B_{C^*}}}(X_i, X_j|Pa_{C^*_{i,j}}(j))$ because from proposition 1 if $\frac{\Psi(N)}{N} \leq \frac{1}{|G^*|-n}L$ then $S_N(G^*) = \max_G S_N(G)$.

To do so, we define $\forall j, pa_{C^*}(j)$ two variables: $\Theta^{\varepsilon}_{j|pa_{C^*}(j\backslash i:0)}$ and $\Theta^{\varepsilon}_{j|pa_{C^*}(j\backslash i:1)}$ with the following characters: First of all, the difference between them is ε, Meaning

$$|\Theta^{\varepsilon}_{j|pa_{C^*}(j\backslash i:1)} - \Theta^{\varepsilon}_{j|pa_{C^*}(j\backslash i:0)}| = \varepsilon \tag{25}$$

Second, eq. 16 is also true with $\Theta^{\varepsilon}_{j|pa_{C^*}(j\backslash i:0)}$ and $\Theta^{\varepsilon}_{j|pa_{C^*}(j\backslash i:1)}$ instead of $\Theta_{j|pa_{C^*}(j\backslash i:0)}$ and $\Theta_{j|pa_{C^*}(j\backslash i:1)}$ respectively. Meaning, $P_{B_{C^*_{i,j}}}(x_j|pa_{C^*_{i,j}(j)})$ is also the weighted average of these two variables, with the same weights as it is the weighted average of $\Theta_{j|pa_{C^*}(j\backslash i:0)}$ and $\Theta_{j|pa_{C^*}(j\backslash i:1)}$:

$$P_{B_{C^*_{i,j}}}(pa_{C^*_{i,j}(j)})P_{B_{C^*_{i,j}}}(x_j|pa_{C^*_{i,j}(j)}) = \sum_{k=0}^{1} P_{B_{C^*}}(pa_{C^*(j\backslash i:k)})\Theta^{\varepsilon}_{j|pa_{C^*}(j\backslash i:k)} \tag{26}$$

Denote

$$I^{\varepsilon}_{P_{B_{C^*}}}(X_i, X_j|Pa_{C^*_{i,j}}(j)) =$$

$$\sum_{pa_{C^*_{i,j}}(j)} \{P_{B_{C^*_{i,j}}}(pa_{C^*_{i,j}}(j))h_b(\Theta_{j|pa_{C^*_{i,j}}(j)}) - \sum_{k=0}^{1} P_{B_{C^*}}(Pa_{C^*}(j \setminus i : k))h_b(\Theta^{\varepsilon}_{j|pa_{C^*}(j\backslash i:k)})\} \tag{27}$$

So the difference between $I_{P_{B_{C^*}}}(X_i, X_j|Pa_{C^*_{i,j}}(j))$ and $I^{\varepsilon}_{P_{B_{C^*}}}(X_i, X_j|Pa_{C^*_{i,j}}(j))$ is that in $I^{\varepsilon}_{P_{B_{C^*}}}(X_i, X_j|Pa_{C^*_{i,j}}(j))$ we use $\forall j, pa_{C^*}(j)$ $\Theta^{\varepsilon}_{j|pa_{C^*}(j\backslash i:k)}$ instead of $\Theta_{j|pa_{C^*}(j\backslash i:k)}$, $k = 0, 1$.

Lemma 4

$$I_{P_{B_{C^*}}}(X_i, X_j|Pa_{C^*_{i,j}}(j)) \geq I^{\varepsilon}_{P_{B_{C^*}}}(X_i, X_j|Pa_{C^*_{i,j}}(j)) \tag{28}$$

Proof From the concavity of h_b, from the fact that $\Theta_{j|pa_{C^*_{i,j}}(j)}$ is the same weighted average of both $\Theta_{j|pa_{C^*}(j\setminus i:1)}$ and $\Theta_{j|pa_{C^*}(j\setminus i:0)}$ and of $\Theta^{\varepsilon}_{j|pa_{C^*}(j\setminus i:1)}$ and $\Theta^{\varepsilon}_{j|pa_{C^*}(j\setminus i:0)}$ (eq. 16 and 26 respectively), and from the fact that the difference between $\Theta_{j|pa_{C^*}(j\setminus i:1)}$ and $\Theta_{j|pa_{C^*}(j\setminus i:0)}$ is larger from the difference between $\Theta^{\varepsilon}_{j|pa_{C^*}(j\setminus i:1)}$ and $\Theta^{\varepsilon}_{j|pa_{C^*}(j\setminus i:0)}$ (eq. 24 and 25 respectively) we get that

$$\sum_{k=0}^{1} P_{B_{C^*}}(Pa_{C^*}(j\setminus i:k))h_b(\Theta^{\varepsilon}_{j|pa_{C^*}(j\setminus i:k)}) \geq \sum_{k=0}^{1} P_{B_{C^*}}(Pa_{C^*}(j\setminus i:k))h_b(\Theta_{j|pa_{C^*}(j\setminus i:k)}) \tag{29}$$

and thus

$$I_{P_{B_{C^*}}}(X_i, X_j|Pa_{C^*_{i,j}}(j)) =$$

$$\sum_{pa_{C^*_{i,j}}(j)} \{P_{B_{C^*_{i,j}}}(pa_{C^*_{i,j}}(j))h_b(\Theta_{j|pa_{C^*_{i,j}}(j)}) - \sum_{k=0}^{1} P_{B_{C^*}}(Pa_{C^*}(j\setminus i:k))h_b(\Theta_{j|pa_{C^*}(j\setminus i:k)})\} \geq$$

$$\sum_{pa_{C^*_{i,j}}(j)} \{P_{B_{C^*_{i,j}}}(pa_{C^*_{i,j}}(j))h_b(\Theta_{j|pa_{C^*_{i,j}}(j)}) - \sum_{k=0}^{1} P_{B_{C^*}}(Pa_{C^*}(j\setminus i:k))h_b(\Theta^{\varepsilon}_{j|pa_{C^*}(j\setminus i:k)})\} =$$

$$I^{\varepsilon}_{P_{B_{C^*}}}(X_i, X_j|Pa_{C^*_{i,j}}(j)) \tag{30}$$

Which gives us the required result ■

Lemma 5 *If $\varepsilon > 0$ and $\gamma = \gamma(C^*) < \frac{1}{3}$ (see 3), then*

$$\lim_{\varepsilon\to 0} \frac{I_{P_{B_{C^*}}}(X_i, X_j|Pa_{C^*_{i,j}}(j))}{\varepsilon^2} \geq \gamma^3 \tag{31}$$

Proof From lemma 4 it is enough to show that

$$\lim_{\varepsilon\to 0} \frac{I^{\varepsilon}_{P_{B_{C^*}}}(X_i, X_j|Pa_{C^*_{i,j}}(j))}{\varepsilon^2} \geq \gamma^3$$

Denote

$$\varepsilon_{pa_{C^*_{i,j}}(j)} = \Theta_{j|pa_{C^*_{i,j}}(j)} - \Theta^{\varepsilon}_{j|pa_{C^*}(j\setminus i:1)} \tag{32}$$

from 26 and from the fact that

$$P_{B_{C^*_{i,j}}}(pa_{C^*_{i,j}}(j)) = \sum_{k=0}^{1} P_{B_{C^*}}(pa_{C^*}(j\setminus i:k)) \tag{33}$$

we get

$$\Theta_{j|pa_{C^*_{i,j}}(j)} - \Theta^{\varepsilon}_{j|pa_{C^*}(j\setminus i:0)} = -\varepsilon_{pa_{C^*_{i,j}}(j)} \frac{P_{B_{C^*}}(pa_{C^*}(j\setminus i:1))}{P_{B_{C^*}}(pa_{C^*}(j\setminus i:0))} \tag{34}$$

Combining the last two equations with eq. 25 we get that

$$|\varepsilon_{pa_{C^*_{i,j}(j)}}| = \frac{\varepsilon}{[1 + \frac{P_{B_{C^*}}(pa_{C^*}(j\backslash i{:}1))}{P_{B_{C^*}}(pa_{C^*}(j\backslash i{:}0))}]} \tag{35}$$

Since $\gamma < \frac{1}{3}$ we get that $\forall pa_{C^*}(j)$

$$\gamma\varepsilon \leq |\varepsilon_{pa_{C^*_{i,j}(j)}}| \leq (1-\gamma)\varepsilon \tag{36}$$

and thus we can say that $\varepsilon \to 0 \Rightarrow \forall pa_{C^*}(j), \varepsilon_{pa_{C^*_{i,j}(j)}} \to 0$.

Since $\forall pa_{C^*}(j), \varepsilon_{pa_{C^*_{i,j}(j)}} \to 0$ we can execute Taylor expansion for $\log(\Theta_{j|pa_{C^*}(j\backslash i:k)})$ and for $\log(1-\Theta_{j|pa_{C^*}(j\backslash i:k)})$ around $\Theta_{j|pa_{C^*_{i,j}}(j)}$ $\forall j, pa_{C^*_{i,j}}(j)$ and get:

$$I^{\varepsilon}_{P_{B_{C^*}}}(X_i, X_j | Pa_{C^*_{i,j}}(j)) =$$

$$\sum_{pa_{C^*_{i,j}}(j)} \{P_{B_{C^*_{i,j}}}(pa_{C^*_{i,j}}(j)) h_b(\Theta_{j|pa_{C^*_{i,j}}(j)}) - \sum_{k=0}^{1} P_{B_{C^*}}(Pa_{C^*}(j \setminus i:k)) h_b(\Theta^{\varepsilon}_{j|pa_{C^*}(j\backslash i:k)})\} =$$

$$\sum_{x_j, pa_{C^*_{i,j}}(j)} \{\frac{P_{B_{C^*_{i,j}}}(pa_{C^*_{i,j}}(j)) P_{B_{C^*}}(pa_{C^*}(j \setminus i:1))}{2P_{B_{C^*}}(pa_{C^*}(j \setminus i:0))\Theta_{j|pa_{C^*_{i,j}}(j)}(1-\Theta_{j|pa_{C^*_{i,j}}(j)})} \varepsilon^2_{pa_{C^*_{i,j}(j)}} + O(\varepsilon^3_{pa_{C^*_{i,j}(j)}})\} \tag{37}$$

and from 3 and 36 we can say that

$$\lim_{\varepsilon \to 0} \frac{I^{\varepsilon}_{P_{B_{C^*}}}(X_i, X_j | Pa_{C^*_{i,j}}(j))}{\varepsilon^2} \geq \frac{2^{j-2}\gamma^3}{(1-\gamma)^3} \geq \gamma^3 \tag{38}$$

■

Proposition 2 *Let $\varepsilon_0 > 0$ small enough. For $\Psi(N) = \frac{\log(N)}{2}$ and $0 < \varepsilon < \varepsilon_0$ we get:*

$$N \geq N^*_0 = \frac{(|G^*| - n)}{\gamma^3} \cdot (\frac{1}{\varepsilon})^2 \log(\frac{(|G^*| - n)}{\gamma^3} \cdot (\frac{1}{\varepsilon})^2) =$$

$$O(2^n (\frac{1}{\gamma})^3 (\frac{1}{\varepsilon})^2 (\log\frac{1}{\varepsilon} + \log\frac{1}{\gamma} + n))$$

$$\Rightarrow S_N(G^*) = \max_G S_N(G) \tag{39}$$

Proof From the combination of proposition 1 and lemma 5 we get for $\Psi(N) = \frac{\log(N)}{2}$ and $\varepsilon < \varepsilon_0$, for some $\varepsilon_0 > 0$ small enough:

$$\frac{\log(N)}{N} \leq \frac{1}{|G^*| - n} \cdot \gamma^3 \cdot \varepsilon^2$$

$$\Rightarrow S_N(G^*) = \max_G S_N(G) \tag{40}$$

and it can be seen that

$$N \geq \frac{(|G^*| - n)}{\gamma^3} \cdot (\frac{1}{\varepsilon})^2 \log(\frac{(|G^*| - n)}{\gamma^3} \cdot (\frac{1}{\varepsilon})^2) =$$

$$O(2^n(\frac{1}{\gamma})^3(\frac{1}{\varepsilon})^2(\log\frac{1}{\varepsilon} + \log\frac{1}{\gamma} + n)) \tag{41}$$

satisfies eq. 40 ■

3.2 Treatment of the Noisy Case

Clearly, when sampling noise is introduced, we can not guarantee that the correct (*minimal*) structure will attain the highest score. Since every sample has positive probability, there will always be samples whose distributions $\hat{P}_N$ are better represented by other structures. If, however, the statistical weight of such samples approaches zero when $N \to \infty$, the probability for an incorrect structure getting the highest score will be arbitrarily small for large enough N. Here we also assume that the *ordering* of the variables is known. In the noisy case we want to show that the MDL score is asymptotically consistent, meaning, we will show the existence of a function $N_0(B^*, \delta)$, such that for $N > N_0(B^*, \delta)$ the correct structure gets the highest score with prob. $> 1 - \delta$ (see proposition 3 and proposition 5). Then, we will investigate the dependency of the upper bound we got on the BN B and on δ (see proposition 4 and proposition 6). Here also, the bound we specify (N_0^*) is not tight ($N_0^* > N_0$).

In order to show that the entropy of a variable in the graph given its parents in the noisy case (sample entropy) is not far away from this entropy in the ideal case (true entropy) with high probability we use the following concentration lemma:

Lemma 6 *Let $B = (G, \Theta)$ be a BN. Take two disjoint subsets $S, T \subset \{X_1, .., X_n\}$ of r.v.s. Then if S=s and T=t:*
a. For any $\alpha \in (0, 1)$:

$$Pr(|\hat{P}_N(s) - P_B(s)| \leq \alpha P_B(s)) \geq 1 - 2e^{-\alpha^2 P_B(s) N/2} \tag{42}$$

b. For any $\alpha \in (0, 1/3)$:

$$Pr(|\hat{P}_N(t|s) - P_B(t|s)| \leq 3\alpha P_B(t|s)) \geq$$

$$1 - 4e^{-\alpha^2 P_B(s,t) N/2} \tag{43}$$

Proof a. Let $Y_N^s = \sum_{j=1}^N 1_{S^{(j)}=s}$ be the r.v. counting the number of samples in which the value of the S variables was s. Then $Y_N^s \sim Binomial(N, P_B(s))$. Using Chernoff bounds we get for $\alpha \in (0,1)$,

$$Pr((1-\alpha)P_B(s)N \leq Y_N^s \leq (1+\alpha)P_B(s)N) \geq$$

$$1 - 2e^{-\alpha^2 P_B(s)N/2} \tag{44}$$

Noting that $Y_N^s = N\hat{P}_N(s)$ gives the desired result.
b. Take $\alpha \in (0, 1/3)$. Applying (a) on the sets S and $S \bigcup T$ and the union bound, we get that:

$$P_B(s)[(1-\alpha)\hat{P}_N(t|s) - P_B(t|s)]$$

$$\leq \hat{P}_N(s,t) - P_B(s,t) \leq \alpha P_B(s,t) \tag{45}$$

With prob. $\geq 1 - 4e^{-\alpha^2 P_B(s,t)N/2}$. So :

$$(1+\alpha)P_B(t|s) \geq (1-\alpha)\hat{P}_N(t|s) \tag{46}$$

And finally :

$$\hat{P}_N(t|s) - P_B(t|s) \leq$$

$$(\frac{1+\alpha}{1-\alpha} - 1)P_B(t|s) \leq 3\alpha P_B(t|s) \tag{47}$$

w.p. $\geq 1 - 4e^{-\alpha^2 P_B(s,t)N/2}$.

Showing the other direction is done similarly. ■

A particular choice of our interest will be $T = X_j, S = Pa_G(j)$, which gives us :

$$Pr(|\hat{\Theta}_{j|pa_G(j)} - \Theta_{j|pa_G(j)}| \leq 3\alpha\Theta_{j|pa_G(j)}) \geq$$

$$1 - 4e^{-\alpha^2 P_B(pa_G(j))\Theta_{j|pa_G(j)}N/2} \tag{48}$$

And by symmetry $\Theta_{j|pa_G(j)}$ can be replaced by $1 - \Theta_{j|pa_G(j)}$ in the upper bound and the probability term of the above inequality.

We now bound the difference between the sample and the true entropies:

Lemma 7 *Let B be a BN. Then for any DAG G, $\exists \alpha_0 \in (0,1)$ such that for $\alpha < \alpha_0$:*

$$Pr(|H_{\hat{P}_N}(X_j|Pa_G(j)) - H_{P_B}(X_j|Pa_G(j))| < 4\alpha) \geq$$

$$1 - 6\sum_{pa_G(j)} e^{-\alpha^2 P_B(pa_G(j))\Theta_{j|pa_G(j)}N/2} \tag{49}$$

Proof

$$H_{\hat{P}_N}(X_j|Pa_G(j)) - H_{P_B}(X_j|Pa_G(j)) =$$
$$\sum_{pa_G(j)} [\hat{P}_N(pa_G(j))h_b(\hat{\Theta}_{j|pa(j)}) - P_B(pa_G(j))h_b(\Theta_{j|pa(j)})] \quad (50)$$

Using lemma 6 both parts, with $S = Pa_G(X_j), T = X_j$, and the union bound, we get:

$$\hat{P}_N(pa_G(j))h_b(\hat{\Theta}_{j|pa(j)}) - P_B(pa_G(j))h_b(\Theta_{j|pa(j)}) \geq$$
$$-\{(1-\alpha)P_B(pa_G(j))$$
$$[(1-\alpha)\Theta_{j|pa(j)}\log((1+\alpha)\Theta_{j|pa(j)}) +$$
$$(1-\alpha)(1-\Theta_{j|pa(j)})\log((1+\alpha)(1-\Theta_{j|pa(j)}))] -$$
$$P_B(pa_G(j))h_b(\Theta_{j|pa(j)})\} =$$
$$-P_B(pa_G(j))[(1-2\alpha)\alpha + 2\alpha h_b(\Theta_{j|pa(j)})] + O(\alpha^2) \quad (51)$$

With prob. $\geq 1 - 6e^{-\alpha^2 P_B(pa_G(j))\Theta_{j|pa_G(j)}N/2}$.

Take α_0 for which the $O(\alpha^2)$ term is bounded by α and for which $\max\{(1+\alpha)\Theta_{j|pa_G(j)}, (1+\alpha)(1-\Theta_{j|pa_G(j)})\} < 1$ for $\alpha < \alpha_0$. Since $h_b(\Theta_{j|pa(j)}) \leq 1$, we can get :

$$H_{\hat{P}_N}(X_j|Pa_G(j)) - H_{P_B}(X_j|Pa_G(j)) \geq$$
$$-4\alpha \sum_{pa_G(j)} P_B(pa_G(j)) = -4\alpha \, , \, \forall \alpha < \alpha_0 \quad (52)$$

w.p. $\geq 1 - 6\sum_{pa_G(j)} e^{-\alpha^2 P_B(pa_G(j))\Theta_{j|pa_G(j)}N/2}$. Showing the upper bound is done similarly. ■

Our approach for bounding $N_0(B^*, \delta)$ is by considering separately graphs G with $P_{B^*} \in B^* \setminus B$ and graphs G with $P_{B^*} \in B \cap B^*$

3.2.1 Graphs G with $P_{B^*} \in B^* \setminus B$

Lemma 8 *Lemma 3 is still valid, assuming that we require only* $S_N(G^*) \geq \max_{G, P_{B^*} \in B^* \setminus B} S_N(G)$.

Proof If $G^* \subset G$ then $P_{B^*} \in B \cap B^*$, otherwise we have some edge $(i,j) \in E^* \setminus E$, $G \subseteq C^*_{i,j}$ and the proof of lemma 3 remains the same. ■

The likelihood difference in eq. 22 is changed to:

$$LL_N(C^*) - LL_N(C^*_{i,j}) =$$
$$N[H_{\hat{P}_N}(X_j|Pa_{C^*_{i,j}}(j)) - H_{\hat{P}_N}(X_j|Pa_{C^*}(j))] \quad (53)$$

We will bound the difference between $LL_N(C^*) - LL_N(C^*_{i,j})$ in the noisy case and in the ideal case:

Lemma 9 *$\forall (i,j) \in E^*$, $\exists \alpha_0 \in (0,1)$ such that for any $\alpha < \alpha_0$:*

$$Pr(|[H_{\hat{P}_N}(X_j|Pa_{C^*_{i,j}}(j)) - H_{\hat{P}_N}(X_j|Pa_{C^*}(j))] -$$

$$I_{P_B}(X_i, X_j|Pa_{C^*_{i,j}}(j))| < 8\alpha) >$$

$$1 - 12 \cdot 2^{j-1} \max_{pa_{C^*}(j)} e^{-\alpha^2 P_B(pa_{C^*}(j))\Theta_{j|pa_{C^*}(j)}N/2} \tag{54}$$

Proof First apply lemma 7 on C^* and $C^*_{i,j}$. Then use the union bound and note that the Θ parameters in $C^*_{i,j}$ are weighted averages of the Θ's in C^*. Due to convexity of e^{-x}, of the two probabilities we subtract, the one related to C^* is therefore larger. Finally replace the sum by the maximum of the summands. ■

Now we can give an asymptotic bound on N:

Proposition 3 *For Graphs with $P_{B^*} \in B^* \setminus B$ and $\Psi(N) = o(N)$:*

$$\frac{(|G^*| - n)\Psi(N)}{N} \leq \frac{1}{2} \min_{(i,j)\in E} I_{P_{B^*}}(X_i, X_j|Pa_{C^*_{i,j}}) \tag{55}$$

and

$$\delta > \sum_{(i,j)\in E^*} 12 \cdot 2^{j-1} \max_{pa_{C^*}(j)} e^{-\alpha^2 P_B(pa_{C^*}(j))\Theta_{j|pa_{C^*}(j)}N/2} \tag{56}$$

for $\alpha = \frac{1}{16} \min_{(i,j)\in E^*} I_{P_{B_{C^*}}}(X_i, X_j|Pa_{C^*_{i,j}}(j))$

$$\Rightarrow Pr(S_N(G^*) \geq \max_{G, P_{B^*} \in B^* \setminus B} S_N(G)) > 1 - \delta \tag{57}$$

(Or: the MDL score with $\Psi(N) = o(N)$ is asymptotically consistent for graphs with $P_{B^*} \in B^* \setminus B$)

Proof Let $\delta \in (0,1)$ be given. In the same way as in the ideal case (proposition 1) we can say that:

$$\frac{(|G^*| - n)\Psi(N)}{N} \leq \min_{(i,j)\in E^*} \{H_{\widehat{P}_N}(X_j|Pa_{C^*_{i,j}}(j)) - H_{\widehat{P}_N}(X_j|Pa_{C^*}(j))\}$$

$$\Rightarrow S_N(G^*) = \max_{G, P_{B^*} \in B^* \setminus B} S_N(G) \tag{58}$$

From lemma 9 with $\alpha = \frac{1}{16} \min_{i,j} I_{P_{B_{C^*}}}(X_i, X_j|Pa_{C^*_{i,j}}(j))$ we get, using the union bound, that

$$Pr(\min_{(i,j)\in E^*} \{H_{\widehat{P}_N}(X_j|Pa_{C^*_{i,j}}(j)) - H_{\widehat{P}_N}(X_j|Pa_{C^*}(j))\} \geq \frac{1}{2} \min_{(i,j)\in E^*} I_{P_{B_{C^*}}}(X_i, X_j|Pa_{C^*_{i,j}})) \geq$$

$$1 - \sum_{(i,j)\in E^*} 12 \cdot 2^{j-1} \max_{pa_{C^*}(j)} e^{-\alpha^2 P_B(pa_{C^*}(j))\Theta_{j|pa_{C^*}(j)}N/2} \tag{59}$$

So from the last two equations we get that if

$$\frac{(|G^*| - n)\Psi(N)}{N} \leq \frac{1}{2} \min_{(i,j)\in E} I_{P_{B^*}}(X_i, X_j | Pa_{C^*_{i,j}}) \tag{60}$$

then

$$Pr(S_N(G^*) \geq \max_{G, P_{B^*} \in B^* \setminus B} S_N(G)) > 1 - \delta \tag{61}$$

for

$$\delta > \sum_{(i,j)\in E^*} 12 \cdot 2^{j-1} \max_{pa_{C^*}(j)} e^{-\alpha^2 P_B(pa_{C^*}(j))\Theta_{j|pa_{C^*}(j)}N/2} \tag{62}$$

Finally, we got two requirements for N: eq. 60 and eq. 62. Assuming $\Psi(N) = o(N)$, the first requirement (eq. 60) is satisfied for N larger than $N_0 = N_0(B^*)$, independent of δ. The second requirement is satisfied for $N > N_0(B^*, \delta)$. Thus, we can conclude that the MDL score is asymptotically consistent for G with $P_{B^*} \in B^* \setminus B$. ■

Proposition 4 *Let $\varepsilon_0 > 0$ small enough and $\delta > 0$. For $\Psi(N) = \frac{\log(N)}{2}$ and $0 < \varepsilon < \varepsilon_0$ we get:*

$$N \geq N_0^* =$$

$$\max\{\frac{2(|G^*| - n)}{\gamma^3} \cdot \frac{1}{\varepsilon^2} \log(\frac{2(|G^*| - n)}{\gamma^3} \cdot \frac{1}{\varepsilon^2}), 512[\log(\frac{1}{\delta}) + \log(6) + n\log(2) + \log\binom{n}{2}](\frac{1}{\gamma})^{n+6}(\frac{1}{\varepsilon})^4\} =$$

$$O((\frac{1}{\gamma})^n(\frac{1}{\varepsilon})^4(n + \log(\frac{1}{\delta})))$$

$$\Rightarrow S_N(G^*) = \max_{G, P_{B^*} \in B^* \setminus B} S_N(G) \tag{63}$$

Proof From proposition 2 we can conclude that for $\gamma = \gamma(C^*)$ the first requirement (eq. 60) is translated to

$$N \geq \frac{2(|G^*| - n)}{\gamma^3} \cdot \frac{1}{\varepsilon^2} \log(\frac{2(|G^*| - n)}{\gamma^3} \cdot \frac{1}{\varepsilon^2}) =$$

$$O(2^n(\frac{1}{\gamma})^3(\frac{1}{\varepsilon})^2(\log\frac{1}{\varepsilon} + \log\frac{1}{\gamma} + n)) \tag{64}$$

(the multiplication with 2 is performed because of the extra $\frac{1}{2}$ in eq. 60).

The second requirement (eq. 62) is satisfied for

$$\delta > 12 \cdot 2^{n-1} \cdot \binom{n}{2} \cdot e^{-\frac{\gamma^6 \varepsilon^4 \gamma^n N}{512}} \tag{65}$$

where $\gamma = \gamma(C^*)$. The last equation is translated to

$$N > 512[\log(\frac{1}{\delta}) + \log(6) + n\log(2) + \log\binom{n}{2}](\frac{1}{\gamma})^{n+6}(\frac{1}{\varepsilon})^4 =$$

$$O((\frac{1}{\gamma})^n(\frac{1}{\varepsilon})^4(\log(\frac{1}{\delta}))) \tag{66}$$

Combining the two requirements together results in

$$N_0^*(B_{C^*}, \delta) =$$

$$\max\{\frac{2(|G^*|-n)}{\gamma^3}\cdot\frac{1}{\varepsilon^2}\log(\frac{2(|G^*|-n)}{\gamma^3}\cdot\frac{1}{\varepsilon^2}), 512[\log(\frac{1}{\delta})+\log(6)+n\log(2)+\log\binom{n}{2}](\frac{1}{\gamma})^{n+6}(\frac{1}{\varepsilon})^4\} =$$

$$O((\frac{1}{\gamma})^n(\frac{1}{\varepsilon})^4(n+\log(\frac{1}{\delta}))) \tag{67}$$

■

3.2.2 Graphs G with $P_{B^*} \in B \cap B^*$

Now, we require that that $Pr(S_N(G^*) \geq \max_{G, P_{B^*} \in B \cap B^*} S_N(G)) \geq 1-\delta$. In this section lemma 3 is not valid, so we will use a different approach to show that the MDL score is asymptotically consistent and to find the dependency of the upper bound on B^* and on δ.

Notice that in this section $|G| > |G^*|$, because otherwise G is a smaller graph that contains P_{B^*}, which is a contradiction to the minimality of G^*.

Lemma 10

$$LL_N(G) - LL_N(G^*) \leq \Psi(N)\ \forall G, |G| > |G^*| \Rightarrow$$

$$S_N(G^*) \geq \max_{G, |G|>|G^*|} S_N(G) \tag{68}$$

Proof Suppose $LL_N(G) - LL_N(G^*) \leq \Psi(N)\ \forall G, |G| > |G^*|$.

$$S_N(G^*) =$$

$$LL_N(G^*) - |G^*|\Psi(N) \geq LL_N(G) - \Psi(N) - |G^*|\Psi(N) =$$

$$LL_N(G) - (|G^*|+1)\Psi(N) \geq LL_N(G) - |G|\Psi(N) =$$

$$S_N(G) \tag{69}$$

$$\Rightarrow S_N(G^*) \geq \max_{G, |G|>|G^*|} S_N(G)$$

■

Lemma 11 *Let* $\alpha \in (0,1),\ \forall G, |G| > |G^*|$:

$$Pr(LL_N(G) - LL_N(G^*) \leq$$

$$-N\sum_{i=1}^{n} H_{P_B}(X_i|Pa_G(i)) + N\sum_{i=1}^{n} H_{P_{B^*}}(X_i|Pa_{G^*}(i)) + 8\alpha nN) \geq$$

$$1 - 2^{\binom{n}{2}} \max_{G,|G|\geq|G^*|} n \max_{1\leq i\leq n} 6 \sum_{pa_G(i)} e^{-\alpha^2 P_B(pa_G(i))\Theta_{i|pa_G(i)}N/2} \tag{70}$$

Proof From lemma 7 we can conclude, using the union bound, that for $G \in \Lambda_n$

$$Pr(|LL_N(G) + N\sum_{i=1}^{n} H_{P_B}(X_i|Pa_G(i))| \leq 4\alpha nN) \geq$$

$$1 - n \max_{1\leq i\leq n} 6 \sum_{pa_G(i)} e^{-\alpha^2 P_B(pa_G(i))\Theta_{i|pa_G(i)}N/2} \tag{71}$$

Thus, we can say that $\forall G, |G| \geq |G^*|$:

$$Pr(LL_N(G) - LL_N(G^*) \leq$$

$$-N\sum_{i=1}^{n} H_{P_B}(X_i|Pa_G(i)) + N\sum_{i=1}^{n} H_{P_{B^*}}(X_i|Pa_{G^*}(i)) + 8\alpha nN) \geq$$

$$1 - 2^{\binom{n}{2}} \max_{G,|G|\geq|G^*|} n \max_{1\leq i\leq n} 6 \sum_{pa_G(i)} e^{-\alpha^2 P_B(pa_G(i))\Theta_{i|pa_G(i)}N/2} \tag{72}$$

■

Proposition 5 *For graphs with* $P_{B^*} \in B \cap B^*$ *and* $\Psi(N) = N^\eta$, $\eta \in (\frac{1}{2}, 1)$*:*

$$\delta \geq 2^{\binom{n}{2}} \max_{G,|G|\geq|G^*|} n \max_{1\leq i\leq n} 6 \sum_{pa_G(i)} e^{-\frac{N^{2\eta-1}P_B(pa_G(i))\Theta_{i|pa_G(i)}}{162n^2}} \Rightarrow$$

$$Pr(S_N(G^*) \geq \max_{G,P_B\in B\cap B^*} S_N(G)) > 1 - \delta \tag{73}$$

(Or: the MDL score with the above penalty function is asymptotically consistent for graphs with $P_{B^*} \in B^* \cap B$ *in the noisy case)*

Proof Let $\alpha = \frac{\Psi(N)}{9nN}$. Since we know that in the ideal case

$$LL_N(G^*) = LL_N(C^*) \geq LL_N(G),\ \forall G$$

Or

$$-N\sum_{i=1}^{n} H_{P_{B^*}}(X_i|Pa_{G^*}(i)) = -N\sum_{i=1}^{n} H_{P_{B_{C^*}}}(X_i|Pa_{C^*}(i)) \geq -N\sum_{i=1}^{n} H_{P_B}(X_i|Pa_G(i))$$

using the last equation and lemma 11, we get that $\forall G, |G| > |G^*|$:

$$Pr(LL_N(G) - LL_N(G^*) \leq 8\alpha nN) \geq$$

$$1 - 2^{\binom{n}{2}} \max_{G,|G|\geq|G^*|} n \max_{1\leq i\leq n} 6 \sum_{pa_G(i)} e^{-\alpha^2 P_B(pa_G(i))\Theta_{i|pa_G(i)}N/2} \tag{74}$$

(Since: $-N\sum_{i=1}^{n} H_{P_B}(X_i|Pa_G(i)) + N\sum_{i=1}^{n} H_{P_{B^*}}(X_i|Pa_{G^*}(i)) + 8\alpha nN \leq 8\alpha nN$) And for the specified α we get that $\forall G, |G| > |G^*|$:

$$Pr(LL_N(G) - LL_N(G^*) \leq \frac{8}{9}\Psi(N)) \geq$$

$$1 - 2^{\binom{n}{2}} \max_{G,|G|\geq|G^*|} n \max_{1\leq i\leq n} 6 \sum_{pa_G(i)} e^{-\frac{N^{2\eta-1}P_B(pa_G(i))\Theta_{i|pa_G(i)}}{162n^2}} \tag{75}$$

From lemma 10 we get that

$$Pr(S_N(G^*) \geq \max_{G,|G|>|G^*|} S_N(G)) > 1 - \delta$$

for

$$\delta \geq 2^{\binom{n}{2}} \max_{G,|G|\geq|G^*|} n \max_{1\leq i\leq n} 6 \sum_{pa_G(i)} e^{-\frac{N^{2\eta-1}P_B(pa_G(i))\Theta_{i|pa_G(i)}}{162n^2}} \tag{76}$$

and since the graphs G with $P_B \in B \cap B^*$ form a subset of the graphs G with $|G| > |G^*|$ we can say that for $\Psi(N) = N^\eta$, $\eta \in (\frac{1}{2}, 1)$ we get

$$\delta \geq 2^{\binom{n}{2}} \max_{G,|G|\geq|G^*|} n \max_{1\leq i\leq n} 6 \sum_{pa_G(i)} e^{-\frac{N^{2\eta-1}P_B(pa_G(i))\Theta_{i|pa_G(i)}}{162n^2}} \Rightarrow$$

$$Pr(S_N(G^*) \geq \max_{G,P_B\in B\cap B^*} S_N(G)) > 1 - \delta \tag{77}$$

The last equation is satisfied for $N > N_0(B^*, \delta)$, and thus we can say that for G with $P_{B^*} \in B \cap B^*$ the MDL score, with $\Psi(N) = N^\eta$, $\eta \in (\frac{1}{2}, 1)$, is asymptotically consistent ■

Proposition 6 *Let $\delta > 0$ and $\Psi(N) = N^\eta$, $\eta \in (\frac{1}{2}, 1)$*

$$N \geq N_0^*(B, \delta) = [162n^2(\frac{1}{\gamma})^n\{\log(\frac{1}{\delta}) + \binom{n}{2}\log(2) + 2\log(n) + \log(6)\}]^{\frac{1}{2\eta-1}} =$$

$$O([n^2(\frac{1}{\gamma})^n(\log(\frac{1}{\delta}) + n^2)]^{\frac{1}{2\eta-1}}) \Rightarrow$$

$$P(S_N(G^*) = \max_{G,P_{B^*}\in B\cap B^*} S_N(G)) > 1 - \delta \tag{78}$$

Proof In order to find out when eq. 77 is satisfied we denote:

$$\widetilde{\gamma} = \min_{G,|G|\geq|G^*|} \gamma(G)$$

From the definition of $\widetilde{\gamma}$ we can say that eq. 77 is satisfied for

$$\delta \geq 2^{\binom{n}{2}} \cdot n \cdot 6 \cdot (n-1) \cdot e^{-\frac{N^{2\eta-1}\gamma^n}{162n^2}} \tag{79}$$

And the N that satisfies the last equation is

$$N \geq [162n^2(\frac{1}{\gamma})^n\{\log(\frac{1}{\delta}) + \binom{n}{2}\log(2) + 2\log(n) + \log(6)\}]^{\frac{1}{2\eta-1}} \tag{80}$$

And we can say that

$$N_0^*(B,\delta) = [162n^2(\frac{1}{\gamma})^n\{\log(\frac{1}{\delta}) + \binom{n}{2}\log(2) + 2\log(n) + \log(6)\}]^{\frac{1}{2\eta-1}} =$$

$$O([n^2(\frac{1}{\gamma})^n(\log(\frac{1}{\delta}) + n^2)]^{\frac{1}{2\eta-1}}) \tag{81}$$

■

4 Comparing our results to prior works

As far as we know the only results that deal with the consistency of the MDL score, with respect to learning the correct BN structure, and give bounds to the sample complexity of learning the correct BN structure are derived from the combination of the works of Geiger *et. al.* [16] and Haughton [13], [14].

Before introducing these results a few definitions are in order.

An *exponential family* is a set of probability density functions which are given by

$$P(x|\eta) = e^{\langle\eta,t(x)\rangle - \Psi(\eta)} \tag{82}$$

where x is an element of a sample space χ with a dominating measure η and t(x) is a sufficient statistic defined on χ taking values in R^k with an inner product. The quantity $\Psi(\eta)$ is the normalization constant. Where η has k coordinates and when $P(x|\eta)$ can't be represented with a parameter vector smaller then k, then the representation is *minimal* and the *order* (dimension) of this family is k, and the parameters are called *natural parameters*. The *natural parameter space* is given by

$$N = \{\eta \in R^k | \int e^{\langle\eta,t(x)\rangle - \Psi(\eta)} d\eta(x) < \infty\} \tag{83}$$

A $diffeomorphism\ f : U \subset R^n \to R^m$ is a smooth (C^∞) $1-1$ function having a smooth inverse. A subset M of R^n is called a *k-dimensional smooth manifold* in R^n if for every point $x \in M$ there exists an open set U in R^n containing x and a diffeomorphism $f : U \cap M \to R^k$ [16].

A *curved exponential family* of dimension n is a subfamily of an exponential family of order k, such that its natural parameter space $N_0 \subset N$ is a n-dimensional smooth manifold in R^k.

Geiger *et. al.* [16] showed that BNs without hidden variables are curved exponential families. Haughton [13] proved that the MDL score with the BIC penalty is asymptotically consistent, when trying to learn the structure of a curved exponential model from data.

Translating this result to our context, meaning, to BNs without hidden variables, we get that Haughton showed that the MDL score with the BIC penalty is asymptotically consistent when trying to learn the correct BN structure from data.

When comparing these results to our work, we can see that for graphs G with $P_{B^*} \in B^* \setminus B$ our results are more general, since we prove that the MDL score is asymptotically consistent with any penalty $\Psi(N) = o(N)$. For graphs G with $P_{B^*} \in B \cap B^*$ we showed that the MDL score is asymptotically consistent with the penalty $\Psi(N) = N^\eta$, $\eta \in (\frac{1}{2}, 1)$. The results of Haughton are stronger than our results, since she proved that the MDL score is asymptotically consistent with the BIC penalty (which is a smaller penalty function), but since in this part we are talking about graphs G larger than G^* ($|G| > |G^*|$), and if the MDL score is asymptotically consistent with the BIC penalty, it will sure be asymptotically consistent with penalties larger than the BIC penalty.

In a later work of Haughton [14], she derived bounds on the size of the error of learning a wrong model when using the MDL score with the BIC penalty. Meaning, she bounded the probability of learning a wrong model instead of the correct one.

She got the following results (translated to BNs language):

1. For G with $P_{B^*} \in B^* \setminus B$ Haughton said that the probability to learn a wrong model is bounded by $O(e^{-N\alpha})$, for some $\alpha > 0$ as $N \to \infty$ (N is the samples number).

In order to compare this to our results of Section 3.2.1, we have to use the probabilities that Haughton got to understand the upper bound on the sample complexity needed to learn the correct BN with probability $> 1 - \delta$. Since there are at most $2^{\binom{n}{2}}$ binary ordered BNs we can say, using Haughton's results, that if

$$\frac{\delta}{2^{\binom{n}{2}}} = O(e^{-N\alpha}) \tag{84}$$

we will learn the correct BN with probability $> 1 - \delta$. The upper bound on the sample complexity is:

$$\frac{\log(\frac{1}{\delta}) + \log(a) + n^2}{\alpha} \tag{85}$$

where $a > 0$ is some constant. Meaning, if

$$N \geq \frac{\log(\frac{1}{\delta}) + \log(a) + n^2}{\alpha} \tag{86}$$

the right BN structure is learned with probability $> 1 - \delta$.

When comparing this to our result (eq. 67), we see that the dependency on δ is the same in our result and in Haughton result ($N \geq O(\log(\frac{1}{\delta}))$). Moreover, we gave upper bounds on α and a as a function of the BN parameters (γ, ε, $|G^*|$ and n).

2. For G with $P_{B^*} \in B^* \cap B$ Haughton showed that the probability to learn a wrong model is bounded by $O(N^{-k})$ for some $k > 0$, as $N \to \infty$. She also expressed k as a function of the size of the wrong learned model ($|G|$), the size of the correct model ($|G^*|$), and the the eigenvalues of the second derivative matrix $Q = (\frac{\partial^2 \Psi}{\partial \eta_i \partial \eta_j}(\eta))_{i,j=1,\ldots,s}$, where $\eta = (\eta_1, \ldots, \eta_s)$ is the dominating measure of the correct exponential family, $\Psi(\eta)$ is the normalization factor of the probability distribution of the correct exponential family and s is the dimension of the correct exponential family (see eq. 82).

In order to compare these results with ours of Section 3.2.2, we will present the results Haughton got in the way we did in 1:

If

$$\frac{\delta}{2^{\binom{n}{2}}} = O(N^{-k}) \tag{87}$$

we will learn the correct BN with probability $> 1 - \delta$. For

$$N \geq (\frac{2^{\binom{n}{2}} a}{\delta})^{\frac{1}{k}} \tag{88}$$

where $a > 0$ is some constant, we will learn the right BN structure with probability $> 1 - \delta$.

When comparing this with our results in eq. (81) we can see that in our result the dependency of N on δ is logarithmic, whereas Haughton gets a polynomial dependence of N on δ. Moreover, we specified the dependence of the sample complexity bound on the BN parameters. So for the penalty we work with ($\Psi(N) = N^\eta$, $\eta \in (\frac{1}{2}, 1)$) our result is better, but Haughton derived bounds on the sample complexity when working with the BIC penalty, and thus she gives more information than we do about the bounds on the sample complexity, when trying to learn the correct BN structure from data in this case.

5 Discussion

The goal of this work was to investigate the sample complexity (N) needed in order to learn the right BN structure from data. The scoring criterion we used in order to rank the different BN structures is the MDL score. As a first step we asked whether it is possible to learn the structure of the correct BN from data using the MDL score, for some N large enough, with high probability. Another way of asking this question is to ask whether the MDL score is asymptotically consistent. Only if the answer to the above question is yes, can we start talking about an upper bound on the sample complexity one needs in order to learn the right BN structure from data with high probability. Throughout our work, we assume that the ordering of the variables is known.

In order to simplify the discussion, we first assumed that the data has no sampling noise (ideal case). In this case we are talking about consistency of the MDL score and not asymptotical consistency, because we work with sampling without noise, and thus we can be sure that as $N \to \infty$ the right BN structure is learned. We proved that in this case the MDL score with penalty $\Psi(N) = o(N)$ is consistent. Then we derived an upper bound on the sample complexity needed to learn the right BN structure in the ideal case, using the BIC penalty. We got

$$\frac{(|G^*| - n)}{\gamma^3} \cdot \frac{1}{\varepsilon^2} \log(\frac{(|G^*| - n)}{\gamma^3} \cdot \frac{1}{\varepsilon^2}) =$$

$$O((\frac{1}{\gamma})^3(\frac{1}{\varepsilon})^2(\log\frac{1}{\varepsilon} + \log\frac{1}{\gamma} + n)) \tag{89}$$

as an upper bound on the sample complexity needed to learn the right BN structure in the ideal case. This bound depends on the BN parameters of G^*. We can see that the sample complexity needed to learn the right structure is decreases (increases) as ε increases (decreases), meaning, when the influence of a parent on its child is low (high), it will be hard (easy) to learn this edge in the graph. We can also see that the sample complexity needed to learn the right structure is decreases (increases) as γ increases (decreases). $\gamma > 0$ is the lowest distance of the parameters of C^* from 0 and 1 (the 'non-determinism' parameter, see eq. 3). This can be explained with the following example: Suppose X_1 and X_2 are two binary variables that are connected with an edge from X_1 to X_2. Let $P(X_1 = 1) = 0.0001$ (or even lower). Then in most of the samples X_1 will be 0 and it will seem as if $P(X_2 = 1|X_1 = 0) = P(X_2 = 1)$, thus, it will be very hard to learn the edge (X_1, X_2).

In the same way we derived the last bound, we can also derive an upper bound on the sample complexity needed to learn the right structure, using any other penalty

$\Psi(N) = o(N)$.

After understanding the behavior of the sample complexity bound in the ideal case we continued with the noisy case. In this case we divided the BNs space to two subsets. The first one includes BNs that don't contain the real probability. Meaning, $P_{B^*} \in B^* \setminus B$. It can be noticed that all graphs G with less parameters then G^* ($|G| < |G^*|$) are in this subset. The second one includes the rest of the BNs, meaning, it includes all BNs with $P_{B^*} \in B \cap B^*$. All the graphs in this subgroup have more parameters than the correct graph.

For the BNs in the first subgroup we showed the the MDL score is asymptotically consistent for any penalty $\Psi(N) = o(N)$. The upper bound we got on the sample complexity needed to learn the correct structure with probability $> 1-\delta$, using the BIC penalty is

$$N \geq \max\{\frac{2(|G^*|-n)}{\gamma^3}\cdot\frac{1}{\varepsilon^2}\log(\frac{2(|G^*|-n)}{\gamma^3}\cdot\frac{1}{\varepsilon^2}), 512[\log(\frac{1}{\delta})+\log(6)+n\log(2)+\log\binom{n}{2}](\frac{1}{\gamma})^{n+6}(\frac{1}{\varepsilon})^4\} =$$

$$O((\frac{1}{\gamma})^n(\frac{1}{\varepsilon})^4(n+\log(\frac{1}{\delta}))) \tag{90}$$

We can see that in this case the sample complexity needed to learn the right structure is decreases (increases) when ε / γ / δ increases (decreases). The relation between the upper bound and ε is the same as in the ideal case. The relation between the upper bound and γ and the upper bound and ε is much stronger here compared to the ideal case. This can be explained in one of two ways: the first reasonable explanation is that it is much harder to learn the BN structure from noisy data; the second explanation is that our bound in the noisy case is not tight. Probably, both of the explanations are correct. The relation between the upper bound and δ means that the higher (lower) the probability we want to learn the correct structure, the higher (lower) the sample complexity needed (which is obvious). In the same way we derived the last bound, we can also derive an upper bound for the sample complexity needed to learn the right structure, using any other penalty $\Psi(N) = o(N)$.

For the BNs in the second subgroup we showed the the MDL score is asymptotically consistent for penalty $\Psi(N) = N^\eta$, $\eta \in (\frac{1}{2}, 1)$. The upper bound we got on the sample complexity needed to learn the correct structure with probability $> 1-\delta$, using the above penalty is

$$[162n^2(\frac{1}{\gamma})^n\{\log(\frac{1}{\delta}) + \binom{n}{2}\log(2) + 2\log(n) + \log(6)\}]^{\frac{1}{2\eta-1}} =$$

$$O([n^2(\frac{1}{\gamma})^n(\log(\frac{1}{\delta}) + n^2)]^{\frac{1}{2\eta-1}}) \Rightarrow \tag{91}$$

We can see that as in the previous case, in this case the sample complexity needed to learn the right structure is decreases (increases) as γ / δ increases (decreases). Moreover, the higher (lower) the penalty is, the lower (higher) the sample complexity needed to learn the correct structure (dependence on η), which is very reasonable since in this subgroup we are looking only on graphs with more parameters than the correct one.

All the bounds also increase (decrease) when the number of variables in the correct BN, n, increase (decrease): In order to learn the correct structure of a BN with more (less) variables one needs more (less) samples.

Our work follows prior work, which is a combination of Haughton ([14], [13]) and Geiger *et. al.* ([16]). This work talks indirectly about the asymptotical consistency of the MDL score with the BIC penalty, when learning the correct BN structure, and derives bounds on the size of the error of learning a wrong structure instead of the correct one. In our work we expend this work in a few ways.

First, for BNs with $P_{B^*} \in B^* \setminus B$, we showed that the MDL score is asymptotically consistent for any penalty $\Psi(N) = o(N)$, whereas, they showed that the MDL score is asymptotically consistent only for the BIC penalty. Moreover, we presented the upper bound on the sample complexity of learning the right structure, as a function of both δ and the correct BN parameters. They only gave an bound on the error which depends on δ, and didn't specify the dependency on the BN. For BNs with $P_{B^*} \in B^* \cap B$ they showed that the MDL score is asymptotically consistent with a better penalty then us, but we also specified the dependency of the upper bound we found on the correct BN parameters.

There is more to be done in order to complete and expend our work:

1. Complete the proof of asymptotical consistency of BNs with $P_{B^*} \in B^* \cap B$ for any penalty $\Psi(N) = o(N)$, $\Psi(N) \to \infty$, and derive the upper bound on the sample complexity of learning the correct structure from data. One way of doing it is by translating the work on curved exponential families that Haughton ([14], [13]) did to the language of BNs. We have started to work in this direction.

2. Check whether the bounds we got are tight. One way of doing it is by doing simulations on the process of learning the correct BN structure from data.

3. Derive lower bounds on the sample complexity of learning the right BN structure from data.

4. Work with all kinds of graphs, not only with graphs with ordered variables.

5. Think about ease the requirements of the structure learned. Meaning, allow to learn BNs with graphs which are *similar* to the correct one. This similarity can be

defined in various ways, for example: graphs with less then 5% difference between their edges can be called similar.

To conclude, the upper bounds we got in this work indicate that it is very difficult to learn the correct structure of a BN from data, with high probability. In order to do so one needs an enormous amount of data. This amount is increased when the variables number of the BN are high and when we want to learn the correct structure with a very high probability (relation to δ). It is also increased when there are parameters in the BN which are very close to 0 or to 1 (relation to γ), and when there are 'weak' parents in the BN, i.e., parents with low influence on their children (relation to ε). Moreover, the upper bounds we gave can be even higher in real life since we worked with ordered variables, and we assumed that we are computationally unbounded.

In the next part of my thesis I developed a heuristic algorithm for reconstructing biological transcriptional networks from gene expression microarray data and sequence data. In light of the results in this part, and the fact that there are only a few dozen samples in gene expression microarray data, I chose not to use BNs in order to learn the structure of the networks.

Part II

An algorithm for Reconstructing Transcriptional Regulatory Networks

6 Introduction

6.1 Biological Motivation

Understanding transcriptional regulatory networks is a crucial step towards understanding fundamental cellular processes, such as growth control, cell-cycle progression and development, as well as differentiated cellular function such as hormone secretion and cell-cell communication [22]. On a fundamental level, transcription determines when and which genes are expressed. The determination of factors that control expression can offer further insight into the misregulated expression that is common in many human diseases ([23], [24]). In order to understand transcriptional regulatory networks one needs to 'reverse engineer' the regulatory mechanism, given information that we have, like gene expression data under various conditions, the upstream region of all genes and biological experiments that check whether a transcription factor binds to a gene's promoter.

6.2 Recent work

Much research has been done in order to understand transcriptional regulatory networks in the past few years. Researchers approached to this issue in various ways. One strategy infers global networks directly from whole genome microarray data.

Qian *et al.* [25] introduced an approach based on support vector machines (SVMs) to predict the targets of a transcription factor by identifying subtle relationships between their expression profiles. They worked with microarray expression data of *Saccharomyces Cerevisiae*, and constructed pairs of a transcription factor (TF) and one of its known targets. Each pair was identified by a vector containing the gene expression data of the TF and the gene expression data of the target. The vector length (L) was twice as long as the number of samples taken for each gene in the gene expression data. They

also constructed pairs of genes that are known not to be a TF and its target. They took the good and bad examples of TF-target relationship as points in an L-dimension hyperplane, and used SVM in order to separate the hyperplane, with good examples on one side of the separating manifold and bad example on the other side. Now they looked at pairs of *Saccharomyces Cerevisiae* genes and checked where their vector of expressions falls; if it falls on the good examples side, this pair is considered to be a TF-target pair.

Ihmels *et al.* [26] used microarray expression data of *Saccharomyces Cerevisiae* in order to build transcription modules which consist of a set of genes that are co-regulated in a specific cellular context, and a set of experimental conditions where this co-regulation is most stringent. They used the signature algorithm [27] in order to do so. The signature algorithm is an iterative algorithm that give scores to the genes and to the conditions. The genes' scores are combined of the condition scores and the gene expression over all conditions. The conditions' scores are combined of the genes scores and the condition expression over all genes. A transcription module contains the genes and conditions that their scores passed a certain threshold.

A different strategy combines between the whole genome microarray data approach and another approach which focuses on the identification of shared regulatory motifs in the promoters of co-regulated genes, signified by similar expression profiles.

Many approaches (e.g. [28], [29], [30], [31], [32]) use gene expression measurements to define clusters of genes that are potentially co-regulated, and then search for common motifs in the upstream regions of the genes in each cluster. Segal *et al.* [4] developed a procedure that identifies modules of co-regulated genes, their regulators and the conditions under which regulation occurs. The difference between their work and other works is the fact that their algorithm works in an iterative way. It begins by clustering the expression data, creating one module from each of the resulting clusters. Then it searches for a common motif in the upstream regions of genes assigned to the same module. It then iteratively refines the model, trying to optimize the extent to which the expression profile can be predicted transcriptionally. They use the Expectation-Maximization (EM) algorithm in order to learn the modules of the genes and the regulators of each module properly. For example, if a gene in a cluster has the motif of a regulator of a different cluster and doesn't have the motifs of the regulators of its own cluster, that gene is moved to the other cluster (a more mathematical explanation of this work can be seen in part I, section 1.2). This algorithm was also tested on *Saccharomyces Cerevisiae* microarray data.

Xing *et. al.* [33] developed a classifier for constructing transcriptional regulatory networks using gene expression and sequence data. They first determined which TFs are active in each experiment, using gene expression data; They assumed that each TF expression has a normal distribution, and associated a p-Value to the expression of each TF in an experiment, indicating the significance of the association between the TF and the gene expression changes. If the p-Value is small enough they assumed that the TF is active in the experiment. Next, for each experiment they calculated the probability that each active TF regulates each of the genes that contain the binding site of this TF in their promoter (the potential targets of the TF). They assumed that the potential targets gene expression of each TF in each experiment are from a mixture of three normal distributions: target genes that are repressed, not significantly regulated and induced, under an experimental condition. They learned the mixing proportion, mean and variance of each such distribution, and using these values they calculated the probability that a gene is regulated (activated/repressed) by a TF in each experiment. Last, for each TF and gene they calculated an average value, over all experiments, that represents the probability that the TF regulates the gene, and if this value passed a certain threshold they assumed that the TF regulates the gene. This classifier was tested on *Saccharomyces Cerevisiae* microarray data.

6.3 Interesting Observations

I measured the relation between two genes in two ways. The first one is measuring the Correlation Coefficients (CC) between expression of the genes, which searches for the linear relation between them (see section 7.2.1). The second one is measuring the Mutual Information (MI) between the expression of the genes, which captures also the non-linear relations between them (see section 7.2.2).

When I looked deeply at various gene expression data sets I discovered that the distribution of the relation between gene expression values of TFs and their target genes is very similar to the distribution of the relation between expressions of two random genes. That is, it is very hard to learn about a transcriptional connection between a TF and a gene from the relation between their expression values. As opposed to this observation, the distribution of the relation between two genes that are regulated by the same TF (also referred to as *SIM genes* in this work) is different from the distribution of the relation between two random genes. This difference is even stronger when the two genes are regulated by the same pair of TFs (also referred to as *MIM genes* in this work). In both cases, when the relation between two genes is higher it is more likely

that the two genes share one or more TFs, and when the relation is low it is more likely that the two genes are not co-regulated. Examples of the distribution functions that demonstrate these observations can be seen in two different data sets for CC in figures 4 and 6 and for MI in figures 5 and 7.

These observations contradict previous approaches that use only correlation between a TF and a putative target gene in order to determine whether the TF indeed regulates the gene, for example, the approach by which Qian *et. al.* [25] learned direct transcriptional relations in *Saccharomyces Cerevisiae*, and Xing *et. al.* approach [33].

I used these findings to develop a method for learning direction relations between TFs and their target genes in human transcriptional network.

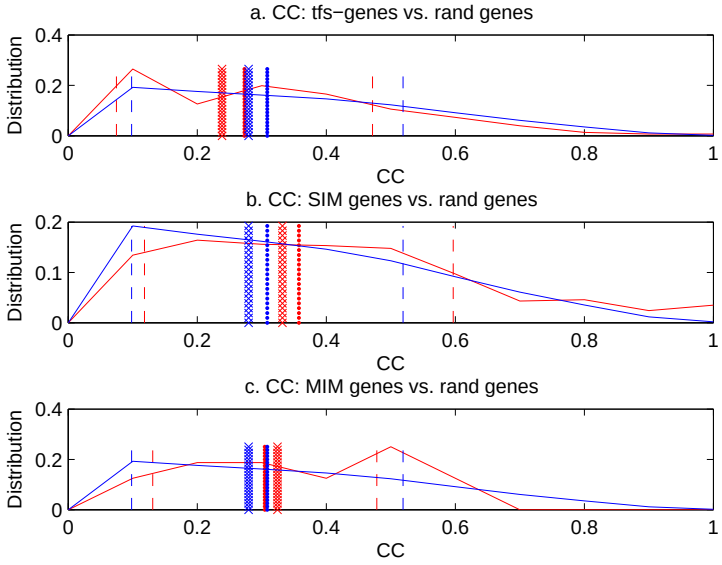

Figure 4: CC Distribution functions of the Milyavsky data (see section 7.5). CC is the absolute value of the correlation coefficients (see section 7.2.1). In all three graphs the blue line is the distribution function of the CC between 2 random genes in this data set (*Dist_CC_rand*, see section 7.2.3 step I). This distribution function is a discrete function that was calculated by dividing the [0,1] interval to 10, equal in size, bins. The vertical dotted blue line represents the mean of the distribution function. The vertical 'XXX' blue line represents the median of the distribution function. The two vertical broken blue lines represent one standard deviation (std) of the distribution function from the mean. All other distribution function were also calculated by dividing the [0,1] interval to 10 equal bins. a. The red line is the distribution function of the CC between a TF and its target (*Dist_CC_tf_target*, see section 7.2.3 step I). The vertical dotted red line, the vertical 'XXX' red line and the vertical broken red lines represent the mean, median and std of the TF-target pairs distribution function respectively. b. The red line is the distribution function of the CC between genes that are known to be regulated by the same TF (*Dist_CC_SIM_genes*, see section 7.2.3 step I). The vertical dotted red line, the vertical 'XXX' red line and the vertical broken red lines represent the mean, median and std of the TF-target pairs distribution function respectively. c. The red line is the distribution function of the CC between genes that are known to be regulated by two identical TFs (*Dist_CC_tf_target*, see section 7.2.3 step I). The vertical dotted red line, the vertical 'XXX' red line and the vertical broken red lines represent the mean, median and std of the TF-target pairs distribution function respectively.

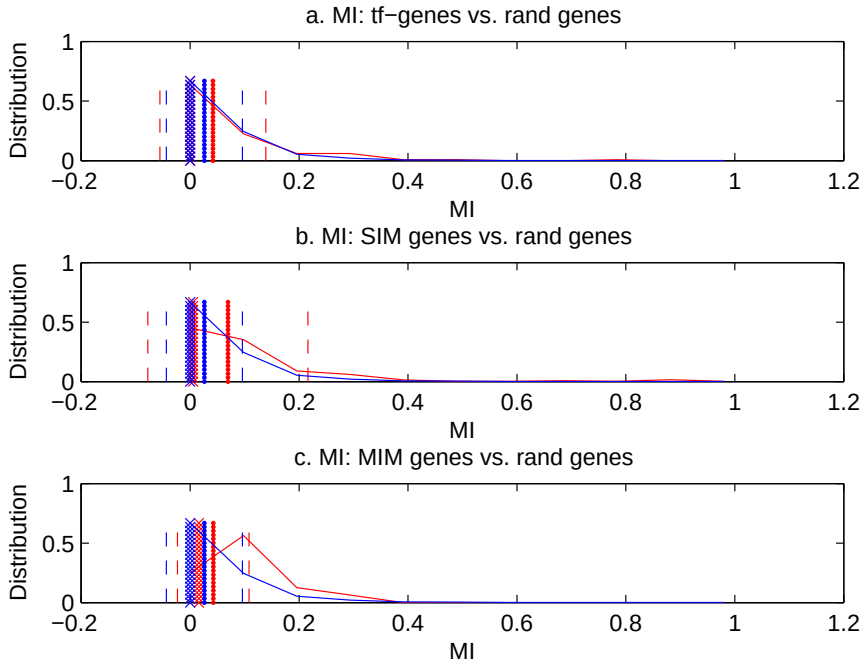

Figure 5: MI Distribution functions of the Milyavsky data (see section 7.5). MI is the mutual information (see section 7.2.2). In all three graphs the blue line is the distribution function of the MI between 2 random genes in this data set ($Dist\_MI\_rand$, see section 7.2.3 step I). This distribution function was calculated by dividing the MI interval to 10, equal in size, bins. The vertical dotted blue line represents the mean of the distribution function. The vertical 'XXX' blue line represents the median of the distribution function. The vertical broken blue lines represent the standard deviation (std) of the distribution function. All Other distribution functions were also calculated by dividing the MI interval to 10 equal bins. a. The red line is the distribution function of the MI between a TF and its target ($Dist\_MI\_tf\_target$, see section 7.2.3 step I). The vertical dotted red line, the vertical 'XXX' red line and the vertical broken red lines represent the mean, median and std of the TF-target pairs distribution function respectively. b. The red line is the distribution function of the MI between genes that are known to be regulated by the same TF ($Dist\_MI\_SIM\_genes$, see section 7.2.3 step I). The vertical dotted red line, the vertical 'XXX' red line and the vertical broken red lines represent the mean, median and std of the TF-target pairs distribution function respectively. c. The red line is the distribution function of the MI between genes that are known to be regulated by two identical TFs ($Dist\_MI\_tf\_target$, see section 7.2.3 step I). The vertical dotted red line, the vertical 'XXX' red line and the vertical broken red lines represent the mean, median and std of the TF-target pairs distribution function respectively.

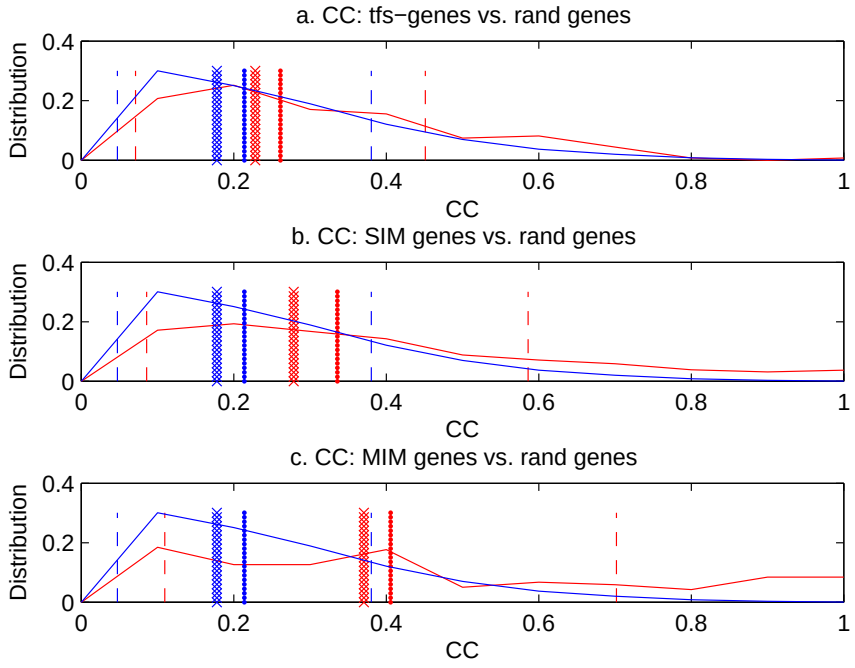

Figure 6: CC Distribution functions of the Colon data (for annotations see figure 4).

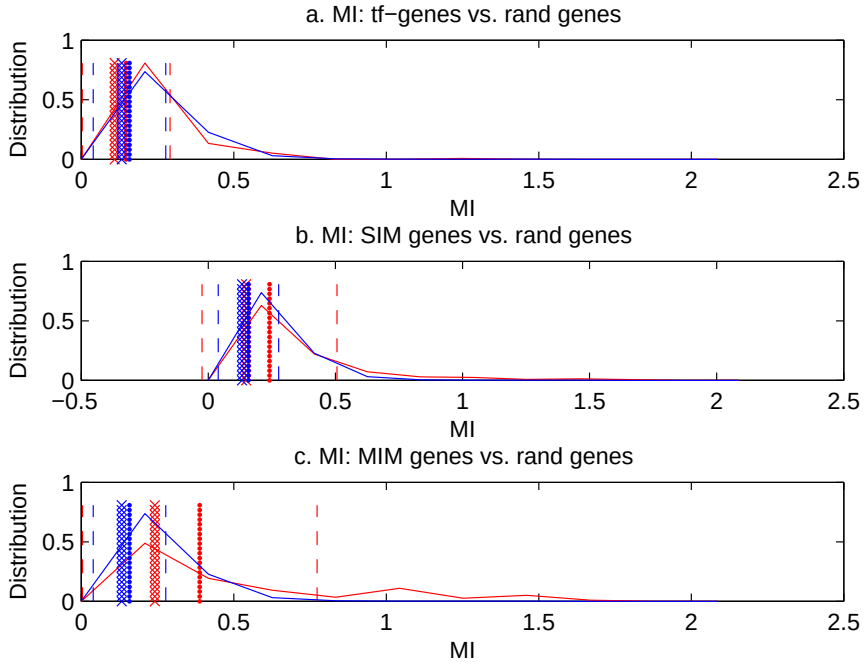

Figure 7: MI Distribution functions of the Colon data (for annotations see figure 5).

6.4 My approach

I present here a method for completing knowledge on a local regulatory network for which I have partial information. This method works with both gene expression microarray data and with information on the binding sites of transcription factors. The key concept of this method is that in order to learn whether a TF T regulates a gene G one needs to look at the relation between G and other genes that T regulates. Moreover, if we have a known target $\widetilde{G}$ of T which is also a known target of another TF that also regulates G, the relation between $\widetilde{G}$ and G can tell us more about whether G is a target of T. In order to determine the relation between two genes our algorithm measures both CC and MI between these two genes. My method receives a TF T, a gene G and some prior knowledge on the regulatory network related to T and / or G . Using gene expression microarray data it calculates a score indicating whether T regulates G. The score is combined of CC and MI measurements of relations between G and T, relations between G and genes that are known to be regulated by T and relations between G and genes that are regulated by both T and by another TF that also regulates G. If a score exceeds a certain threshold I conclude that T regulates G. At this point I also check the promoter of G in order to examine whether T's binding site sequence appears in G's promoter with high probability compared to a random promoter.

The main difference between this method and other methods is the fact that other methods look on the entire genome globally, while this method asks 'local' questions about a particular TF and gene at a time. Another difference is that most approaches use the relation between co-regulated genes (Segal *et al.* [4]) or between a TF and a gene (Qian *et. al.* [25]), but do not combine the two, as this method does. Moreover, this method uses both linear (CC) and non-linear (MI) relations between genes, while others use only linear relation between genes. The main disadvantage of our method is the fact that it needs prior information about genes that the TF under study is known to regulate. Another disadvantage of this method is that due to the fact that it looks on specific genes and not on clusters of genes it is very sensitive to noise.

I checked the method on human microarray datasets, which is also interesting since most works in this area checked their results on *Saccharomyces Cerevisiae* microarray data.

7 Methods and Materials

7.1 Database of human TF-target pairs (HTFT)

I constructed a database of human TF-target pairs. The database contains 54 TFs, 80 targets and 141 pairs. A TF in one pair can be a target in another pair. Most of the database pairs are taken from TRANSFATH, Some of them are also taken from the literature.

7.2 The Algorithm

Before describing the algorithm I want to explain how I calculate CC and MI.

Let $X = \{X_1, \ldots, X_n\}$ and $Y = \{Y_1, \ldots, Y_n\}$.

7.2.1 Calculating Correlation Coefficients

The known equation of Correlation Coefficients is:

$$corrcoef(X,Y) = \frac{Cov(X,Y)}{\sqrt{Cov(X,X)Cov(Y,Y)}} \tag{92}$$

where $Cov(X,Y)$ is the covariance between X and Y. $corrcoef(X,Y)$ returns a number between -1 and 1. In my calculations I take the absolute value of this number, because at the first stage of my work I only ask if a transcription factor regulates a gene and not whether the regulation is activation or repression, and thus I am not interested in the sign. So, in this work I set

$$CC(X,Y) = \mid corrcoef(X,Y) \mid$$

7.2.2 Calculating Mutual Information

The known equation of Mutual Information between two factors X, Y is:

$$I(X,Y) = \sum_{x \in X} \sum_{y \in Y} P(X,Y) \log \frac{P(X,Y)}{P(X)P(Y)}$$

The way I calculate MI in my context is the following. Say X and Y stand for the expression levels of two genes, measured for n samples. I get n pairs of the kind (X_i, Y_i), $i = 1, \ldots, n$. I divide the plane into rectangular bins of points. The way I do it is by dividing the square with the corners $(0,0)$, $(\max_{i=1\ldots n} X_i, 0)$, $(0, \max_{i=1\ldots n} Y_i)$, $(\max_{i=1\ldots n} X_i, \max_{i=1\ldots n} Y_i)$ into $\frac{n}{6}$ rows and $\frac{n}{6}$ columns (if $\frac{n}{6}$ is not a natural number I

take the rounded value of it). I get $\frac{n^2}{36}$ bins. Now, I take bins with a number of points that is much higher then average and divide them further, because I want to take a closer look into regions with a large concentration of points. So, if a row (column) contains more than four times the average number of points in each row (column) I divide this row (column) into a few, equal in size, parts (the number of parts is proportional to the number of points in this row (column)). An example can be seen in figure 8. Now, for these bins the MI can be calculated:

$$MI(X,Y) = \sum_{i=1}^{\sharp columns} \sum_{j=1}^{\sharp rows} r(i,j) \log \frac{r(i,j)}{p(i)q(j)}$$

Where $r(i,j)$ is the fraction of points in square (i,j) (out of the n points in the plane), $p(i)$ is the fraction of points in column i (out of the n points) and $q(j)$ is the fraction of points in row j (out of the n points).

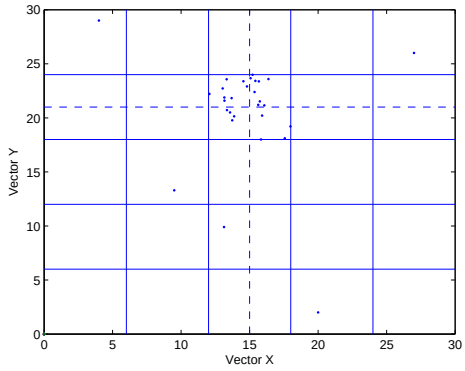

Figure 8: The squared bins of points from which I calculate the MI. Each vector (Vector X in axes X and vector Y in axes Y) contains 30 points. The column between 12 and 18 is divided into two sub columns, because the number of points in this column $> 24 = 6 * 4$, where fix is the average number of points in each column. From the same reason the row between 18 and 24 is divided into two sub rows (the broken lines).

7.2.3 Algorithm Description

The algorithm receives as an **input** the following arguments:

1. A Transcription factor T.
2. A list $\{g_1, \ldots, g_n\}$ of the n genes that are known to be regulated by T.
3. A gene G.
4. Gene expression microarray data.
5. The database of human TF-target pairs (HTFT).

Remark: we assume that HTFT contains the pairs (T, g_i), $i = 1, \ldots, n$. If these pairs are not in HTFT they should be added to the table before running the algorithm for this T.

The algorithm checks whether T regulates G. The algorithm gives a score to T, indicating the certainty that T regulates G. If this score passes a certain threshold Θ, the algorithm assumes that G is a target of T, and it's output is 1. Otherwise, the algorithm assumes that G is not a target of T and it's output is 0.

Step I: Constructing the distribution functions

As a first step the algorithm constructs the distribution functions of the following distributions (these functions describe the distributions of the gene expression microarray data that is one of the algorithm inputs):

1. Distribution of CC between random genes ($Dist\_CC\_rand$): the algorithm chooses randomly 10000 pairs of genes of the gene expression microarray, and calculates for each pair the CC between the two expression vectors. The distribution function is a discrete function that is calculated in the following way: I divide the CC interval (the $[0,1]$ interval) into 100 non-intersecting bins, count the number of CCs (from the 10000 random genes' CCs) in each bin and divide this number by 10000 (for normalization). More formally, Let $x \in [0,1]$ be a CC value of two genes

$$Dist\_CC\_rand(x) = \frac{\frac{i}{101} < \sharp CC \leq \frac{i+1}{101}}{10000}$$

for $\frac{i}{101} < x \leq \frac{i+1}{101}$ and $i = 0, \ldots, 100$.

2. Distribution of MI between random genes ($Dist\_MI\_rand$): This distribution function is calculated in the same way as the previous one, with one exception. Here the interval is between 0 and max_MI (instead of 1), where max_MI is the maximum MI of all 10000 random MIs.

3. Distribution of CC between transcription factors and their targets ($Dist\_CC\_tf\_target$): the algorithm goes over the pairs in HTFT and calculates the CC between each pair. The distribution function is also discrete. It is calculated in the following way: I divide the CC interval (the $[0,1]$ interval) into 40 bins. Now, each bin starts in the middle of its previous bin. I then count the number of CCs in each bin and divide this number by twice the number of pairs in HTFT (for normalization). More formally, Let $x \in [0,1]$ be a CC value of two genes

$$Dist\_CC\_tf\_target(x) = \frac{\frac{i-2}{80} < \sharp CC \leq \frac{i+2}{80}}{2 * \sharp TFs - targets}$$

for $\frac{i-1}{80} < x \leq \frac{i+1}{80}$ and $i = 0, 2, 4, 6, \ldots, 80$.

In this distribution function calculation the segments intersect. In this way we use each CC measurement to calculate two values of the function, and work as if we have twice as much points as we calculated. The difference between this distribution

calculation and ($Dist\_CC\_rand$) lays on the fact that there are much less points from which we build this distribution.

4. Distribution of MI between transcription factors and their targets ($Dist\_MI\_tf\_target$): This distribution function is calculated in the same way as the previous one, with one exception. Here the interval is between 0 and max_MI (instead of 1), where max_MI is the maximum MI of all TF-target pairs.

5. Distribution of CC between genes that are regulated by the same TF ($Dist\_CC\_SIM\_genes$): This distribution function is the same as $Dist\_CC\_tf\_target$ (3), only measured for all the pairs of genes that are regulated by the same TF from HTFT.

6. Distribution of MI between genes that are regulated by the same TF ($Dist\_MI\_SIM\_genes$): This distribution function is calculated in the same way as the previous one, with one exception. Here the interval is between 0 and max_MI (instead of 1), where max_MI is the maximum MI of all the pairs of genes that are regulated by the same TF from HTFT.

7. Distribution of CC between genes that are regulated by two identical TFs ($Dist\_CC\_MIM\_genes$): This distribution function is the same as $Dist\_CC\_tf\_target$ (3), only measured for all the pairs of genes that are regulated by two identical TFs from HTFT. Meaning, if the pairs TF1-target1, TF1-target2, TF2-target1 and TF2-target2 are all in the table, then target1-target2 is one of the pairs from which this distribution function is measured.

8. Distribution of MI between genes that are regulated by two identical TFs ($Dist\_MI\_MIM\_gene$ This distribution function is calculated in the same way as the previous one, with one exception. Here the interval is between 0 and max_MI (instead of 1), where max_MI is the maximum MI of all the pairs of genes that are regulated by the same 2 TFs from HTFT.

Step II: Calculating the score

The algorithm calculates the score vector $\overrightarrow{S} = S_1, \ldots, S_6$ for T and G, using the gene expression data set. The score vector is combined of the following components:

S_1 - The log likelihood of the CC of T and G:

$$\log \frac{Dist\_CC\_tf\_target(CC(T,G))}{Dist\_CC\_rand(CC(T,G))}$$

S_2 - The log likelihood of the MI of T and G:

$$\log \frac{Dist\_MI\_tf\_target(MI(T,G))}{Dist\_MI\_rand(MI(T,G))}$$

S_3 - The log likelihood of the CC between G and the genes that are regulated by T:

$$\log \prod_{i=1}^{n} \frac{Dist\_CC\_SIM\_genes(CC(G, g_i))}{Dist\_CC\_rand(CC(G, g_i))}$$

S_4 - The log likelihood of the MI between G and the genes that are regulated by T:

$$\log \prod_{i=1}^{n} \frac{Dist\_MI\_SIM\_genes(MI(G, g_i))}{Dist\_MI\_rand(MI(G, g_i))}$$

S_5 - The log likelihood of the CC between G and the genes that are regulated by both T and by another TF from HTFT that regulates G:

$$\log \prod_{i=1 \wedge \exists TF \neq T, s.t.(TF, g_i) \in HTFT}^{n} \frac{Dist\_CC\_MIM\_genes(CC(G, g_i))}{Dist\_CC\_rand(CC(G, g_i))}$$

S_6 - The log likelihood of the MI between G and the genes that are regulated by T and by another TF from HTFT that regulates G:

$$\log \prod_{i=1 \wedge \exists TF \neq T, s.t.(TF, g_i) \in HTFT}^{n} \frac{Dist\_MI\_MIM\_genes(MI(G, g_i))}{Dist\_MI\_rand(MI(G, g_i))}$$

The score of T for this G is:

$$Score(T, G) = \overrightarrow{S} * \overrightarrow{W}$$

Where $\overrightarrow{W} = W_1, \ldots, W_6$ is a weight vector determined by a learning process, indicating the influence of each score component $S_i; i = 1, \ldots, 6$ on the score.

Step III: Algorithm Output

The last step of the algorithm is to check if $(Score(T, G) - \Theta) > 0$. If so, the algorithm assumes that T regulates G and returns 1, otherwise, it returns 0.

The algorithm's parameters $\overrightarrow{W}$ and Θ are determined by a learning process.

7.2.4 Learning the parameters

In order to learn the algorithm parameters I construct a set of examples. Each example is a pair of a TF and a gene. The TF is one of the TFs in HTFT and the gene is one of the targets in HTFT. For each pair I know from the database whether the TF regulates the gene (if the pair exists in HTFT I know that the TF regulates the gene, otherwise I assume that the TF doesn't regulate the gene). Moreover, I can construct, using the algorithm, a vector $\overrightarrow{S} = S_1, \ldots, S_6$ that contains the score components for each pair.

I divide the examples into two groups, a training set and a test set, in the following way: I choose randomly 80% of the TFs in HTFT. All the pairs with one of these chosen

TFs belong to the training set. The rest of the pairs belong to the test set. Suppose the training set size is N and that the training set examples are $\{e_1, \ldots, e_N\}$.

I use the *Delta rule* algorithm on the training set examples in order to learn the parameters. The Delta rule algorithm is a version of the *Perceptron algorithm* for examples which are not linearly separable. I chose a method which doesn't impose linear separation of the examples, because this problem is not necessarily linearly separable; moreover, even if the problem were, I believe that HTFT is not a complete and perfect database, and thus one should allow for false positives examples.

The key idea of the Delta rule algorithm is to use a *gradient descent search* (an algorithm for finding the nearest local minimum of a function, using the gradient). The Delta rule algorithm defines an error function

$$E = \frac{1}{2}\sum_{i=1}^{N}(t_i - O_i)^2$$

where t_i is 1 if the couple in e_i exists in HTFT, and -1 otherwise, and $O_i = tanh(\overrightarrow{S_i} * \overrightarrow{W})$, where $\overrightarrow{S_i} = \{S_{i1}, \ldots, S_{i6}, -1\}$ ($S_{i1}, \ldots, S_{i6}$ are the score vector components of e_i TF and e_i gene from the algorithm) and $\overrightarrow{W} = \{W_1, \ldots, W_6, W_7 = \Theta\}$. The idea is to find a minimum for the error function E in the space of weights (see figure 9).

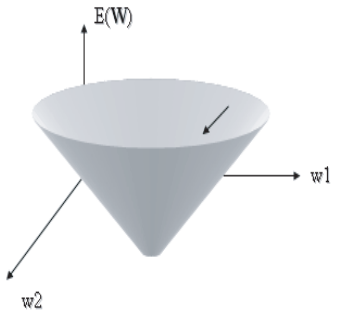

Figure 9: Delta rule minimization of the energy.

In order to do so we calculate the gradient of E with respect to the weights vector $\overrightarrow{W}$, denoted as $\nabla E(W) = [\frac{\partial E}{\partial W_1}, \ldots, \frac{\partial E}{\partial W_6}, \frac{\partial E}{\partial \Theta}]$.

The delta rule is:

$$W_i = W_i - \eta\frac{\partial E}{\partial W_i} \tag{93}$$

for $i = 1, \ldots, 7$ and η being a learning rate. I chose it to be 0.001.

The method for learning the parameters $\overrightarrow{W} = \{W_1, \ldots, W_7\}$ is:

Repeat 10 times or until E doesn't improve (whatever comes first):

1. Choose randomly values for $\overrightarrow{W}$.

2. Calculate new values for $\overrightarrow{W}$, using (93). In order to perform this step one needs to run the algorithm on each example from the training set in order to receive new values for $S_1, \ldots, S_6$.

3. Calculate E with new $\overrightarrow{W}$ values.

I repeated this procedure 20 times, and chose the $\overrightarrow{W}$ that minimizes the error, but now instead of calculating E, I calculate the exact error - using sign function instead of tanh. The reason that I repeat this procedure is because this method can get stuck in a local minimum.

7.3 Tools for searching binding site motifs in genes promoters

7.3.1 STOP - Searching TFs Of Promoters

STOP is a tool that was developed in our lab and is based on the work of Hertzberg *et. al.* [34]. It receives a gene as an input and specifies the TFs whose binding sites appear in that gene's promoter with high probability compared to a random sequence. In order to do so STOP uses the promoter of the gene and the Position Weight Matrix (PWM) of each TF. A PWM is a common representation of transcription factor binding sites. It is built out of all the known motifs to which the transcription factor binds, and it counts the number of appearances of every nucleotide in every position of the motif. For each TF, STOP calculates a score specifying the confidence that the TF binds to the gene promoter. The p-Value of a TF and a gene is then determined to be the probability to get a better score than the one the TF would get on a random promoter. Now, in order to decide if the TF binds to the promoter of the given gene, STOP sets a threshold on the p-Value. Each TF has a different threshold since each PWM was built from a different number of promoters and each PWM has a different length. If the p-Value of the gene and the TF is lower than the TF threshold, STOP will say that this TF binds to the gene promoter. STOP works with the 392 TFs from TRANSFAC. It looks for each promoter at 1000 bp upstream of the transcription start site.

7.3.2 POC - Promoters of Clusters

POC is another tool that was developed in our lab by Tabach [35]. It receives a cluster of genes as an input, and finds statistically significant sequences of transcription factor binding sites, that are enriched in the promoters of the cluster's genes, compared to the entire genome. POC examines 326 TFs binding sites within the promoters of the

cluster's genes. In order to do so it uses the human regulatory position specific score matrices (PSSMs [36],[29]). To identify regulatory motifs in the promoter of a gene, the PSSM assigns a weight to the presence of each nucleotide (A, C, T, G) at each position in the motif; this weight represents the extent to which the nucleotide's presence at this position is associated with the motif. It is based on the PWM (see 7.3.1). The 326 TFs that are examined by POC are the ones that their PSSMs appear in the Mathinspector [37] database. POC looks for each promoter at 1000 bp upstream of the transcription start site.

7.4 Tools for data analysis

7.4.1 SPIN

Another exploratory analysis method that uses groups of correlated genes for meaningful ordering of patients is SPIN (Sorting Points Into Neighborhoods) (Tsafrir *et. al.*, [38]) our recently proposed methodology for data organization and visualization. At the heart of this method is a presentation of the full pairwise distance matrix of the samples, viewed in pseudo-color. The samples are iteratively permuted in search of an optimal ordering, i.e. one that can be used to study embedded shapes. Hence, certain structures in the data (elongated, circular and compact) manifest themselves visually in a SPIN generated distance matrix.

7.5 Data sets

I worked with two gene expression microarray data sets in order to examine this method.

1. **The Milyavsky data.** Milyavsky *et al.* [39] have modeled in-vitro cellular transformation through a stepwise process that began with isogenic cells. A 600-days long transformation process (see figure 10) started with normal WI-38 human diploid fibroblasts that entered replicative senescence after 40 population doublings (PDLs). In order to overcome replicative senescence, the cells were infected with over-expressed human telomerase (hTERT), resulting in immortalization. In addition H-Ras was over-expressed in the cells in several stages during the process. Samples of mRNA were taken at 12 time points (stages), with two repeats from each stage. Expression profiles were determined using the Affymetrix Human Genome Focus GeneChip Array. I worked with the 24 samples and performed some pre-processing on the data. First I removed from the data the genes whose expression level was less than 100 for all samples. Next, I

changed all remaining expression values that were less than 1 to 1 and took $\log_2$ of the data.

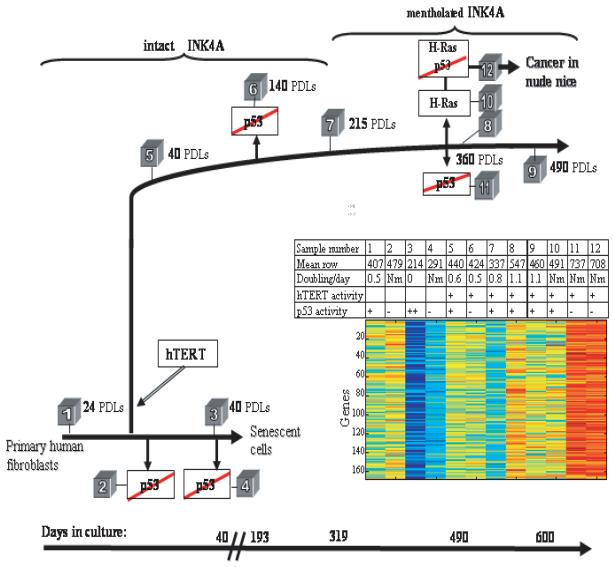

Figure 10: The 12 stages of the Milyavsky data.

2. **The Colon data.** I used 38 samples of normal tissues: 22 normal colon tissues, 11 normal liver tissues and 5 normal lung tissues. Expression profiles were determined using the Affymetrix Human Genome U133 GeneChip Array (see [38] for more information). I performed on this data the same pre-processing steps I did on the Milyavsky data.

8 Results

8.1 Learning the parameters of the data sets

First of all I checked the parameters that the algorithm learned (see section 7.2.4) for each one of the data sets I worked with. Then, I compared the results of $\overrightarrow{W}$ to the distribution functions of each data and saw if the results are logical. I also calculated the error rate and the p-Value of the algorithm for each data set in order to evaluate its performance and compare it to a random classifier.

8.1.1 The Milyavsky data

The parameters that were learned for the algorithm are:

$$\overrightarrow{W} = (0.0225, 0.029, 0.218, 0.4063, 0.3756, 0.5884)$$

and

$$\Theta = 1.2$$

It can be seen that S_1 and S_2, which are the score components that measure the log likelihood of the relation (CC and MI respectively) between T and G, got significantly lower weights, than the other score components. We can also see that the weight of S_3 (S_4), which is the log likelihood of the CC (MI) between G and the genes that T regulates, is lower then the weight of S_5 (S_6), which is the CC (MI) between G and the genes that are regulated by T and by another TF that also regulates G.

All these observations strengthen my basic assumptions: the relation between a TF and its putative target gene can't be studied by the CC/MI between the TF and the gene. However, it can be studied by the CC/MI between this putative target gene and another gene that is a known target of the TF (if the CC/MI between these two genes is high, there is a better probability that the putative target gene is indeed regulated by the TF), and if these two genes have another TF that regulates both of them, the CC/MI between them teaches even more about the connection between the TF and the putative target gene.

It can be seen that the weights of the score components that measure MI (W_2, W_4, W_6) are higher than their corresponding CC score components: $W_2 > W_1$; $W_4 > W_3$; $W_6 > W_5$. This last finding indicates that MI measurements teach us more than CC measurements about the relationship between genes, at least for this data.

It is interesting to compare these results to the distribution functions of this data (see 7.2.3, step II). Figure 4 and figure 5 contain the Milyavsky data CC and MI distribution functions (respectively) of TF-target pairs (a), SIM genes pairs (two genes that are regulated by the same TF) (b) and MIM genes pairs (two genes that are regulated by the same pair of TFs) (c), as opposed to the distribution function of random genes.

It is easy to understand from figure 4 the reasons that caused W_1 to be lower than W_3 and W_5; the mean and median of SIM and MIM genes pairs are much higher than the mean and median of TF-gene pairs (the last mean and median are even lower than the mean and median of random genes). Moreover, the scattering of SIM and MIM genes pairs are shifted to the right, compared to TF-gene pairs. It is harder to understand why $W_5 > W_3$, since the mean of SIM genes pairs is much higher than the mean of MIM genes pairs, and their medians are almost the same. One way of explaining this result is by the fact that in the examples that are used to learn the parameters, there are genes that appear only once, while others appear a few times (if we know of several TFs that regulate them), so not all the MIM genes pairs have the same influence on the scores

of all examples. It could be that taking into account the influence of each pair to all the scores, will get us a new mean and median to the MIM genes pairs CCs, which are higher than the SIM genes pairs CCs.

Now, let us look at figure 5. $W_2 > W_1$ can be justified by the fact that the mean and median of CC of TF-gene pairs are lower than the mean and median of random genes' CC, whereas in MI this is not the case. The fact that $W_4 > W_2$ can be explained using the observations that the mean and median of SIM gene pairs' MIs are higher than the mean and median of TF-gene pairs' MI. The explanation of $W_6 > W_4$ could be the same as the explanation of $W_5 > W_3$.

I checked the percent of false positives (pairs that are not in HTFT but the algorithm output to them was 1) and false negatives (pairs that are in HTFT but the algorithm output to them is 0) over the test set examples. I ran the algorithm with the chosen $\overrightarrow{W}$ on each pair of the test set. For each pair the algorithm determined whether the gene is the TF's target. I counted the number of FPs, and the number of FNs and found that the algorithm had 10% false positives rate and the false negatives fraction was 60%.

The false positive (FP) percent and false negative (FN) percent are correlated with the prior knowledge we have. The larger the prior information we have, the accurate the algorithm results. Thus, although on the test set the FP percent and FN percent are not so good (10% and 60% respectively), these values will be much lower when one researches a specific transcriptional network, on which he has a lot of prior information. Moreover, in order to reduce the FP fraction of the algorithm, I also check if the binding site of the TF appears on the promoter of the gene with high probability, compared to a random gene, for each pair that is chosen by the algorithm to be a TF-gene pair. Only if this is the case I assume that the TF indeed regulates the gene.

I compared this classifier to a random classifier, also using the test set examples. In order to determine whether this classifier is better than a random one I checked the probability to get better results (lower FPs percent and / or lower FNs percent) using the following formula:

$$\frac{\sum_{i=1}^{FN}\sum_{j=1}^{FP}\binom{\sharp 1s}{i}\binom{\sharp 0s}{j}}{\sum_{i=1}^{\sharp 1s}\sum_{j=1}^{\sharp 0s}\binom{\sharp 1s}{i}\binom{\sharp 0s}{j}} \tag{94}$$

where $\sharp 1s$ is the number of pairs in the test set that are in HTFT, and $\sharp 0s$ is the number of pairs in the test set that are not in HTFT.

I got that this probability is equal to $1.0695 * 10^{-125}$, which is almost 0. Meaning, this classifier is much better than a random one.

8.1.2 The Colon data

The parameters that were learned by the algorithm are:

$$\overrightarrow{W} = (0.002, 0.0001, 0.01, 0.025, 0.2243, 0.3212)$$

and

$$\Theta = 1.5$$

It can be seen that here also $W_6 > W_4 > W_2$ and $W_5 > W_3 > W_1$, and indeed looking at the various distribution functions of the CC and MI of this data set (figure 6 and figure 7) one can see that both for MI and CC the mean of MIM genes is higher than the mean of SIM genes, and that the last mean is higher than the mean of TF-target genes. Moreover, all of these means are higher than the random genes' mean. These results also strengthen our basic assumptions.

There are two obvious differences in this data set parameters, compared to the previous data set parameters. The first one is that $W_1 > W_2$. This difference can be explained by looking at the distribution functions of this data set, figure 6 (a) and figure 7 (a), and the distribution functions of the previous data set, figure 4 (a) and figure 5 (a). In this data set we can see that the mean and median of CC of TF-target genes is higher than the mean and median of CC of random genes, however, when looking at MI of TF-target genes, it can be seen that the mean and median of random genes is higher than the mean and median of TF-target genes. In the previous data set we can see the opposite. The second difference between this data set parameters and the previous data set parameters is that the difference between $W_3(W_4)$ and $W_5(W_6)$ is much higher in this data set (as opposed to the previous one). We can see in figure 6 (b,c) and figure 7 (b,c) that the mean and median of MIM genes' CC and MI is much higher than the mean and median of SIM genes' CC and MI (respectively). This sharp difference is not seen in the previous data set distribution functions.

I checked the percent of false positives and false negatives over the test set examples (the same way as in the previous data set) and got that the false positives percent is 20% and the false negatives percent is 60%.

When comparing this classifier to a random one, using eq. 94, I got that the probability that a random classifier will be better than this one is $5.89 * 10^{-69}$, which means that on this data also this classifier is much better than random.

8.2 Learning A Transcriptional Network related to Smooth Muscle Cells Differentiation

As a next step, I tried to learn part of the transcriptional network that controls vascular smooth muscle cells (SMC) differentiation. SMCs provide hemostatic control and protect new endothelium-lined vessels against rupture or regression. SMCs also assist endothelial cells in acquiring specialized functions in different vascular beds and maintain vascular tone and integrity in the adult [40]. Smooth muscle cells are important for the functions of the circulatory, genitourinary, respiratory, and digestive systems [41]. Abnormal growth and proliferation of vascular SMCs is a key feature of vascular diseases such as atherosclerosis, restenosis, and hypertension [42]. Despite the importance of changes in the differentiated state of SMCs in vascular diseases, the molecular mechanisms controlling SMC differentiation are still largely unknown. An understanding of the normal regulation of SMC development and differentiation will not only provide the foundation for elucidating how these processes may be disrupted in vascular diseases but will also be critical to understanding congenital defects in vascular development and vascular development within solid tumors [42].

I chose to concentrate on the transcription factor SRF (Serum Response Factor) and on two of its target genes: SM22α and SM α-actin.

SRF is a known regulator of SMC differentiation. SRF binding to CArG (CC[A/T]GG) box has been shown to regulate numerous muscle-specific genes. In particular, SRF binds to the promoters of SM α-actin and SM22α through the CArG box [42].

Both SM α-actin and SM22α are muscle differentiation marker genes [43]. SM α-actin is a contractile protein that comprises 40% of total SMC protein [44]. It is required for the contractile function of SMCs and is the first SMC differentiation marker to appear during development [45]. SM22α is a calponin-related protein that is specific to adult SMCs [42].

I wanted to discover more targets of SRF and more transcription factors of SM α-actin and of SM22α. The first step I took is building a partial transcriptional network around SRF, SM α-actin and SM22α (see figure 11). This network's connections are from [42], [46], [47], [48], [49] and [50]. There are more known SM α-actin TFs (PRRX1, NKX3-1, KLF4, PURA, PURB, TEAD1 ARID5B and MYB [42]) and SM22α TFs (BAPX1, SP1, SP3, KLF4, KLF5, TWIST1, SSRP1 and ARID5B [42]), but they are not in HTFT, and thus I don't have any prior information about their known target genes, so I didn't add them to the partial network, and I didn't check whether my

method identifies them as SM α-actin or SM22α TFs. There are more known SRF targets (FOS, DES) but they do not appear in the FOCUS chip (the Milyavsky data set) and thus I didn't add them to the partial network.

I started with this partial network in order to check whether my method identifies the known TFs of SM α-actin and of SM22α, and the known targets of SRF, and to discover unknown transcriptional relationships.

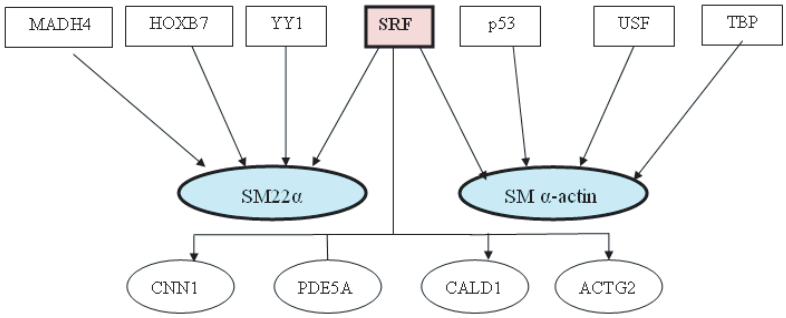

Figure 11: The partial network of SRF, SM α-actin and SM22α. In the squares are the TFs, and in the circles are the target genes. An arrow between each TF and its target represents an activation relation. A clean line represents an unknown relation.

8.2.1 SM α-actin TFs

(I) I ran the algorithm on the Milyavsky data set with SM α-actin as the target gene G, and with each of the 54 TFs from HTFT as a TF. When running the algorithm with any known TF of SM α-actin (SRF / USF / TBP / p53) I didn't use the transcriptional relation between that TF and SM α-actin as prior knowledge of the algorithm. Otherwise, the score of each of these TFs would have been biased because the relation between SM α-actin and itself would be measured in S_3 and S_4.

The TFs that were chosen by the algorithm to be SM α-actin TFs can be seen in table 1 (the ones with the Milyasvky data score > 1.2). It can be seen that the algorithm identified SRF and USF as SM α-actin TFs, but didn't identify TBP and p53 as its TFs (the scores of these TFs can be seen in table 1).

In order to figure out which of the remaining TFs are more likely to be SM α-actin TFs (and which of the remaining TFs are noise) I checked the promoter of SM α-actin, using STOP (7.3.1), in order to examine which of these TFs binding sites appear in the SM α-actin promoter with high probability comparing to a random promoter.

ETV7 binding site was found by STOP with p-Value < 0.05. I checked in the literature to see whether a connection between ETV7 and SM α-actin is known. Such a connection is not found in the literature.

The binding site of PBX1 was found by STOP with p-Value < 0.1. This binding site is known to contain the binding site of HOX-PBX binding site, which means that if the binding site of PBX1 is found in the promoter of SM α-actin, then the binding site of HOX-PBX is also on this promoter. HOXB7 is known to bind to the HOX-PBX binding site [51], so it is possible that HOXB7 binds to the promoter of SM α-actin. There is no known direct transcriptional connection in the literature between HOXB7 and SM α-actin, but it is known (see partial network in figure 11) that HOXB7 regulates SM22α.

The rest of the TFs got from STOP p-Value > 0.5 and are assumed to be noise.

(II) I repeated the same procedure using the Colon data set. The TFs that were chosen by the algorithm to be SM α-actin TFs can be seen in table 1 (the ones with the Colon data score > 1.5). Here we can also see that the algorithm identified both SRF and USF as SM α-actin TFs, but it didn't identify TBP and p53 as SM α-actin TFs (the scores of these TFs can be seen in table 1).

Checking the promoter of SM α-actin, using STOP, returned the following results:

TTF1 binding site was found on the promoter of SM α-actin with p-Value < 0.05. There is no known connection between SM α-actin and TTF1 in the literature.

VDR binding site was found on the promoter of SM α-actin with p-Value < 0.5. A direct transcriptional connection between VDR and SM α-actin is not known, however, there are some references of an influence of VDR on vascular SMCs. It is assumed that the steroid hormone 1 ,25-dihydroxyvitamin D3 [1 , 25-(OH)2D3] promotes vascular SMC growth and calcification via VDR [52], [53].

HOXB7 binding site was found in the promoter of SM α-actin with p-Value < 0.1 (see (I)).

The rest of the TFs got from STOP p-Value > 0.5 and are assumed to be noise.

TF (Gene Symbol)	The Milyavsky data score	The Colon data score	STOP p-Value	Literature
HOXB7	2.72	4.05	< 0.1	SMC related
SRF	3.8	3.84	< 0.05	Known
YY1	-2.44	3.4	> 0.5	
CREM	-0.79	2.91	> 0.5	
IRF7	0.21	2.78	> 0.5	
HMGA1	0.21	2.43	> 0.5	
HIF1A	3.4	2.17	> 0.5	
CTCF	-5.94	2.1	> 0.5	
TTF1	0.98	2.07	< 0.05	
GATA2	0.22	2.04	> 0.5	
NME2	-0.5	1.95	> 0.5	
ETS2	1.17	1.91	> 0.5	
RB1	0.77	1.88	> 0.5	
SP2	0.24	1.88	> 0.5	
PSMD12	1	1.81	> 0.5	
SMARCA3	-0.004	1.71	> 0.5	
HNRPK	0.38	1.59	> 0.5	
JUN	-0.9	1.58	> 0.5	
VDR	0.62	1.54	< 0.5	SMC related
ZNF148	1.28	1.509	> 0.5	
USF2	1.81	1.503	< 0.05	Known
TOPORS	1.27	1.45	> 0.5	
ACTR1A	2.59	1.36	> 0.5	
STAT5B	1.76	1.33	> 0.5	
MEF2D	1.25	1.26	> 0.5	
ETV7	1.56	1.08	< 0.05	
ELF2	1.56	0.86	> 0.5	
p53	-2.78	0.5	> 0.5	Known
TBP	-0.15	0.45	< 0.05	Known
ATBF1	1.43	-8.9	> 0.5	

Table 1: SM α-actin TFs. The table contains the known SM α-actin TFs and new TFs that passed the algorithm threshold, when running it with the TF and with SM α-actin, using the Milyavsky and / or the Colon data. The first column contains the TF gene symbol. The second (third) column contains the score assigned to the TF by the algorithm, using the Milyavsky (Colon) data. The fourth column contains the STOP p-Value of the appearance of the TF binding site in SM α-actin promoter compared to a random gene. The fifth column contains literature information, it specifies whether each TF is known to be SM α-actin TF (Known) or known to have a role in SMC (SMC related). Yellow TFs are TFs that are known from literature to regulate SM α-actin. Blue indicates new TFs that were chosen by my method to be SM α-actin TFs. These are TFs that both passed the algorithm threshold (for at least one of the data sets) and whose binding site appears in the SM α-actin promoter with high probability compared to a random promoter (according to STOP).

I checked the location of the known and new putative SM α-actin TFs on the promoter of SM α-actin, in order to try to understand if any of these TFs work together, as a complex, when regulating SM α-actin, or if one TF affects the regulation of SM α-actin by another TF. As can be seen in figure 12 the pairs that might be connected in some way are TTF1 and HOXB7, and/or ETV7 and USF.

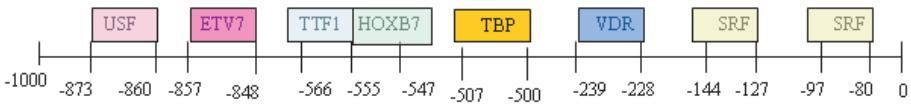

Figure 12: The promoter of SM α-actin. The location of each known and new putative TF of SM α-actin on the 1000 base-pairs of this promoter. p53 binding site is not found on the promoter.

Next, I asked which of the new TFs I found are SM α-actin activators and which are its repressors. In order to answer this question I compared the gene expression of SM α-actin to the gene expression of each of the new putative TFs found and to the gene expression of the target gene of each TF, taken from HTFT. Moreover, I checked the correlation coefficients (now without absolute value, see eq. 92) between SM α-actin and each TF, and between SM α-actin and the target gene of that TF (from HTFT). I used the two data sets I work with in order to do this. The results for the Milyavsky data can be seen in figure 13, the results for the Colon data are very similar.

When specifying the correlation coefficients between any two genes the first value denotes the correlation coefficients measured using the Milyavsky data and the second value denotes the correlation coefficients measured using the Colon data.

In figure 13 (a) we can see the gene expressions of SM α-actin (red), of ETV7 (green) and of LYN (blue), which is a known target of ETV7. We can see that there is high correlation between LYN and SM α-actin (correlation coefficient 0.71/0.57). We can also see that the correlation between SM α-actin and ETV7 is low (0.11/0.14) and thus we can't learn from it about the sign of the transcriptional connection between them. It is not known from the literature if ETV7 activates or represses LYN, but I can assume that ETV7 activates both LYN and SM α-actin or represses both of them.

In figure 13 (b) we can see the gene expressions of SM α-actin (red), of TTF1 (green) and of SELENBP1 (blue), which is a known target of TTF1. We can see that there is high correlation between SELENBP1 and SM α-actin (correlation coefficient 0.4/0.56). We can also see that the correlation between SM α-actin and TTF1 is very low (-0.08/-0.02) and thus we can't learn from it about the sign of the transcriptional connection between them. It is not known from the literature if TTF1 activates or represses SELENBP1, but I can assume that TTF1 activates both SELENBP1 and SM α-actin or represses both of them.

In figure 13 (c) we can see the gene expressions of SM α-actin (red), of HOXB7 (green) and of SM22α (blue), which is a known target of HOXB7. We can see that there is a very high correlation between SM22α and SM α-actin (correlation coefficients between them is 0.95/0.95). We can also see that the correlation between SM α-actin and HOXB7 is low (0.02/-0.22) and thus we can't learn from it about the sign of the transcriptional connection between them. It is known from literature that HOXB7 activates SM22α, so we can conclude that HOXB7 also activates SM α-actin.

In figure 13 (d) we can see the gene expressions of SM α-actin (red), of VDR (green) and of CDKN1A (blue), which is a known target of VDR. We can see that although the

correlation coefficients between SM α-actin and CDKN1A is not very high (0.26/0.33) the behavior of their gene expression is similar in most samples. Moreover, the correlation coefficients between SM α-actin and VDR is relatively high (0.45/0.46). Combining these facts with the fact that VDR is an activator of CDKN1A, leaded me to conclude that VDR is an activator of SM α-actin.

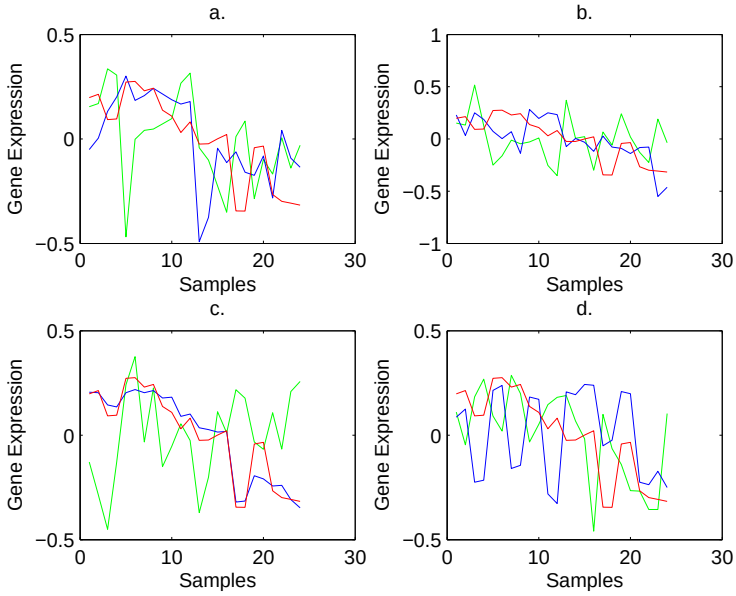

Figure 13: Gene expression of SM α-actin, it's new putative TFs and their known targets, using the Milyavsky data (these graphs using the Colon data are very similar). In each of the plots the x-axis contains the 24 samples and the y-axis contains the gene expression of the gene, after the following pre-processing: log2, center and normalization. the red line is SM α-actin gene expression, the green line is the TF gene expression and the blue line is the gene expression of the TF known target gene. (a) TF: ETV7, known target gene: LYN. (b) TF: TTF1, known target gene: SELENBP1. (c) TF: HOXB7, known target gene: SM22α. (d) TF: VDR, known target gene: CDKN1A.

8.2.2 SM22α TFs

(I) As before, I started by running the algorithm with SM22α as the gene G and with each of the HTFT TFs using the Milyavsky data set. The results can be seen in table 2 (the TFs with the Milyasvky data score > 1.2).. When running the algorithm with any known TF of SM22α (SRF / HOXB7 / YY1 / SMAD4) I didn't use the transcriptional relation between that TF and SM22α as prior knowledge of the algorithm. Otherwise, the score of each of these TFs would have been biased because the relation between SM22α and itself would be measured in S_3 and S_4.

It can be seen that the algorithm identified SRF and MADH4 as SM22α TFs, and didn't recognize YY1 and HOXB7.

I checked, using STOP, which binding sites of the TFs that passed the algorithm threshold appear in the promoter of SM22α with high probability compared to a random promoter.

The binding site of ATF was found on the promoter of SM22α with p-Value < 0.05. A direct transcriptional relation between ATF and SM22α is not known, but there are a few references to the fact that ATF regulates other genes, like cyclin A gene and NOX1 (in rat), in vascular SMCs [54], [55].

The binding site of TOPORS was found on the promoter of SM22α with p-Value $<$ 0.05. Any relation between TOPORS and SM22α on Vascular SMC is not known.

The binding site of VDR was also found on the promoter of SM22α with p-Value $<$ 0.05. A direct transcriptional connection between VDR and SM α-actin is not known, however, there are some references of an influence of VDR on vascular SMCs (see 8.2.1 (II)).

The rest of the TFs got from STOP p-Value > 0.5 and are assumed to be noise.

(II) I repeated the same procedure using the Colon data set. The results are in table 2. Using this data the algorithm identified SRF, YY1 and HOXB7 as SM22α TFs, but didn't identified MADH4 as SM22α TF. STOP found the binding sites of both VDR and TOPORS on the promoter of SM22α with p-Value < 0.05.

The rest of the TFs got from STOP p-Value > 0.5 and are assumed to be noise.

TF (Gene Symbol)	The Milyavsky data score	The Colon data score	STOP p-Value	Literature
SRF	3.38	5.18	< 0.05	Known
GATA2	0.09	3.59	> 0.5	
USF2	3.04	2.81	> 0.5	
HMGA1	0.24	2.51	> 0.5	
SMARCA3	-0.23	2.47	> 0.5	
HIF1A	0.93	2.209	> 0.5	
VDR	1.27	1.92	< 0.05	SMC related
NME2	-0.52	1.92	> 0.5	
SP2	0.31	1.85	> 0.5	
TOPORS	1.45	1.809	< 0.05	
RB1	1.38	1.801	> 0.5	
ATBF1	1.39	1.73	> 0.5	
ACTR1A	-4.31	1.72	> 0.5	
MEF2D	1.55	1.68	> 0.5	
HOXB7	-0.41	1.66	> 0.5	Known
NFYA	1.22	1.59	> 0.5	
ELF2	2.09	1.56	> 0.5	
NR2C1	-0.11	1.53	> 0.5	
PSMD12	1.02	1.51	> 0.5	
YY1	-5.82	1.507	> 0.5	Known
ETS2	1.57	1.501	> 0.5	
ZNF148	3.05	1.4	> 0.5	
ATF	1.95	1.28	< 0.05	SMC related
RELA	1.3	1.18	> 0.5	
ETV7	2.16	0.82	> 0.5	
BRCA1	2.17	0.75	> 0.5	
MADH4	1.77	0.64	< 0.1	Known
IRF7	1.39	-0.27	> 0.5	

Table 2: SM22α TFs. The table contains the known SM22α TFs and new TFs that passed the algorithm threshold, when running it with the TF and with SM22α, using the Milyavsky and / or the Colon data. The first column contains the TF gene symbol. The second (third) column contains the score assigned to the TF by the algorithm, using the Milyavsky (Colon) data. The fourth column contains the STOP p-Value of the appearance of the TF binding site in SM22α promoter compared to a random gene. The fifth column contains literature information, it specifies whether each TF is known to be SM22α TF (Known) or known to have a role in SMC (SMC related). Yellow TFs are TFs that are known from literature to regulate SM22α. Blue indicates new TFs that were chosen by my method to be SM22α TFs. These are TFs that both passed the algorithm threshold (for at least one of the data sets) and whose binding site appears in the SM22α promoter with high probability compared to a random promoter (according to STOP).

I checked the location of the known and new putative SM22α TFs on the promoter of SM22α, in order to try to understand if any of these TFs work together, as a complex, when regulating SM22α, or if one TF affects the regulation of SM22α by another TF. As can be seen in figure 14 the pairs that might influence one another are ATF and VDR and/or SRF and SMAD4.

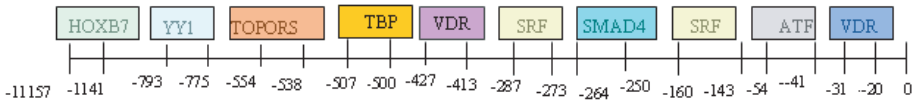

Figure 14: The promoter of SM22α. The location of each known and new putative TF of SM α-actin on the 1157 base-pairs of this promoter.

I checked which of the new SM22α putative TFs are its activators and which are its repressors. I repeated the procedure I performed in 8.2.1 in order to do so. The results,

using the Milyavsky data, can be seen in figure 15 (the Colon results are very similar to the Milyavsky results).

In figure 15 (a) we can see the gene expressions of SM22α (red), of TOPORS (green) and of TLN1 (blue), which is a known target of TOPORS. We can see that there is high correlation between TLN1 and SM22α (correlation coefficients between them is 0.77/0.75). We can also see that the correlation between SM22α and TOPORS is low (0.14/0.08) and thus we can't learn from it about the sign of the transcriptional connection between them. It is not known from the literature if TOPORS activates or represses TLN1, but I can assume that TOPORS activates both TLN1 and SM22α or represses both of them.

In figure 15 (b) we can see the gene expressions of SM22α (red), of ATF (green) and of GCA (blue), which is a known target of ATF. We can see that both the correlation between SM22α and GCA (-0.15/0.11) and between SM22α and ATF (-0.15/-0.22) is very low and thus I can't say anything about the sign of the transcriptional relation between ATF and SM22α.

In figure 15 (c) we can see the gene expressions of SM α-actin (red), of VDR (green) and of CDKN1A (blue), which is a known target of VDR. We can see that although the correlation coefficients between SM22α and CDKN1A is not very high (0.15/0.23) the behavior of their gene expression is similar in most samples. Moreover, the correlation coefficients between SM22α and VDR is high (0.53/0.53). From the combination of these facts with the fact that VDR is an activator of CDKN1A, I concluded that VDR is an activator of SM22α.

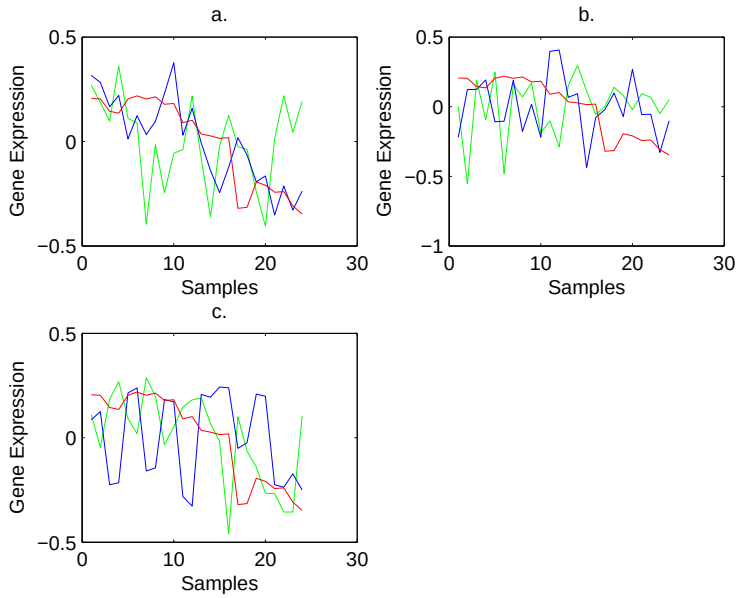

Figure 15: Gene expression of SM22α, it's new putative TFs and their known targets, using the Milyavsky data (these graphs using the Colon data are very similar). In each of the plots the x-axis contains the 24 samples and the y-axis contains the gene expression of the gene, after the following pre-processing: log2, center and normalization. the red line is SM22α gene expression, the green line is the TF gene expression and the blue line is the gene expression of the TF known target gene. (a) TF: TOPORS, known target gene: TLN1. (b) TF: ATF, known target gene: GCA. (c) TF: VDR, known target gene: CDKN1A.

8.2.3 SRF targets

I ran the algorithm, using the Milyavsky data set, with SRF as the TF and with each of the 5582 genes on the Milyavsky data set chip. I used only SM α-actin and SM22α as prior knowledge about SRF targets (and not CNN1, CALD1, ACTG2 and PDE5A), in order to make the algorithm task more difficult, and examine how it deals with little prior knowledge. In order to eliminate noise I decided to work with threshold 2 (instead of 1.2). 277 genes passed the threshold, meaning, the algorithm found 277 putative targets to SRF. Then, I ran the algorithm, using the Colon data set, with SRF as the TF and with each of the 22215 genes on the Colon data set chip. Here I also worked with threshold 2 (instead of 1.5). 3499 genes passed that threshold.

Now, I looked at the intersection of the genes that were chosen by the algorithm as SRF targets, both using the Milyavsky data and using the Colon data. There were 78 such genes. The promoter sequence of 75 of them is known.

I wanted to examine which of the promoters of the 75 genes contain the binding sequence of SRF with high probability, compared to a random promoter. To do so I worked with POC (see 7.3.2). I entered POC the 75 genes as a cluster. According to POC the p-Value of SRF as a TF of this cluster of genes is 0.0047. Moreover, 41 of

these genes' promoters contain the binding site sequence of SRF with high probability compared to a random promoter. I assume that these 41 genes are genes that are regulated by SRF, and that the rest of the 75 genes (34 genes) are noise. The 41 genes can be seen in table 4. We can see that both SM α-actin (gene symbol = ACTA2) and SM22α (gene symbol = TAGLN) have been chosen as SRF targets. We can also see that the four other genes that are known from literature to be regulated by SRF (CNN1, ACTG2, PDE5A and CALD1) have been chosen by my method as SRF targets. Eight of the remaining chosen genes: C3, SMTN, PTPLA, LIGP, ITPR1, IGFBP2, IL6R and TNFRSF21 have a role in a process related to SMC ([56], [57], [58], [59], [60], [61], [62], [63] and DAVID web site). The rest of the 27 genes are not known to have any role in SMC.

An interesting phenomenon that occurs in the group of these 41 genes is related to the location of the binding site of SRF in the promoters of the genes. In 23 genes (56 %) the binding sequence of SRF is found in the first 200 nucleotides of the promoter (the 200 nucleotides that are closest to the starting site of the gene). Moreover, five of these 23 genes are genes that are known to be SRF targets (SM22α, SM α-actin, CNN1, ACTG2 and CALD1). This can indicate that SRF usually binds to the promoter of its target genes in the first 200 nucleotides of the promoter. Therefore, the more confident target candidates of SRF are the remaining 18 genes in this group. Especially the three genes in this group that are known to have a role in SMC: C3, LIPG and IGFBP2.

There are some SRF targets that were not identified as SRF targets using this method. These targets are DES (desmin) and FOS (c-foc) [64]. The reason that they were not identified is because they do not appear on the Focus chip (the Milyavsky data). When checking the initial results of the algorithm on the Colon data, we can see that the algorithm identified DES as a putative SRF target, but didn't identify FOS as SRF target.

I now wanted to understand which of these 41 genes are activated by SRF and which are repressed by it. I looked at the gene expression of the 41 genes and the gene expression of SRF after log2, center and normalization of the data (for each of the two data sets I work with). I sorted these 42 gene expressions using SPIN (see 7.4.1).

(I) The Milyavsky data: The results can be seen in figure 16. We can see that the first 19 genes expressions are high in the first $\sim$10 samples, medium in the $\sim$4 samples and low in the last $\sim$10 samples. Since five of these 19 genes (SM22alpha, SM alpha-actin, CNN1, ACTG2 and CALD1) are known to be activated by SRF from literature, I conclude that SRF also activates all the other 14 genes that are in this group.

We can see that the last 11 genes expressions are low in the first ∼14 samples, and high in the last ∼10 samples. Since the expression pattern of this group is opposite to the expression pattern of the previous group I concluded that SRF represses this group.

There is no significant signal to the rest of the genes.

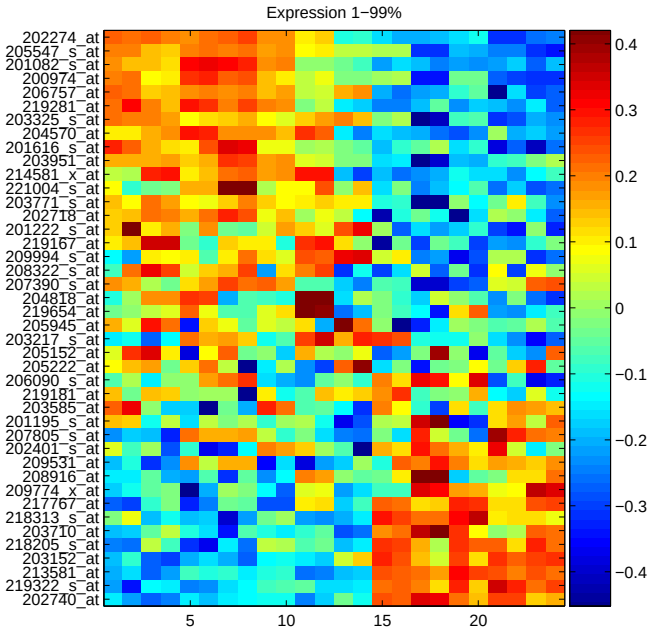

Figure 16: Sorted gene expressions of SRF and its 41 putative targets, using the Milyavsky data. We can see that the genes can be divided into three groups. The first one contains the 19 genes on the top, which are high in the first samples and low in the last samples. The second one contains the 11 genes on the bottom, which are low in the first samples and high in the last samples. The third one contains the rest of the 12 genes, which has no significant pattern in their expression. The expression of SRF is the 12 from the end (SRF probe id is 202401_*s_at*).

It is interesting to see that SRF is neither in the activated nor the repressed group, but its expression is much closer to the later (SRF probe id is 202401_*s_at*). Obviously, naively we would expect SRF to be positively correlated with genes it activates, and we seem to observe the opposite. This can be explained by the fact that the scattering of SRF gene expression is relatively low, compared to the expression of its 41 putative target genes; thus we can't learn from its expression about its relation with its target genes.

In order to make sure that this is the case I performed the following checks:

1. I plotted the gene expression of SRF and of its 41 putative target genes (figure 17). In the x-axis are the 24 samples of the data, in the y-axis are the gene expression values (with log2 of the data). In each subplot we can see 5-6 of the 41 genes' expressions in different colors and the expression of SRF in red. The 41 genes are sorted according

to the difference between maximum and minimum values of their expression; the first subplot contains genes with the lowest difference between maximum and minimum values of their expression, and the last subplot contains genes with the highest difference. We can see that the variation of SRF expression is higher than the scattering in the first subplot; the variation of SRF expression is similar to the scattering in the second subplot; the variation of SRF expression is lower than the scattering in the rest of the subplots.

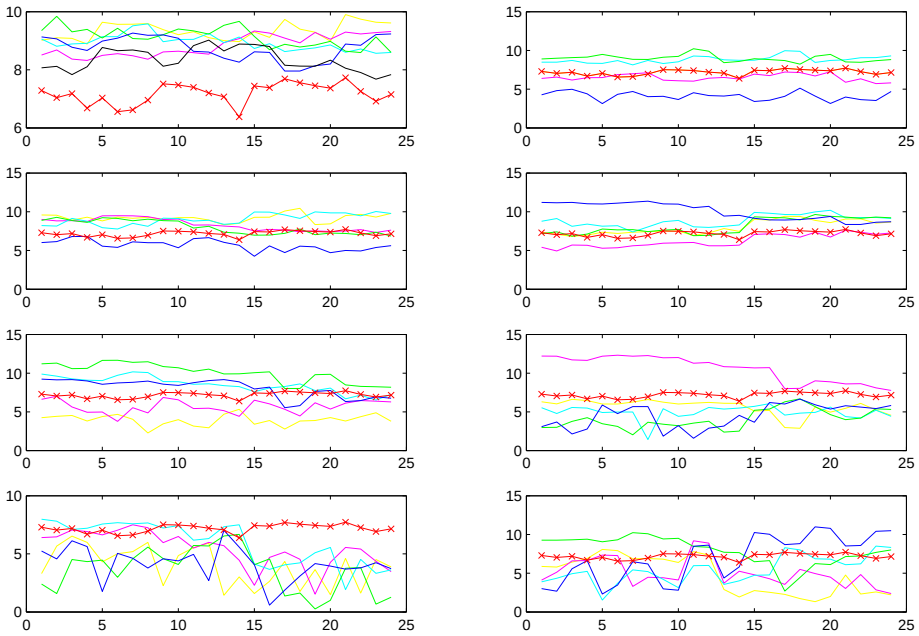

Figure 17: Gene expressions of SRF 41 putative targets compared to SRF expression. In the x-axis are the 24 samples of the data, in the y-axis are the gene expression values (with log2 of the data). In each subplot we can see 5-6 of the 41 genes expressions in different colors and the expression of SRF in red. The 41 genes are sorted according to their difference between maximum and minimum values of their gene expression; in the first subplot there are genes with the lowest difference between maximum and minimum values of their gene expression, in the last subplot there are genes with the highest difference between maximum and minimum values of their gene expression.

2. In order to determine the 'scattering' parameter of each gene, I measured for each gene the std of the gene expression using raw data and divided by its mean. Only 2 of the 41 genes got smaller results than SRF. The results can be seen in table 3.

The same results can be seen in figure 18: The x-axis contains the 41 putative SRF target genes + SRF, the y-axis contains the 'scattering' parameter of each gene. The red x represents SRF. We can see here very clearly that SRF scattering is very low compared to most of the other 41 genes.

Genes	Raw data std / mean
ITM2C	0.1968
PSMD9	0.2017
SRF	0.2268
RAD23B	0.2313
MRPL40	0.2447
SMTN	0.2624
SIAT9	0.2716
DISC1	0.306
PTPLA	0.3661
SLC6A1	0.3665
SLC1A5	0.3682
SLC7A5	0.3742
LIPG	0.3948
BLVRA	0.4272
LOC51285	0.4511
EHHADH	0.4542
MKNK2	0.4638
ZNF185	0.5208
GALNT7	0.5228
DCTN1	0.5245
COL5A1	0.538
WDR8	0.5565
MSRA	0.5786
ACY1	0.6083
ACTG2	0.6349
CALD1	0.6488
PDCD2	0.6525
ACTA2	0.6685
TAGLN	0.7395
IGFBP2	0.7556
GSTZ1	0.7721
PDE5A	0.7757
CNN1	0.8173
SIAT4A	0.9165
ITPR1	0.9762
TNFRSF21	0.9884
IL6R	1.0116
ABCB1	1.1201
COX7A1	1.1303
CXCL2	1.2462
C3	1.2627
HSD17B2	1.7468

Table 3: 'Scattering' parameter of SRF and of its 41 putative targets. The first column contains the gene symbol of each gene; the second column contains the gene's 'scattering' parameter. We can see that only two genes (the ones in blue) have lower 'scattering' parameter then SRF (the one in yellow).

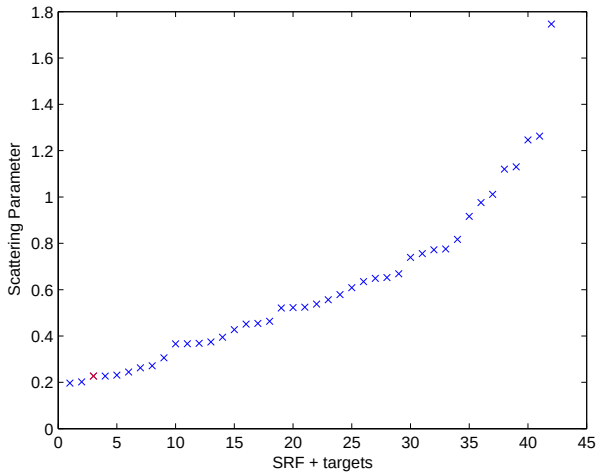

Figure 18: Plot of the 'scattering' parameter of SRF and of its 41 putative targets. The x-axis contain SRF and its 41 putative target genes, the y-axis contains the 'scattering' parameter of each gene. The red x is SRF point in this plain.

From the last two tests we can indeed conclude that we can't learn about the relation between SRF and its putative targets from its expression. This explains the fact that its expression pattern goes in the opposite direction to that of the group of 19 genes, that it is assumed to activate.

(II) The Colon data: The results can be seen in figure 19. We can see that the first 19 genes expressions are high in the first ∼21 samples, low in the next ∼11 samples and high again in the last ∼6 samples. Since four of these 19 genes (SM22α, SM α-actin, CNN1 and ACTG2) are known from literature to be activated by SRF, I concluded that the rest of the 15 genes are also activated by SRF. We can see that the last 15 genes expressions are of opposite pattern to the previous group and thus I concluded that these 15 genes are repressed by SRF. The rest of the genes don't have any significant pattern. SRF is one of these genes. But in this data, we can see that its expression pattern seems more like the group of genes it activates (SRF probe id is 202401_*s_at*).

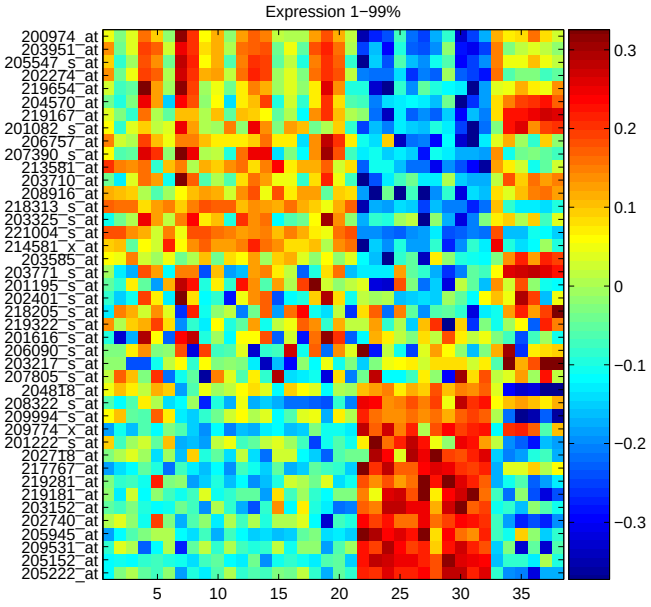

Figure 19: Sorted gene expressions of SRF and its 41 putative targets, using the Colon data. We can see that the genes can be divided into three groups. The first one contains the 19 genes on the top. The second one contains the 15 genes on the bottom. The third one contains the rest of the eight genes.

I assumed that the 15 genes that were in the activated genes group in both data sets or were in the activated genes group in one data set, and in the non significant group in the other data set are activated by SRF.

In the same way, I assumed that the 13 genes that were in the repressed genes group in both data sets or were in the repressed genes group in one data set, and in the non

significant group in the other data set are repressed by SRF.

I can't learn whether SRF actives or represses the 3 genes that were in the non significant group in both data sets, and the 10 genes that were in the activated genes group in one data set and in the repressed genes group in the other data sets

(the list of activated / repressed genes can bee viewed in table 4, last column).

Gene Symbol	Milyavsky data score	Colon data score	Promoter location	POC score	Literature	Activator / Repressor
CNN1	**2.28**	**6.28**	**100**	**-19.2919**	**Known**	**a**
TAGLN	**3.41**	**5.7**	**104**	**-16.1157**	**Known**	**a**
GALNT7	**2**	**4.91**	**124**	**-19.2647**		**n**
EHHADH	**2.51**	**4.15**	**74**	**-19.1314**		**r**
ACTG2	**4.21**	**4.13**	**105**	**-17.451**	**Known**	**a**
ACTA2	**4.2**	**3.82**	**85**	**-16.5413**	**Known**	**a**
C3	**2.16**	**3.45**	**85**	**-17.7349**	**SMC related**	**r**
SMTN	3.3	3.24	729	-16.5391	SMC related	a
IL6R	2.09	3.21	893	-17.8924	SMC related	r
TNFRSF21	2.29	3.2	975	-18.4166	SMC related	a
GSTZ1	2.14	3.16	670	-17.9371		r
PTPLA	2	3.13	321	-19.2333	SMC related	a
SLC6A1	3.12	3.1	609	-16.674		r
ITM2C	2.2	3.09	890	-18.6114		a
LIPG	**3.61**	**2.95**	**85**	**-16.7142**	**SMC related**	**r**
COL5A1	2.2	2.72	468	-18.9202		a
SIAT4A	**2.001**	**2.68**	**104**	**-19.217**		**n**
MSRA	2.04	2.62	621	-19.2633		n
LOC51285	**2.36**	**2.62**	**146**	**-19.4977**		**n**
ITPR1	2	2.53	314	-19.5477	SMC related	n
MRPL40	2.34	2.51	652	-16.7735		r
DCTN1	**3.38**	**2.47**	**21**	**-18.7972**		**a**
IGFBP2	**2.45**	**2.44**	**59**	**-18.8373**	**SMC related**	**r**
COX7A1	2	2.36	938	-17.6874		a
SLC1A5	**2**	**2.35**	**191**	**-17.2753**		**n**
MKNK2	**2**	**2.35**	**120**	**-18.7386**		**r**
SIAT9	**2.002**	**2.35**	**139**	**-18.0968**		**n**
ACY1	2.02	2.33	874	-18.2305		r
HSD17B2	**2.11**	**2.28**	**151**	**-19.223**		**n**
CXCL2	**2.3**	**2.23**	**145**	**-18.849**		**r**
DISC1	**2.96**	**2.22**	**11**	**-18.5476**		**n**
WDR8	**2.1**	**2.2**	**205**	**-19.2844**		**r**
PDE5A	2.05	2.19	310	-19.3478	Known	a
CALD1	**2.01**	**2.1**	**77**	**-16.8938**	**Known**	**a**
ZNF185	**2.27**	**2.07**	**109**	**-17.4073**		**a**
RAD23B	**2.36**	**2.06**	**121**	**-19.2545**		**n**
BLVRA	2.06	2.04	611	-19.0106		a
PDCD2	2.28	2.04	832	-18.4967		n
ABCB1	**2**	**2.02**	**89**	**-18.1524**		**n**
SLC7A5	2.4	2.02	647	-17.274		n
PSMD9	2.39	2.01	416	-19.5687		r

Table 4: The 41 genes that were chosen to be SRF targets. The first column contains the genes symbols. The second (third) column contains the score that each gene got from the algorithm using the Milyavsky (Colon) data. The forth column contains the location of the binding site of SRF on the gene promoter. The fifth column contains POC scores (the higher the score, the better it is). The sixth column contains literature information, it specifies whether each gene is known to be SRF target (Known) or known to have a role in SMC (SMC related). The seventh column contains information about the sign of the transcriptional relation: a - activator; r - repressor; n - unknown. The yellow genes are the ones that are known to be SRF targets. The bold genes are the ones which SRF binds to their promoters in locations 0-205.

To conclude, the network I learned can be seen in figure 20. The blue connections are the new ones.

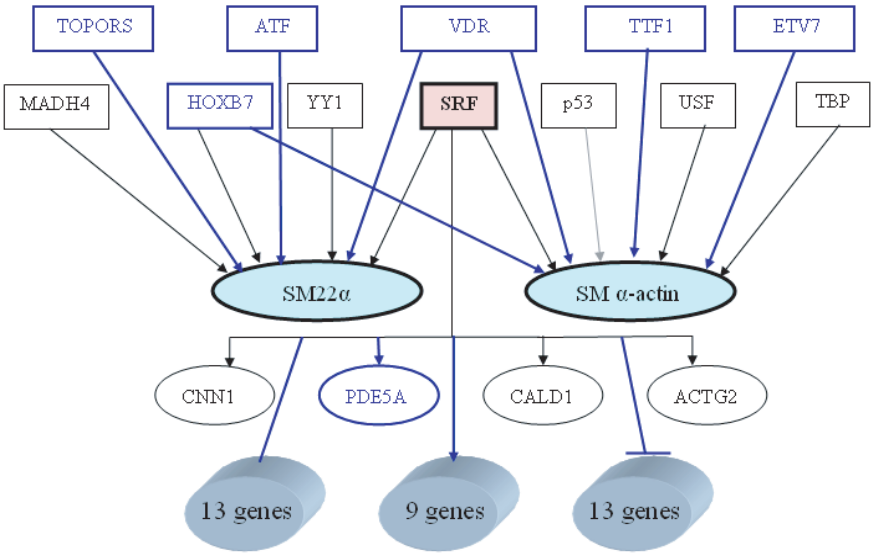

Figure 20: The SRF, SM α-actin and SM22α network. In the squares are the TFs and in the circles are the target genes. A black connection represent a prior known connection (see figure 11) and a blue connection represents a new connection, that was learned using my method. An arrow represents an activation connection, a broken line represents a repression connection and a clean line represents an unknown relation.

9 Discussion

In this work I developed a method for reconstructing a local transcriptional network. In order to use this method, one needs to have some prior knowledge about the network. This method uses gene expression microarray data, information about transcription factors binding sites and promoter sequences of genes. The main idea behind this method is that the relation, measured from gene expression data by CC and MI, between a TF and its target genes is very similar to the relation between two random genes, and thus, one can't learn about the direct transcriptional connection between two genes only from measuring the CC and / or MI between them. However, it is possible to learn whether a TF regulates a gene from gene expression data. To do so one needs to look at the relation (again, using CC and MI) between the gene and other genes, that are known to be regulated by that TF. I showed, using comparison between distribution functions, that in this case, the higher (lower) the relation between the gene checked and the TF known target genes, the higher (lower) the probability that the TF regulates the gene. Moreover, if the checked TF has known target genes that are also regulated by another TF that is known to regulate the checked gene, the relation between these target genes and the checked gene, can teach us even more about the transcriptional connection between the TF and the gene: If the TF regulates the checked gene, the relation between that checked gene and these target genes will be even higher (in most

cases) than the relation between the checked gene and target genes that don't share another mutual TF with the checked gene. All the previous observations are based on a database of known human TF-target pairs I built from literature.

In the center of my method, lays a classifier, that receives a TF and a gene and specifies wether the TF regulates the gene. It uses gene expression microarray data to do so. In order to decide if the TF regulates the gene, the classifier calculates a score and checks whether the score passes a certain threshold. The score is combined of six components that describe the relation between the TF and the gene, the relation between known TF targets and the gene and the relation between known TF targets that share another TF with the gene and the gene. These relations are measured by both CC and MI. The threshold and the weights of the score components are determined separately to each gene expression data set, using the delta rule learning algorithm.

If the classifier claims that the TF regulates the gene, this method will then check whether the binding site of the TF exists on the promoter of the gene, with high probability compared to a random promoter. Only if this is the case my method assumes that the TF indeed regulates the gene.

This method is different from existing methods for reverse engineering of transcriptional networks in a few ways. First of all, most methods look at the entire genome globally and try to reconstruct the complete transcriptional network, whereas my method aims to reconstruct a local network. To do so, it asks a local question, about a transcriptional connection between a TF and a gene, at a time. The advantage of my approach is that most of the times one wants to search for information about a specific transcriptional network that interests him, and go deeply into that specific transcriptional network.

Moreover, my method uses both the relation between the TF and the gene and the relation between genes that are co-regulated, whereas other methods use the relation between co-regulated genes ([4]) or the relation between TFs and genes ([25]), but not both relations.

Another difference between my method and other methods is that I use both correlation coefficient and mutual information in order to measure the relation between two genes, while others use only correlation coefficient. Mutual information is stronger than correlation coefficient, since it measures the non linear relation between two genes, while correlation coefficient measures only the linear relation between two genes.

My method has some weaknesses. The first, and most central one is the fact that prior knowledge about the network must be provided. Another drawback of this method

is the fact that it is very sensitive to noise, since it looks at specific gene expressions from the microarray chip, and not on clusters of genes. The technology of microarray chips is sensitive to noise, and when working with large groups of genes (clusters of genes) that noise is eliminated, however, if we work with single gene expression, that noise is still there.

In order to build and check this method I used 2 human microarray data sets. One of these data sets contains in-vitro human cells that passed a series of transformations. The other one contains samples of normal human tissues, taken from colon, liver and lung.

First, I learned the algorithm parameters for both data sets. In both cases the weights that were specified to the score components by the learning algorithm strengthen my key assumptions, since the learned weights of the relation between the TF and the gene were very low, whereas the learned weights of the relation between the gene and the known targets of the TF were higher, and the learned weights of the relation between the gene and the known targets of the TF that share another TF with the gene were even higher.

I also checked the error rate of the algorithm with the learned parameters, and discovered that in both data sets the false negatives rate was 60%. In one of the data sets the false positives rate was 10% and in the other data set the false positives rate was 20%. These results can teach us that the algorithm is noisy, moreover it recognizes only part of the direct transcriptional connections between TFs and genes. One way that reduces the noise of the algorithm tremendously is the fact that if the algorithm predicts that a TF regulates a gene, I check whether the TF binding site exists in the promoter of the gene, and only if this is the case I assume that the algorithm prediction was true. Not surprisingly, when I checked the algorithm on known transcriptional networks, which I had a lot of prior information on, I saw that the algorithm discovered more than 40% of the known transcriptional connections, and the rate of the connections that were discovered falsely was less than 10%. Meaning, there is an anti-correlation between the prior knowledge of the algorithm and its error rate; when the prior knowledge on the network is high, the error rate of the algorithm is low and we can be more confident about the algorithm results, and vise-versa.

I compared this error rate to error rates of other methods that reconstruct transcriptional networks. Qian *et. al.* [25] got false negative rate of 64% and the false positives rate of 1.8%. Xing *et. al.* [33] got false positive rate of 1% - 6.5%, and overall (false positives and false negatives) rate of 2% - 33%, depending on the error of the system

they developed, and the experiments number they used.

Segal *et. al.* [4] and Ihmels *et. al.* [26] didn't provide direct error rate measurements.

I used this method in order to reconstruct a human transcriptional network, related to SMC differentiation, around the TF SRF and two of its target genes: SM α-actin and SM22α. I wanted to discover the TFs that regulate SM α-actin and /or SM22α and the target genes of SRF. I worked with 54 TFs which I have prior information on. My method identified two of the four known SM α-actin TFs (SRF and USF). These TFs were identified by both of the data sets. Moreover, it discovered four new TFs as SM α-actin regulators (ETV7, TTF1, VDR and HOXB7). Non of these TFs is known from literature to regulate SM α-actin. One of these four TFs (HOXB7) was identified using both of the data sets, the rest of the three TFs were found by one of the data sets. My method also identified one of the four SM22α known TFs (SRF), using both data sets. It identified the other three known SM22α TFs when using only one of the data sets. Furthermore, this method identified three SM22α new TFs (VDR, ATF, TOPORS). Two of these three TFs (VDR and TOPORS) were discovered by both data sets.

In order to learn which of the new TFs are activators and which are repressors I looked at the correlation coefficients (now without absolute value) of SM α-actin / SM22α with the new TFs and with the known target genes of these new TFs. My results here were partial. I discovered that VDR activates both SM α-actin and SM22α, and HOXB7 activates SM α-actin. I couldn't draw any conclusions about the rest of the new TFs, since I didn't have any information about whether they activate / repress their known target genes, and I couldn't rely on the correlation coefficient between the TF and SM α-actin / SM22α in order to examine the regulation relation between them.

Next, I checked which of the genes in each data chip are SRF targets. My method found 41 putative SRF target genes. These genes were found using both of the data sets. Six of these 41 genes are known to be SRF targets (five of them are known to be activated by SRF). In order to check which of these 41 genes are activators and which are repressors, I looked at their expression patterns, using both data sets. In each data set the 41 genes could be divided to three groups (in each data set the groups were different): One group with a certain expression pattern, another group with the opposite pattern and a third group with no pattern. In each data set one of the groups with expression pattern contained at least four of the genes that are known to be activated by SRF. Thus, I concluded that the genes in this group are activated by SRF and the genes in the group with the opposite expression pattern are repressed by SRF. 10 of the 41 genes turned out to be activated by SRF using one data set and repressed by SRF by another

data set, and thus I couldn't conclude the regulation sign between SRF and those genes. The rest of the 31 contained 15 activated genes (were in the activated genes group for at least one data set), 13 repressed genes (were in the repressed genes group for at least one data set) and three not known genes (were in the group without expression signal in both data sets).

We can see that this method identifies known transcriptional relations, as well as new ones. It would be very interesting to perform a biological experiment in order to verify some of the new connections I found.

Moreover, it would be interesting to examine more gene expression data sets in order to verify that the characters of the distribution functions are carried out for other data sets.

Another goal for the future is to expend the human database of TF-target pairs in order to be able to work with more TFs.

References

[1] N. Friedman and D. Koller. Learning bayesian networks from data. *Presentation.*

[2] N. Friedman, M. Linial, I. Nachman, and D. Peer. Using bayesian networks to analize expression data. *Proceedings of the Fourth Annual International Conference on Computational Molecular Biology.*, 2000.

[3] D. Peer, A. Regev, G. Elidon, and N. Friedman. Inferring subnetworks from pertubated expression profiles. *Bioinformatics*, 17(1):215–224, 2001.

[4] E. Segal, R. Yelensky, and D. Koller. Genome-wide discovery of transcriptional modules from dna sequence and gene expression. *Bioinformatics*, 19:273–282, 2003.

[5] K. Sachs, O. Perez, D. Peer, D.A. Lauffenburger, and G.P. Nolan. Causal protein-signaling networks derived from multiparameter single-cell data. *Science*, 308:523–529, 2005.

[6] L.A. Herzenberg, D. Parks, B. Sahaf, O. Perez, and M. Reederer. The history and future of the fluorescence activated cell sorter and flow cytometry: a view from stanford. *Clin. Chem.*, 48(10):1819–1827, 2002.

[7] O.D. Perez and G.P. Nolan. Simultaneous measurement of multiple active kinase states using polychromatic flow cytometry. *Nat Biotechnol.*, 20(2):155–162, 2002.

[8] T. Mitchell and H. McGraw. *Machine Learning*. 1997.

[9] S. Dasgupta. The sample complexity of learning fixed-structure bayesian nets. *Machine Learning*, 29(2-3):165–180, 1997.

[10] N. Friedman and Z. Yakhini. On the sample complexity of bayesian networks. In *UAI*, pages 274–282, 1996.

[11] S. Kullback and R. A. Leibler. On information and sufficiency. *Ann. Math. Statist.*, 22:79–86, 1951.

[12] R. Greiner, A. Grove, and D. Schuurmans. Learning bayesian nets that perform well. In *UAI*, pages 198–207, Aug 1997.

[13] D. Haughton. On the choise of a model to fit data from an exponential family. *Thee Annals of Statistics*, 16(1):342–355, 1988.

[14] D. Haughton. Size of the error in the choice of a model to fit data from an exponential family. *The Indian Journal of Statistics*, 51:45–58, 1989.

[15] D.M. Chickering and C. Meek. Finding optimal bayesian networks. In *UAI*, pages 94–102, 2002.

[16] D. Geiger, D. Heckerman, H. King, and C. Meek. Stratified exponential families: graphical models and model selection. *The Annals of Statistics*, 29(2):505–529, 2001.

[17] J. Pearl. *Probabilistic reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufman, San Francisco Calif, 1988.

[18] D. M. Chickering. Learning equivalence classes of bayesian-network structures. *The Journal of Machine Learning Research*, 2:445 – 498, 2002.

[19] J. Rissanen. Modeling by shortest data description. *Automatica*, 14:465–471, 1978.

[20] G. Schwarz. Estimating the dimension of a model. *Ann. Statist*, 6(2):461–464, 1978.

[21] T. M. Cover and J.M. Thomas. *Elements of Information Theory*. John Wiley, New York, NY, 1991.

[22] B. Alberts, D. Bray, J. Lewis, M. Raff, K. Roberts, and J. Watson. *Molecular Biology of the Cell*, 1994.

[23] R. Tupler, G. Perini, M.A. Pellegrino, and M.R. Green. Profound misregulation of muscle-specific gene expression in facioscapulohumeral muscular dystrophy. *Proc. Natl Acid. Sci.*, 96:12650 – 12654, 1999.

[24] D.H. Ly, R.A. Lockhart, and P.G. Schultz. Mitotic misregulation and human aging. *Science*, 287:2468–2492, 2000.

[25] J. Qian, J. Lin, N.M Luscombe, H. Yu, and M. Gerstein. Prediction of regulatory networks: genome-wide identification of transcription factor targets from gene expression data. *Bioinformatics*, 19:1917–2196, 2003.

[26] J. Ihmels, S. Bergmann, and N. Barkai. Defining transcription modules using large-scale gene expression data. *Bioinformatics*, 20:1993–2003, 2004.

[27] S. Bergmann, J. Ihmels, and N. Barkai. Iterative signature algorithm for the analysis of large-scale gene expression data. *Physical review*, 67:031902–1 – 031902–18, 2003.

[28] A. Brazma, I. Jonassen, J. Vilo, and E. Ukkonen. Predicting gene regulatory elements in silico on a genomic scale. *Genome Res.*, 8:1202–1215, 1998.

[29] F. Roth, P. Hughes, J.D. Estep, and G. Church. Finding dna regulatory motifs within unaligned noncoding sequences clustered by whole-genome mrna quantitation. *Nat. Biotechnol.*, 16:939–945, 1998.

[30] X. Liu, D. Brutlag, and J. Liu. Bioprospector: discovering conserved dna motifs in upstream regulatory regions of co-expressed genes. *Pac. Symp. Biocomput.*, pages 127–138, 2001.

[31] S. Sinha and M. Tompa. A statistical method for finding transcription factor binding sites. *Proc. Int. Conf. Intell. Syst. Mol. Biol.*, 8:344–354, 2000.

[32] S. Tavazoie, J.D. Hughes, M.J. Campbell, R.J. Cho, and G.M Church. Systematic determination of genetic network architecture. *Nat. Genet.*, 22:281–285, 1999.

[33] B. Xing and M.J. Van Der Laan. A statistical method for constructing transcriptional regulatory networks using gene expression and sequence data. *Journal of computational biology*, 12(2):229–246, 2005.

[34] L. Hertzberg, O. Zuk, G. Getz, and E. Domany. Finding motifs in promoter regions. 2003.

[35] Y. Tabach. Revealing the regulatory network underlying carcinogenesis in human immortalized fibroblast. *Master Thesis*, 2004.

[36] T.L. Bailey and C. Elkan. Fitting a mixture model by expectation maximization to discover mitifs in biopolymers. *Proc. Int. Conf. Intell. Syst. Mol. Biol.*, 2:28–36, 1994.

[37] K. Quandt, K. Frech, H. Karas, E. Wingender, and T. Werner. Matind and matinspector: new fast and versatile tools for detection of consensus matches in nucleotide sequence data. *Nucleic Acids Res*, 23:4878–4884, 1995.

[38] D. Tsafrir, W. Liu, Y. Yamaguchi, I. Tsafrir, Y. Wen, W. Gerald, R. Stengel, F. Barany, P. Paty, E. Domany, and D. Notterman. A novel mathematical approach to analyzing gene expression data: results from an international colon cancer consortium. *submitted*, 2004.

[39] M. Milyavsky, I. Shats, N. Erez, X. Tang, S. Senderovich, A. Meerson, Y. Tabach, N. Goldfinger, D. Ginsberg, C.C. Harris, and V. Rotter. Prolonged culture of telomerase-immortalized human fibroblasts leads to a premalignant phenotype. *Cancer Res*, 63:7147–7157, 2003.

[40] P. Carmeliet. Mechanisms of angiogenesis and arteriogenesis. *Nature Med.*, 6:389395, 2000.

[41] L. Li, J.M. Miano, P. Cserjesi, and E.N. Olson. Sm22, a marker of adult smooth muscle, is expressed in multiple myogenic lineages during embryogenesis. *Circulation Research*, 78:188–195, 1996.

[42] M.S. Kumar and G.K. Owens. Combinatorial control of smooth muscle-specific gene expression. *Arterioscler Thromb Vasc Biol.*, 23:737–747, 2003.

[43] G.K. Owens. Molecular control of vascular smooth muscle cell differentiation. *Acta Physiol Scand.*, 164(4):623–635, 1998.

[44] V. Fatigati and R.A. Murphy. Actin and tropomyosin variants in smooth muscles: dependence on tissue type. *J Biol Chem.*, 259:1438314388, 1984.

[45] J.L. Duband, M. Gimona, S. Scatena, S. Sartore, and J.V. Small. Calponin and sm 22 as differentiation markers of smooth muscle: spatiotemporal distribution during avian embryonic development. *Differentiation*, 55:111, 1993.

[46] S. Chen, M. Kulik, and R.J. Lechleider. Smad proteins regulate transcriptional induction of the sm22alpha gene by tgf-beta. *Nucleic Acids Res.*, 31(4):1302–1310, 2003.

[47] K. Stolle, B. Weitkamp, J. Rauterberg, S. Lorkowski, and P. Cullen. Laser microdissection-based analysis of mrna expression in human coronary arteries with intimal thickening. *J Histochem Cytochem.*, 52(11):1151–1158, 2004.

[48] J.A. Carson, D.E. Culberson, R.W. Thompson, R.A. Fillmore, and W. Zimmer. Smooth muscle gamma-actin promoter regulation by rhoa and serum response factor signaling. *Biochim Biophys Acta.*, 1628(2):133–139, 2003.

[49] D. Kim, T. Aizawa, H. Wei, X. Pi, S.D. Rybalkin, B.C. Berk, and C. Yan. Angiotensin ii increases phosphodiesterase 5a expression in vascular smooth muscle cells: a mechanism by which angiotensin ii antagonizes cgmp signaling. *J Mol Cell Cardiol.*, 38(1):175–184, 2005.

[50] T. Momiyama, K Hayashi, H. Obata, Y. Chimori, T. Nishida, T. Ito, W. Kamiike, H. Matsuda, and K. Sobue. Functional involvement of serum response factor in the transcriptional regulation of caldesmon gene. *Biochem Biophys Res Commun.*, 242(2):429–435, 1998.

[51] N.C. Green, I. Rambaldi, J. Teakles, and M.S. Featherstone. A conserved c-terminal domain in pbx increases dna binding by the pbx homeodomain and is not a primary site of contact for the ypwm motif of hoxa1. *J Biol Chem.*, 273(21):13273–13279, 2002.

[52] M.C. Rebsamen, J. Sun, A.W. Norman, and J.K. Liao. 1 ,25-dihydroxyvitamin d3 induces vascular smooth muscle cell migration via activation of phosphatidylinositol 3-kinase. *Circ Res.*, 91(1):17–24, 2002.

[53] M. Pfeifer B. Begerow H.W. Minne. Vitamin d and muscle function. *Osteoporos Int.*, 13(3):187–194, 2002.

[54] K. Iijima, M. Yoshizumi, M. Hashimoto, S. Kim, M. Eto, J. Ako, Y.Q. Liang, N. Sudoh, K. Hosoda, K. Nakahara, K. Toba, and Y. Ouchi. Red wine polyphenols inhibit proliferation of vascular smooth muscle cells and downregulate expression of cyclin a gene. *Circulation.*, 101(7):805–811, 2000.

[55] M. Katsuyama, C. Fan, N. Arakawa, T. Nishinaka, M. Miyagishi, K. Taira, and C. Yabe-Nishimura. Essential role of atf-1 in induction of nox1, a catalytic subunit of nadph oxidase: involvement of mitochondrial respiratory chain. *Biochem J.*, 386(2):255–61, 2005.

[56] Z.H. Lin, N. Fukuda, X.Q. Jin, E.H. Yao, T. Ueno, M. Endo, S. Saito, K. Matsumoto, and H. Mugishima. Complement 3 is involved in the synthetic phenotype and exaggerated growth of vascular smooth muscle cells from spontaneously hypertensive rats. *Hypertension.*, 44(1):42–47, 2004.

[57] Y. Kojma, K. Hirata, T. Ishida, Y. Shimokawa, N. Inoue, S. Kawashima, T. Quertermous, and M. Yokoyama. Endothelial lipase modulates monocyte adhesion to the vessel wall. a potential role in inflammation. *J Biol Chem.*, 279(52):54032–54038, 2004.

[58] W. Xiaohong, Y. Jun, T. Lijia, S. Jingyi, T. Chaoshu, and L. Naikui. C-type natriuretic peptide inhibits upregulation of alpha1-adrenoceptor and inositol 1,4,5-trisphosphate receptor in rat vascular smooth muscle after vascular endothelial injury. *Chin Med Sci J.*, 15(2):73–78, 2000.

[59] J. Dulak, A. Jozkowicz, R. Foresti, A. Kasza, M. Frick, I. Huk, C.J. Green, O. Pachinger, F. Weidinger, and R. Motterlini. Heme oxygenase activity modulates vascular endothelial growth factor synthesis in vascular smooth muscle cells. *Antioxid Redox Signal.*, 4(2):229–240, 2002.

[60] T. Hsieh, R.E. Gordon, D.R. Clemmons, W.H.Jr. Busby, and C. Duan. Regulation of vascular smooth muscle cell responses to insulin-like growth factor (igf)-i by local igf-binding proteins. *J Biol Chem.*, 278(44):42886–42892, 2003.

[61] P. Guerin, V. Sauzeau, M. Rolli-Derkinderen, O. Al Habbash, E. Scalbert, D. Crochet, P. Pacaud, and G. Loirand. Stent implantation activates rhoa in human arteries: inhibitory effect of rapamycin. *J Vasc Res.*, 42(1):21–28, 2005.

[62] J. Mullberg, T. Geib, T. Jostock, S.H. Hoischen, P. Vollmer, N. Voltz, D. Heinz, P.R. Galle, M. Klouche, and S. Rose-John. Il-6 receptor independent stimulation of human gp130 by viral il-6. *J Immunol.*, 164(9):4672–4677, 2000.

[63] A. Niemann-Jonsson, M.P. Ares, Z.Q. Yan, D.X. Bu, G.N. Fredrikson, L. Branen, I. Porn-Ares, A.H. Nilsson, and J. Nilsson. Increased rate of apoptosis in inti-

mal arterial smooth muscle cells through endogenous activation of tnf receptors. *Arterioscler Thromb Vasc Biol.*, 21(12):1909–1914, 2001.

[64] G.K. Owens, M.S. Kumar MS, and B.R. Wamhoff. Molecular regulation of vascular smooth muscle cell differentiation in development and disease. *Physiol Rev.*, 84(3):767–801, 2004.