Reverse Engineering Gene Interaction Networks
Using the Phi-Mixing Coefficient

Nitin Kumar Singh¹, M. Eren Ahsen¹, Shiva Mankala¹,
Michael A. White² and M. Vidyasagar¹ *

August 21, 2012

Abstract

In this paper, we present a new algorithm for reverse-engineering gene interaction networks (GINs) from expression data, by viewing the expression levels of various genes as coupled random variables. The algorithm is based on using the so-called phi-mixing coefficient between two random variables as a measure of the dependence between them. Unlike existing methods, the GINs constructed using the algorithm presented here have edges that are both directed and weighted. Thus it is possible to infer both the direction as well as the strength of the interaction between genes. The GIN constructed is potentially a minimal network that is compatible with the data. Several GINs have been constructed for various data sets in lung and ovarian cancer. One of the lung cancer networks is validated by comparing its predictions against the output of ChIP-seq data. The neighbors of three transcription factors (ASCL1, PPARG and NKX2-1) are significantly enriched with ChIP-seq genes compared to pure chance.

1 Introduction

Recent advances in experimental techniques in biology have made it possible to measure, in a common experimental setting, the expression levels of almost all the genes or gene products in an organism. Such experiments are referred to as whole-genome expression studies and the outcomes of such experiments are referred to as whole-genome expression data. Within a cell, all of these gene products interact in highly complex ways to facilitate the functioning of

¹Bioengineering Department, The University of Texas at Dallas, 800 W. Campbell Road, Richardson, TX 75080-3021, ²Cell Biology Department, UT Southwestern Medical Center, 5323 Harry Hines Boulevard, Dallas, TX 75390; Emails: {Nitin.Singh, Ahsen, Shiva.Mankala, M.Vidyasagar}@utdallas.edu, Michael.White@utsouthwestern.edu. The work of the UTD authors was supported by the National Science Foundation under Award #1001643, the Cecil H. and Ida Green Endowment, and a developmental grant from the Simmons Cancer Center at UT SW. The work of Michael A. White was supported by the Welch Foundation Grant #I-1414 and the National Cancer Institute Grant #CA71443.
These interactions are referred to by a variety of terms, with ‘gene regulatory networks’ being perhaps the most common. For reasons explained below, in this paper we prefer to use the expression ‘gene interaction network (GIN)’. In order to turn the whole-genome expression data into a corresponding interaction network, one of the most commonly used approaches is to view the expression level of each gene (or gene product) as a random variable, and the measurements of the gene expression level as independent samples of that random variable. To make the idea clear, let $n$ represent the number of genes and $m$ the number of samples. Then the data at hand can be viewed as an array $\{x_{ij}, i = 1, \ldots, n, j = 1, \ldots, m\}$. With this notation, $X_1, \ldots, X_n$ are the random variables corresponding to the expression values of genes 1 through $n$, while the data sets $(x_{1j}, \ldots, x_{nj})$ for various values of $j$ are independent measurements of $(X_1, \ldots, X_n)$. Notice however that it is not assumed that the random variables $X_i$ are independent of each other. Indeed, the objective of the exercise is to infer their interdependence from the available data.

If we had sufficiently many samples, we could in principle make a fairly reliable estimate of the joint distribution of all $n$ random variables. In practice however, the number of genes $n$ is in the tens of thousands, whereas the number of samples $m$ is in the hundreds at best. Hence it is not possible to infer anything remotely resembling the joint probability distribution of all random variables. Therefore we must settle for a more limited objective. At a very coarse level, we can simply ask whether, for two distinct indices $i$ and $j$, the corresponding random variables $X_i$ and $X_j$ are independent. It is common to represent the GIN as a directed graph, where the nodes correspond to genes and the edges denote interactions between them. Thus if $X_i$ and $X_j$ are independent random variables, then genes $i$ and $j$ do not interact at all, and there does not exist any path between the associated nodes $i$ and $j$ in the GIN. However, this is far too coarse a representation. At the next level of detail, one can choose three indices $i, j, k$ and ask whether $X_i$ and $X_j$ are conditionally independent given $X_k$. If the answer is ‘yes’, then this would mean that in the associated GIN, the removal of node $k$ and associated edges would disconnect nodes $i$ and $j$. Therefore all paths from node $i$ to node $j$ and vice versa must pass through node $k$. Or to put it another way, if $X_i$ and $X_j$ are conditionally independent given $X_k$, then gene $i$ does indeed interact with gene $j$, but in an indirect fashion, via gene $k$. It is therefore meaningful to ask: Given a set of whole-genome expression data, what is a minimal interaction network that is consistent with the data, in terms of faithfully reproducing the all the conditional independence properties implied by the data? In the present paper, we present an algorithm that answers this question. The network constructed using the algorithm given here is consistent with the data, while the removal of even a single edge would render it inconsistent, unless other edges are introduced to compensate. In this very specific sense, our algorithm produces a ‘minimal’ GIN that is compatible

\[1\]Hereafter we shall drop the cumbersome expression ‘gene or gene product’ and write just ‘gene’. In biology there is of course a vast difference between a gene and the protein(s) produced by that gene. However, within the narrow scope of constructing regulatory or interaction networks, it is acceptable within the biology community to ignore the distinction.
Ultimately any reverse-engineered GIN is only our guess as to what is really going on within a cell. The only way to validate such a GIN would be to compare it against ‘reality’. A typical GIN consists of tens of thousands of nodes and several hundred thousand or perhaps a few million edges. There is simply no way to know what the ‘reality’ is for such a huge network. Moreover, the measurements are a population average over an ensemble of cells, whereas in reality the interactions could vary from one cell to another, especially in the case of cancer tissue. At the present state of experimentation it is possible to determine ‘reality’ only locally, that is, to determine all (or at least many of) the upstream and downstream nodes of a specified gene. This might be referred to as a ‘first-order neighborhood’ of the node (gene) of interest. This situation also corresponds with the manner in which the biology community does its research. A typical biological researcher would mostly be interested in knowing all the upstream and downstream nodes of one very specific node (gene) that interests him/her. By undertaking very elaborate experimentation, each research group can achieve some measure of success of determining the first-order neighborhood of their gene of interest with some degree of accuracy. Different research groups would of course be interested in different genes. However, even if we were to combine all the available information from various laboratories, at best we would get a glimpse of the ‘real’ GIN around a few dozen nodes.\footnote{Recall that our interest is in constructing GINs that reflect data from a common experimental setting. Therefore, while the biology literature contains hundreds or perhaps even thousands of inferred subgraphs, for a common experimental setting the number of such subgraphs would be a dozen or so.} Thus any validation of the reverse-engineered GIN would have to consist of how faithfully these known first-order neighborhoods overlap with the predictions of the GIN, and computing the probability (or likelihood) that the overlap was achieved purely through chance; this is the so-called $P$-value widely referred to in biological circles.

To date we have used our method to reverse-engineer several GINs for lung cancer and ovarian cancer. In order to validate the reverse-engineered lung cancer GIN, we have used so-called ChIP-Seq data\footnote{This term is also explained in a later section.} from our collaborators, that give the potential ‘target’ (i.e. downstream) genes of four specific genes, namely ASCL1, PPARG, NKX2-1, and SOX2. Our collaborators could identify 236 potential downstream target genes of ASCL1, 235 potential downstream target genes of PPARG, 724 potential downstream target genes of NKX2-1, and 356 potential downstream target genes of SOX2. In the case of ASCL1 our results are truly spectacular, with the $P$-value (likelihood of getting the match purely through chance) being below machine zero. In the case of PPARG, the $P$-value of obtaining our predictions purely by chance is about 0.01, whereas for NKX2-1 the $P$-value is about 0.02. Since the biology community widely adheres to 0.05 as its significance threshold, we can claim that our predictions are validated in all three cases. For SOX2 the predicted neighborhood was not particularly enriched. However, SOX2 ranks very low in terms of connectivity with the data.
in the reverse-engineered GIN (No. 3,856 out of 19,579 genes), and hence cannot be considered a ‘hub’. So perhaps our results are not unexpected. As a further ‘sanity check’, we tested the neighborhood of ASCL1 in the ovarian cancer GIN; it was not in the least enriched for ChIP-Seq genes identified for lung cancer. This is as it should be, and lends further credence to our reverse-engineered GIN for lung cancer.

2 Literature Review

The problem of inferring GINs from expression data is obviously not new, and several researchers have attempted to study this problem. Most existing methods can be grouped into one of two categories, namely: those based on mutual information, and those based on Bayesian networks. Papers such as [5, 17, 24, 12] are representatives of methods based on mutual information, while [11, 2, 10, 15] are representatives of methods based on Bayesian networks. Both classes of methods impose some biologically unrealistic conditions mainly to facilitate the statistical analysis. Specifically, methods based on Bayesian networks require the graph to be acyclic, while methods based on mutual information will result in graphs that are undirected. Neither assumption is justifiable on biological grounds. The Bayesian paradigm, with its information flow restricted to be in one direction, is useful for hierarchical decomposition of GINs into ‘clusters’ of genes where each cluster of genes controls lower-level clusters. This is a much coarser picture of a GIN than the ones obtained by using mutual information-based methods. For this reason, we do not discuss Bayesian-based methods hereafter, and confine ourselves to discussing methods based on mutual information. See [7] for a thorough treatment of all information-theoretic concepts that are discussed only very briefly here.

2.1 Some Mathematical Preliminaries

Suppose $A$ is a finite set, say $A = \{1, \ldots, n\}$. Suppose $X$ is a random variable assuming values in $A$ with associated probability distribution $\mu$. Thus $\mu_i = \Pr\{X = i\}$. Then the entropy of $X$, or the entropy of the probability distribution $\mu$, is defined by

$$H(X) = H(\mu) = -\sum_{i=1}^{n} \mu_i \log \mu_i.$$ 

Now suppose $A = \{1, \ldots, n\}, B = \{1, \ldots, m\}$ are finite sets, and that $X, Y$ are random variables assuming values in $A$ and $B$ respectively. Let $\theta$ denote the joint distribution of $(X, Y)$. Thus $\theta$ is a probability distribution on the product set $A \times B$. Let $\mu, \nu$ denote the marginals of $\theta$ on $A$ and $B$ respectively. Thus $X$ has the probability distribution $\mu$ and $Y$ has the distribution $\nu$. With this
notation, the mutual information between $X$ and $Y$ is defined as

$$I(X, Y) = H(X) + H(Y) - H(X, Y) = H(\mu) + H(\nu) - H(\theta).$$

An alternate and equivalent expression for the mutual information is

$$I(X, Y) = \sum_{i=1}^{n} \sum_{j=1}^{m} \phi_{ij} \log \frac{\phi_{ij}}{\mu_i \nu_j}.$$

Mutual information is always symmetric and nonnegative; that is, $0 \leq I(X, Y) = I(Y, X) \leq \min\{H(X), H(Y)\}$. Moreover, $I(X, Y) = 0$ if and only if $X$ and $Y$ are independent random variables.

Suppose $X, Y, Z$ are random variables assuming values in finite sets $A, B, C$ respectively. Then $X$ and $Z$ are said to be conditionally independent given $Y$, denoted by $(X \perp Z)\mid Y$, if, for all $i \in A, j \in B, k \in C$, it is true that

$$\Pr\{X = i \& Z = k \mid Y = j\} = \Pr\{X = i \mid Y = j\} \cdot \Pr\{Z = k \mid Y = j\}.$$

It is easy to show that the above relationship also implies that, for all $S \subseteq A, j \in B, U \subseteq C$, it is true that

$$\Pr\{X \in S \& Z \in U \mid Y = j\} = \Pr\{X \in S \mid Y = j\} \cdot \Pr\{Z \in U \mid Y = j\}.$$

However, in general it is not true that

$$\Pr\{X \in S \& Z \in U \mid Y \in T\} = \Pr\{X \in S \mid Y \in T\} \cdot \Pr\{Z \in U \mid Y \in T\}$$

for all $S \subseteq A, T \subseteq B, U \subseteq C$. A very useful inequality, known as the ‘data processing inequality’, states that whenever $(X \perp Z)\mid Y$ the following inequality holds:

$$I(X, Z) \leq \min\{I(X, Y), I(Y, Z)\}. \quad (1)$$

See [7, p. 34].

### 2.2 Review of Papers Using Mutual Information

One of the first papers to use the concept of mutual information to construct GINs is [5]. In that paper, the authors compute the mutual information between every pair of genes, and introduce an undirected edge between nodes $i$ and $j$ if and only if the mutual information $I(X_i, X_j)$ between the corresponding random variables $X_i$ and $X_j$ is positive. In the particular example studied, they had 79 samples, which they binned into ten bins, thereby discretizing each random variable into one of ten values. Then the mutual information between these discretized variables is used as an approximation for the true mutual information. Actually, they select a small threshold $\epsilon$ and introduce an undirected edge between nodes $i$ and $j$ if and only if $I(X_i, X_j) \geq \epsilon$. They refer to the resulting (undirected) graph as an ‘influence network’. Indeed, in their framework, the presence of an (undirected) edge between two nodes $i$ and
j makes no distinction between gene i influencing gene j or vice versa. Also, no distinction is made between direct and indirect influence. As a consequence, the influence networks produced by the method in [5] are overly dense.

In an interesting approach termed as context likelihood of relatedness (CLR) [12], the authors attempt to eliminate false correlation and indirect interactions by so-called adaptive background correction step. In this method, mutual information between genes i and j is examined in the context of the distribution of mutual information in their neighborhood. The authors approximate the distribution of mutual information in the neighborhood of a gene i, that is, the set of numbers \( \{I(X_i, X_j), j = 1, \ldots, n\} \), by a Gaussian distribution, and calculate its Z-score \( Z_i \). Similarly, \( Z_j \) is computed from the distribution of mutual information in the neighborhood of gene j. Subsequently, the likelihood estimate between genes i and j is defined as \( f(Z_i, Z_j) = \sqrt{Z_i^2 + Z_j^2} \), which is used as measure of interaction strength between the pair of genes. Now based on a global threshold, edges with smaller likelihood estimate are dropped from the network. It is to be noted that this approach produces an undirected network because the likelihood estimate is symmetric in i and j. Moreover, if the threshold for the likelihood estimate (which is selected globally) is set at too high a value, the resulting network may fail to be connected.

In one of the more popular methods [17], the authors develop a method referred to as ARACNE to distinguish between direct and indirect interactions. They begin with the influence network of [5], which is an undirected graph, and ‘prune’ it using the data processing inequality (1). Specifically, for each triplet i, j, k, they compute all the three mutual informations \( I(X_i, X_j), I(X_i, X_k) \) and \( I(X_j, X_k) \). Since the exact probability distributions are not known and only samples are available, they use Gaussian kernel approximations for the various joint distributions. Then they identify the smallest amongst the three numbers and discard the corresponding edge. Thus if

\[
I(X_i, X_k) \leq \min\{I(X_i, X_j), I(X_j, X_k)\},
\]

then they discard the edge between nodes i and k. They further assume that the joint distribution of all n random variables has the form

\[
\phi(X_1, \ldots, X_n) = \frac{1}{\text{const.}} \prod_{i,j=1}^{n} \phi_{ij}(X_i, X_j).
\] (2)

In other words, they assume that the joint probability distribution can be factored into a product of individual terms depending on just two random variables at a time. Then they cite [6] to justify their algorithm; specifically, if the joint distribution of all random variables is of the form (2), then their algorithm produces the correct interaction graph. However, the assumption that the joint distribution of all random variables is of the form (2) is essentially unverifiable. Furthermore, the pruning strategy used in this algorithm means that the GIN generated will never contain a complete subgraph of three nodes. In other words, if there is an edge between nodes i and j, and between nodes j and k,
then there cannot be an edge between nodes $i$ and $k$. But biology is full of small local networks that contain three-node complete subgraphs. Thus any claim that the GIN produced by ARACNE is ‘accurate’ is open to question, in the view of the authors.

Note that all of the methods based on mutual information such as CLR and ARACNE are by nature restricted to producing undirected graphs, because mutual information is a symmetric measure of interaction between random variables. It would be highly desirable to develop an algorithm that is able to identify the directionality of interaction between genes. Moreover, it is explicitly stated in [5] (and implicitly assumed in [17]) that mutual information can be used as a measure of the strength of interaction between two random variables. But this statement is only partially true. It is true that if $I(X,Y) < I(X,Z)$, then $Z$ tells us more about $X$ than $Y$ does. So in this sense $X$ depends more on $Z$ than on $Y$. However, if $I(X,Y) < I(W,Z)$, it is not correct to conclude that $X$ depends less on $Y$ than $W$ depends on $Z$. The algorithm presented here addresses both of these limitations.

Proceeding further, in all the mutual information-based approaches, the most computationally intensive step is the computation of the pairwise mutual informations. In [25], the authors take the given sample pairs $\{(x_{il}, x_{jl}) \mid l = 1, \ldots, m\}$ for each pair of indices $i, j$, and then fit them with a two-dimensional Gaussian kernel. Then they apply a copula transformation so that the sample space is the unit square, and the marginal probability distribution of each random variable is the uniform distribution. In [20], the authors propose a window-based approach for computing the pairwise mutual informations. It is claimed in this paper that the proposed method results in roughly an order of magnitude reduction in computing effort, as compared to [17]. Finally, in a very recent paper [3], the authors bin the samples into just three bins irrespective of how many samples there are, and propose a highly efficient parallel architecture for computing the pairwise mutual informations. While the proposed architecture is very innovative, it appears to the present authors that quantizing the expression values into just three bins could result in misleading conclusions, because the gene expression level is essentially a real-valued random variable. In the method proposed here, we also discretize the samples by percentile binning. However, instead of computing the mutual information between random variables, we compute the so-called $\phi$-mixing coefficient between them, as defined in the next section. In this case there is a closed-form formula for the $\phi$-mixing coefficient [1], so that computing it is a triviality; thus all the computational requirements associated with determining the mutual information simply disappear.

3 The New Algorithm

The algorithm proposed here is based on computing the so-called $\phi$-mixing coefficient between two random variables. The $\phi$-mixing coefficient was introduced in [13] as a measure of the asymptotic long-term independence of a stationary
stochastic process, and was used to prove laws of large numbers for non-i.i.d. processes. See [23 (2.5.3)] for the general definition. This definition can be readily adapted to define a quantitative measure of the dependence between two random variables; see [9]. From the standpoint of reverse-engineering GINs from expression data, the most appealing feature of the \( \phi \)-mixing coefficient is its directionality. Unlike mutual information or Pearson correlation, the \( \phi \)-mixing coefficient distinguishes between the dependence of \( X \) on \( Y \) and that of \( Y \) on \( X \). The GINs produced by this algorithm have the following features:

1. The GIN is invariant under any monotone transformation of the data. In other words, if \( f_i : \mathbb{R} \rightarrow \mathbb{R}, i = 1, \ldots, n \) are any monotonic functions, then the GIN produced by applying the algorithm to the original set \( \{x_{ij}, i = 1, \ldots, n, j = 1, \ldots, m\} \) will be exactly the same as the GIN produced by applying the algorithm to the transformed data set \( \{f_i(x_{ij}), i = 1, \ldots, n, j = 1, \ldots, m\} \). This feature is very useful because in carrying out gene expression studies, different ‘platforms’ (i.e. commercial products) use different ways of post-processing the raw measurements to produce the expression values.

2. The GIN has weighted, directional edges.

3. Each edge within the GIN has a weight between 0 and 1.

4. In contrast with using mutual information as a weight, in the present case lower weights denote less strong interactions. Thus if the weight of the edge from \( Y \) to \( X \) is lower than the weight of the edge from \( Z \) to \( W \), then \( X \) depends less on \( Y \) than \( W \) does on \( Z \).

5. The resulting GIN is permitted to contain cycles, and the edges are directional. That is, if \( A \) and \( B \) are two nodes in the GIN, then it is possible to have an edge from \( A \) to \( B \) but not from \( B \) to \( A \), and it is also possible to have edges from \( A \) to \( B \) and from \( B \) to \( A \), while the weights are the two edges could be different.

6. The resulting GIN is strongly connected; that is, there is a directed path between every pair of nodes.

7. The resulting GIN is potentially a minimal network that is consistent with the data. In other words, if some of the edges are removed, the resulting graph would no longer be consistent with the data, unless some other edges are added to it by way of compensation.

In the remainder of this section we present the theory as well as some implementational details of the algorithm.
3.1 Phi-Mixing Coefficient: Definition and Computation

If $X$ and $Y$ are random variables assuming values in possibly distinct finite sets $\mathbb{A} = \{1, \ldots, n\}$ and $\mathbb{B} = \{1, \ldots, m\}$ respectively, the $\phi$-mixing coefficient $\phi(X|Y)$ is defined as

$$
\phi(X|Y) := \max_{S \subseteq \mathbb{A}, Y \subseteq \mathbb{B}} |\Pr\{X \in S|Y \in T\} - \Pr\{X \in S\}|.
$$

Thus $\phi(X|Y)$ is the maximum difference between the conditional and unconditional probabilities of an event involving only $X$, conditioned over an event involving only $Y$. Specifically, the $\phi$-mixing coefficient has the following properties:

1. $\phi(X|Y) \in [0, 1]$.
2. In general, $\phi(X|Y) \neq \phi(Y|X)$. Thus the $\phi$-mixing coefficient gives directional information.
3. $X$ and $Y$ are independent random variables if and only if $\phi(X|Y) = \phi(Y|X) = 0$.
4. The $\phi$-mixing coefficient is invariant under any one-to-one transformation of the data. Thus if $f: \mathbb{A} \to \mathbb{C}, g: \mathbb{B} \to \mathbb{D}$ are one-to-one and onto maps, then

$$
\phi(X|Y) = \phi(f(X)|g(Y)).
$$

It is evident that $\phi(X|Y)$ measures the degree of interdependence between $X$ and $Y$. Thus, unlike with mutual information, if $\phi(X|Y) < \phi(W|Z)$, then it can indeed be said that $X$ depends less on $Y$ than $W$ does on $Z$.

The material presented above is all standard. Now we review two new results from [1] that are crucial for the algorithm being proposed here.

While (3) is suitable for defining the quantity $\phi(X|Y)$, it cannot be directly used to compute it. This is because (3) requires us to take the maximum over all subsets of $\mathbb{A}$ and $\mathbb{B}$, and would thus require $2^{|\mathbb{A}|} |\mathbb{B}|$ computations. However, in the special case where the marginal distribution of $Y$ is the uniform distribution, then it is quite easy to compute the associated coefficient $\phi(X|Y)$. Specifically, let $\Theta \in [0, 1]^{n \times m}$ denote the joint distribution of $X$ and $Y$ written out as a matrix. In other words,

$$
\theta_{ij} = \Pr\{X = i&Y = j\}, \forall i \in \mathbb{A}, j \in \mathbb{B}.
$$

Let $\mu, \nu$ denote the marginal distributions of $X$ and $Y$ respectively; thus

$$
\mu_i = \Pr\{X = i\} = \sum_{j=1}^{m} \theta_{ij}, \nu_j = \Pr\{Y = j\} = \sum_{i=1}^{n} \theta_{ij}.
$$

The assumption that both random variables are finite-valued is made purely for convenience in exposition. In the general case, the sets $S$ and $T$ would have to belong to the $\sigma$-algebras generated by the random variables $X$ and $Y$ respectively, and the maximum would have to be replaced by the supremum.
Define $\Psi \in [0, 1]^{n \times m}$ as the outer product of $\mu$ and $\nu$; thus
$$\psi_{ij} = \mu_i \nu_j, \forall i, j.$$ Then $\Psi$ is a rank one matrix, and is the joint distribution that $X$ and $Y$ would have if they were independent. Define $\Lambda \in [-1, 1]^{n \times m}$ by
$$\lambda_{ij} = \theta_{ij} - \psi_{ij}, \forall i, j.$$ Thus $\Lambda$ would be the zero matrix if $X$ and $Y$ were independent. Define
$$\|\Lambda\|_{i1} := \max_{j=1,\ldots,m} \sum_{i=1}^{n} |\lambda_{ij}|$$ to be the matrix norm of $\Lambda$ induced by the $\ell_1$ vector norm. With these definitions, the following facts are established in [1]:

**Theorem 3.1** With the notation as above, it is the case that
$$0.5 \frac{\|\Lambda\|_{i1}}{\max_j \nu_j} \leq \phi(X|Y) \leq 0.5 \frac{\|\Lambda\|_{i1}}{\min_j \nu_j}.$$ (4)
In particular, if $\nu$ is the uniform distribution so that $\min_j \nu_j = \max_j \nu_j = 1/m$, then
$$\phi(X|Y) = 0.5m \|\Lambda\|_{i1}.$$ (5)

The most important property of the $\phi$-mixing coefficient is given next. Again, the proof can be found in [1].

**Theorem 3.2** Whenever $(X \perp Z)|Y$, the following inequality holds:
$$\phi(X|Z) \leq \min(\phi(X|Y), \phi(Y|Z)).$$ (6)

Note that (6) is entirely analogous in appearance to (1). For this reason, we will refer to (6) as the data processing inequality for the $\phi$-mixing coefficient. The observation that the $\phi$-mixing coefficient satisfies an analog of the DPI is new, and a proof of (6) can be found in [1].

### 3.2 Algorithm for Reverse-Engineering GINs: Theory

In this subsection and the next, we describe our proposed algorithm in detail. In the present subsection we present the theory behind the algorithm, which requires that we know the actual coefficient $\phi(X_i|X_j)$ for each pair of indices $i, j$. In the next subsection we discuss how the algorithm can be implemented in practice, taking account the fact that we can only estimate the coefficient based on a finite number of samples.

So let us begin by assuming (somewhat unrealistically) that exact values are available for all $n(n-1)$ coefficients $\phi(X_i|X_j)$ for each pair of indices $i, j$ for $i \neq j$. Then we proceed as follows: Start with a complete graph of $n$ nodes, where
there is a directed edge between every pair of distinct nodes \((n(n-1) \text{ edges})\). For each triplet \(i, j, k\), check whether the DPI-like inequality
\[
\phi(X_i|X_k) \leq \min\{\phi(X_i|X_j), \phi(X_j|X_k)\}
\] (7)
holds. If so, discard the edge from node \(k\) to node \(i\), but retain a ‘phantom’ edge for future comparison purposes.

This step is referred to as ‘pruning’. Note that the pruning operation can at best replace a direct path of length one (i.e. an edge) by an indirect path of length two. Hence the graph that results from the pruning operation is still strongly connected. Also, since any discarded edges are still retained for the purposes of future comparisons, it is clear that the order in which the triplets are processed does not affect the final answer. Note that the complexity of this operation is cubic in \(n\).

At this stage, one can ask whether the graph resulting from the pruning operation has any significance. It is now shown, by invoking the Occam’s razor principle (giving the simplest possible explanation), that the graph resulting from pruning is a minimal graph consistent with the data set. For this purpose, we define a partial ordering on the set of directed graphs with \(n\) nodes whereby \(G_1 \leq G_2\) if \(G_1\) is a subgraph of \(G_2\), ignoring weights of the edges. For a given triplet \(i, j, k\), it is obvious that \((X_i \perp X_k)|X_j\) if and only if every directed path from node \(i\) to node \(k\) passes through node \(j\), and also every directed path from node \(k\) to node \(i\) passes through node \(j\). Now, it follows from the DPI that if \((X_i \perp X_k)|X_j\), then (7) holds. Taking the contrapositive shows that if (7) is false, then \((X_i \not\perp X_k)|X_j\). Consequently, if (7) is false, then there must exist a path from node \(i\) to node \(k\) that does not pass through node \(j\). Given the sequential nature of the pruning algorithm, when (7) is checked for a specific triplet \((i, j, k)\), there already exist edges from node \(i\) to node \(j\) and from node \(j\) to node \(k\); that is, there exists a path of length two from node \(i\) to node \(k\). Now, if (7) is false, then there must exist another path from node \(i\) to node \(k\) that does not pass through node \(j\). It is of course possible that this path consists of many edges. However, by the Occam’s razor principle, the simplest explanation would be that there is a shorter path of length one, i.e. a directed edge from node \(k\) to node \(i\).

What has been shown is that, under the Occam’s razor principle, the graph that results from pruning is minimal in the following sense. First, it is consistent with the \(\phi\)-mixing coefficients, and second, any other graph that is ‘less than’ this graph in the partial ordering defined above would not be consistent with the \(\phi\)-mixing coefficients. Thus, if any edges are removed from the graph that results from applying the pruning step, then some other edges would have to be added in order for the graph to be consistent with the \(\phi\)-mixing coefficients. Note that we are obliged to say a and not the minimal graph, because there might not be a unique minimal graph. Nevertheless, it is obvious that the application of the algorithm will result in a unique graph, irrespective of the order in which all the triplets \((i, j, k)\) are examined.
3.3 Algorithm for Reverse-Engineering GINs: Implementation

Now that the basic theory is in place, it is possible to present an algorithm for reverse-engineering a GIN from experimental expression data. The main issue here is that we do not know the ‘true’ coefficient $\phi(X_i|X_j)$ exactly. Even if we were to discretize the random variables by binning and then use (3), the resulting quantity would still only be an approximation of $\phi(X_i|X_j)$ and not the exact value. Moreover, a direct application of (3) would be too expensive computationally. These are some of the considerations that enter into the implementation described below. Recall that there are $n$ genes and $m$ samples of each.

**Binning the Data:** Choose an integer $k$ such that $k \leq \lfloor (m/3)^{1/2} \rfloor$. For each index $i$, divide the total range of $X_i$ into $k$ bins that correspond to ‘percentiles’. Note that percentile binning is also referred to as ‘data-dependent partitioning’ in the statistics literature. For each pair of indices $i,j$, and each sample label $l$, assign the sample pair $(x_{il}, x_{jl})$ to its associated bin. The discretization ensures that each random variable $X_i$ assumes one of just $k$ values (corresponding to the bins). The choice of $k$ ensures that on average there will be at least three entries in each of the $k^2$ bins of the joint random variable $(X_i, X_j)$ for each pair $(i,j)$. The choice of percentile discretization (as opposed to, for example, uniformly gridding the range), ensures that the marginal distribution of each $X_i$ is nearly equal to the uniform distribution on $k$ labels, and allows us to use the estimates (4). If $m$ is an exact multiple of $k$ then each marginal distribution would indeed be the uniform distribution, but in general $m$ might not be an exact multiple of $k$. Percentile binning also ensures that the joint distribution of the discretized pairs $(X_i, X_j)$ remains invariant under any monotonic transformation of the data. It is important to note here that the invariance property holds even if different monotone transformations are applied to different expression variables.

**Estimating the $\phi$-mixing coefficient:** After binning, for each pair of indices $i,j$, we determine the associated joint distribution of the discretized random variables, which will be a $k \times k$ matrix. For each pair of indices $i,j$, we use (4) to compute an interval $[\phi_l(X_i|X_j), \phi_u(X_i|X_j)]$ that contains the true value $\phi(X_i|X_j)$. Define $\phi_a(X_i|X_j) = (\phi_l(X_i|X_j) + \phi_u(X_i|X_j))/2$ to be the midpoint of this bounding interval. Note that we are being a bit imprecise since $X_i$ now represents the discretized and not the original (continuous) expression value. However, in the interests of notational simplicity, we ignore this distinction. The complexity of this operation is quadratic in $n$, the total number of genes.

**Pruning:** As before, start with a complete graph on $n$ nodes, and then apply the data processing inequality to do the pruning, for each triplet $(i,j,k)$. Since we have only empirically determined values of the mixing coefficient, there are several possible ways of interpreting the data processing inequality. At this stage we examined three different ways of implementing the pruning operation.
1. Eliminate the edge from node \( j \) to node \( i \) if
\[
\phi_u(X_i | X_k) \leq \min\{\phi_l(X_i | X_j), \phi_l(X_j | X_k)\}.
\] (8)

2. Eliminate the edge from node \( j \) to node \( i \) if
\[
\phi_a(X_i | X_k) \leq \min\{\phi_a(X_i | X_j), \phi_a(X_j | X_k)\}.
\] (9)

3. Eliminate the edge from node \( j \) to node \( i \) if
\[
\phi_a(X_i | X_k) \leq \min\{\phi_a(X_i | X_j), \phi_a(X_j | X_k)\}.
\] (10)

Since it is always the case that
\[
\phi_l(X_i | X_j) \leq \phi_a(X_i | X_j) \leq \phi_u(X_i | X_j),
\]
it is easy to see that any edge that gets pruned out under Rule 1 also gets pruned out under Rule 2, but Rule 2 could also prune out other edges that survive Rule 1. Similar remarks apply to Rule 2 vis-à-vis Rule 3. Thus if let \( G_1, G_2, G_3 \) denote the graphs produced by applying Rule 1, Rule 2, and Rule 3 to the same data set, then it is easy to see that \( G_3 \) is a subgraph of \( G_2 \), which is in turn a subgraph of \( G_1 \). Based on several numerical experiments as discussed in the Results section, we finally opted to use Rule 2, but with ‘thresholding’ as explained next.

**Thresholding:** We have constructed several genome-wide networks from expression data, as detailed in the next section. Our numerical experiments have shown that after the pruning step described above, the GINs that result are characterized by the property that the mean value of all the edge weights is noticeably higher than the median. This means that there are relatively far more edges with low weights than edges with high weights, but the high-weight edges have significantly higher weights. One of the main reasons for reverse-engineering GINs is to identify ‘hubs’, that is, genes that are connected to many other genes. Again, the GINs that result from the pruning step do not show sufficient variation between the largest node-degree and the smallest node-degree. In ‘validating’ the reverse-engineered GIN, it is highly desirable to eliminate all of these low-weight edges, while still ensuring that the graph remains strongly connected. Several numerical experiments have suggested the following strategy: After the pruning step, compute the mean \( \mu \) and standard deviation \( \sigma \) of the weights of all edges. Then eliminate all edges whose weights are below \( \mu \). If the thresholded graph is strongly connected, then keep it; otherwise lower the threshold to \( \mu - \sigma \), or use no threshold at all. Thus far, in our various experiments, in about half of the cases the threshold of \( \mu \) has resulted in a strongly connected network, while in one case even the higher threshold of \( \mu + \sigma \) has resulted in a strongly connected network. Interestingly, the highest degree nodes in the pruned network still remain the highest degree nodes in the pruned and thresholded network. The numerical examples in the section on validation make this point clear.
Table 1: Networks Pruned Using Rule 2

| No. | No. of Genes | No. of Samples | No. of Edges & SCCs Under Various Thresholds |
|-----|--------------|----------------|---------------------------------------------|
|     | (n)          | (m)            | 0               | µ               | SCCs |
| 1.  | 19,579       | 148            | 3,853,936       | 1,791,624       | 1    |
| 2.  | 19,579       | 108            | 4,474,834       | 2,283,114\(^\dagger\) | 2    |
| 3.  | 19,579       | 29             | 2,548,501       | 940,785\(^\dagger\) | 6    |
| 4.  | 17,814       | 585            | 545,988         | 192,966\(^\dagger\) | 1,026 |
| 5.  | 17,814       | 57             | 2,882,376       | 1,222,985       | 1    |
| 6.  | 17,814       | 53             | 3,667,170       | 1,132,059       | 1    |
| 7.  | 15,993       | 119            | 2,688,969       | 1,400,242\(^\dagger\) | 2    |

4 Case Studies

Using the algorithm presented here, we have reverse-engineered several GINs for both lung cancer and ovarian cancer. The raw statistics of the various networks obtained using Rule 2 for pruning are given in Table 1. Statistics for the number of edges that remain after thresholding the resulting network with the threshold \(\mu\), as well as the number of strongly connected components (SCCs) are also given. Note that the network with no threshold is always strongly connected so the number of SCCs is always one in that case, and is therefore not displayed. The superscript \(^\dagger\) indicates that the resulting thresholded network is not strongly connected, i.e. that the threshold is too high.

It can be seen that, except in one case (Network No. 4), the networks resulting from pruning using Rule 2 and then thresholding at the level of \(\mu\) are either strongly connected, or else have a very small number of SCCs, meaning that ‘for all practical purposes’ they are strongly connected. If we use Rule 1 for pruning, the resulting networks contain far too many edges to be of any use, whereas if we use Rule 3 for pruning, the networks tend to become disconnected into a large number of SCCs at a threshold of \(\mu\). For this reason, we have chosen to use Rule 2 for pruning and a threshold of \(\mu\) for all future analyses.

Now for some details about these networks. Networks 1, 2 and 3 are based on gene expression data from lung cancer cell lines from the laboratory of Prof. John Minna and were provided to us by his student, Alex Augustyn. There were 148 cell lines in all, consisting of 108 non-small cell lung cancer (NSCLC), 11 neuro-endocrine non-small cell lung cancer (NE-NSCLC), and 29 small-cell lung cancer (SCLC). Network 1 is based on combining the data from all these lines, whence the number of samples is 148. Network 2 is based on the NSCLC samples alone, while Network 3 is based on the SCLC samples alone. Networks 4, 5 and 6 are based on publicly available expression data for ovarian tumor tissues that have been resected (surgically removed); the data is available from the web site of The Cancer Genome Atlas [22]. Network 4 is based on all 585
samples available on the date we downloaded the data. Network 5 is based on a subset of 57 samples within this set of 585, representing patients who responded particularly well to platinum chemotherapy, while Network 6 is based on another subset of 53 samples representing patients who did not respond well to platinum chemotherapy. This identification of the responders and non-responders was carried out by Drs. Keith Baggerly and Anna Unruh. Finally, Network 7 is also a lung cancer GIN. The samples are the NE-NSCLC and NSCLC data sets for a total of 119 samples, while the genes are a subset of 15,993 genes from the original 19,579. The laboratory of Prof. Michael White, specifically Dr. Hyunseok Kim, performed very elaborate experiments on a dozen lung cancer cell lines by systematically knocking one gene at a time, using siRNA technology, on 24,987 genes. This subset of 15,993 genes represent the intersection of the 24,987 and the 19,579, that is, genes for which expression levels in the 119 cancer cell lines as well as the effects of siRNA experiments are both available.

From all these networks, a few general features of our algorithm emerge.

- Recall that the algorithm begins with a complete directed graph on \( n \) nodes; therefore the initial number of edges is \( n(n - 1) \approx n^2 \). Applying the pruning step retains the strong connectivity property, as pointed out earlier. In all of the cases studied here, the pruning step eliminates 99% or more of these edges. Therefore the algorithm is quite efficient in terms of finding a very small GIN consistent with the data.

- It can be seen that in all except one case, the network thresholded with the mean edge weight \( \mu \) has fewer than half of the edges in the pruned network. This implies that the mean of the edge weights of the post-pruning network is higher than the median, or equivalently, the pruned network (before thresholding) has a large number of low-weight edges and relatively fewer high-weight edges.

We conclude this section by studying in greater detail the ‘power law’ nature of the degree distributions GIN No. 1, for lung cancer, with the threshold set equal to \( \mu \). Note that the validation step discussed in the next section is based on this network.

It is widely believed by biologists that real GINs consist of a few master regulators that control many hundreds of other genes. It is also proposed by some authors that biological GINs show a power law behavior. Specifically, let \( d \) denote an integer corresponding to the degree of a node. (For each node, the phrases in-degree, out-degree and total degree are self-explanatory.) Let \( n(d) \) denote the number of nodes in the GIN that have degree \( d \). Then the belief is that \( n(d) \) asymptotically looks like \( d^{-\alpha} \) for some index \( \alpha \). We wished to examine whether this is indeed true for Network No. 1.

It turns out that the total degree of various nodes does not show much variation; the maximum is 1,735 for ATCAY and the minimum degree is 64. Similarly, the in-degree also does not show much variation, ranging from 411 to 34. In contrast, the out-degree, that is to say, the number of downstream neighbors of a gene, varies from a high of 1,626 to a low of 1, which is the
theoretical minimum. (Note that if any node had in- or out-degree of zero, then the network would not be strongly connected.) For this reason, we studied the distribution of \( n(d) \) as a function of the outdegree \( d \). To avoid the graph becoming too jerky, we ‘binned’ the degree \( d \) into bins of width five. That is to say, we computed the number of nodes with degrees between 1 and 5, and then between 6 and 10, and so on. The graph for the entire range of degrees is not particularly informative. However, we observed that the vast majority of nodes have degrees between 100 and 300. A plot of the degrees shows the power law behavior with exponent of \(-6.88\).

5 Validation of the GINs

As mentioned in the introduction, it is virtually impossible to validate an entire GIN since there is no known and confirmed ‘absolute truth’ against which
predictions can be confirmed. Existing examples of GINs such as [14, 19, 4, 21] are often obtained by combining small interaction networks from a variety of sources. The difficulty with this approach is that the context in which these small interaction networks are determined need not be the same across all of them. Hence combining these small individual networks into one large patchwork network cannot be justified biologically, and the resulting network cannot be accepted as reflecting ‘reality’. It must be emphasized that the entire raison d’être of our algorithm is to embrace the entire genome (or as much of it as is covered by the data) to produce a context-specific, genome-wide network. Hence it would be inappropriate to compare the reverse-engineered networks with the networks available in public domain databases. In any case, even the largest networks obtained in this (somewhat dubious) manner still cover only about half of the 22,000 or so human genes; see [21]. In contrast, the networks that we have reverse-engineered routinely combine all genes studied in the experiment, of the order of 20,000, as seen from the previous section.

This therefore raises the question of just how such reverse-engineered networks are to be validated. After considerable thought, we chose to use evidence from so-called ChIP-seq tests for a few transcription factors. A transcription factor is a special kind of gene that is involved in regulating the transcription of other genes. ChIP-seq stands for ‘chromatin immuno-precipitation sequencing’. A good introduction to ChIP-seq for non-biologists can be found at [16]. The basic idea is that a transcription factor is immuno-precipitated with the entire genome. As a result, several DNA fragments bind to the transcription factor. After these fragments are isolated through other experimental techniques, the fragments are then sequenced. In principle each DNA fragment represents one or more genes that are regulated by the transcription factor. However, the raw experimental technique is rife with false positives and produces literally thousands of genes as being potentially regulated by the transcription factor under study. Further analysis is needed to weed out these false positives, and thus produce a realistic set of potential downstream target genes. For one of the transcription factors studied here, namely ASCL1, our collaborators (Prof. Jane Johnson and Mr. Mark Borromeo) used an algorithm called GREAT (Genomic Regions Enhancement of Annotations Tool) [18] to eliminate most of these false positives, and produce a set of potential target genes. For the other two transcription factors studied here, namely PPARG and NKX2-1, the laboratory of Prof. Ralf Kittler, specifically Dr. Rahul Kollipara, produced a set of potential target genes for each transcription factor using a different peak-calling routine.

As mentioned above, applying GREAT or another similar algorithm to the raw ChIP-seq data results in a set of potential downstream target genes of the transcription factor under study. This list of genes can then be compared with the downstream neighbors of the same transcription factor in the reverse-engineered GIN. It is also reasonable to consider all first-order neighbors of the transcription factor, both up-stream as well as down-stream, and to compare

---

6 Recall that the ‘central dogma’ of biology, as enunciated by Francis Crick [8] states that DNA is converted to RNA (transcription) which is then converted to protein(s) (translation).
Table 2: Number of Potential Target Genes for Various Transcription Factors

| Item                  | ASCL1 | PPARG | NKX2-1 |
|-----------------------|-------|-------|--------|
| Total no. of genes    | 19,579| 19,579| 19,579 |
| Total no. of ChIP genes | 236   | 235   | 724    |
| Prob. of being a ChIP gene | 0.0121| 0.0120| 0.0370 |

In the interests of brevity, these are referred to, somewhat inaccurately, as ‘ChIP-seq genes’.

As mentioned above, we obtained ChIP-seq data for three transcription factors in lung cancer tissues, namely: ASCL1, PPARG, and NKX2-1. In biology, a likelihood (P-value) less than 0.05 is considered significant, and in all cases the computed likelihood is smaller than this threshold. The first table below shows the relative likelihood that a randomly chosen gene is a ChIP gene for each of these transcription factors.

The next table shows the enrichment of ChIP genes amongst the first-order downstream neighbors, and amongst all neighbors, for these three transcription factors. In this table, an entry of zero for the likelihood means that the number is smaller than machine zero.
Table 3: Enrichment of Neighbors for ChIP Genes

| Gene Name | Up-Neigh. | ChIP Genes | $L_d$ | Tot. Neigh. | ChIP Genes | $L_t$ |
|-----------|-----------|------------|-------|-------------|------------|-------|
| ASCL1     | 690       | 84         | 0     | 766         | 87         | 0     |
| PPARG     | 84        | 3          | 0.0035| 208         | 5          | 0.0135|
| NKX2-1    | 114       | 6          | 0.0614| 244         | 14         | 0.0203|

Finally, ‘truth in advertising’ compels us to disclose that we had also obtained ChIP-seq data for a fourth transcription factor, namely SOX2, which had 356 potential target genes. Using the simple binomial model above, the downstream neighbor genes were ‘enriched’ with a likelihood value of 0.1513, while the total neighbor genes were enriched with a likelihood value of 0.1829. Thus the fraction of ChIP genes was higher than pure chance would indicate, but the likelihood was not sufficiently small. Our explanation for this phenomenon is as follows: In the reverse-engineered lung cancer GIN, ASCL1 is truly a ‘hub’, ranking at no. 9 out of 19,579 genes in terms of its total degree, and at no. 10 in terms of its out-degree. Hence predictions about its neighborhood using the GIN are perhaps fairly reliable, as evidenced by the fact that the likelihood of pure chance for ChIP genes is below machine zero. NKX2-1 ranks at no. 526 in terms of total connectivity, or within the top 3% of genes, and can thus be considered a hub. For these hub genes, it is reasonable to expect that our reverse-engineered network will be a good predictor of its neighborhood. In contrast, PPARG ranks at no. 4157 and SOX2 ranks at no. 3856, both far too low down the list to be considered hubs.

6 Next Steps and Concluding Remarks

Acknowledgements

The authors thank Prof. John Minna and Mr. Alex Augustyn for the expression data for various lung cancer cell lines; Prof. Jane Johnson and Mr. Mark Borromeo for the ChIP-seq data on ASCL1; Prof. Ralf Kittler and Dr. Rahul Kollipara for the ChIP-Seq data on PPARG and NKX2-1; and Drs. Keith Baggerly and Anna Unruh for identifying the responders and non-responders to platinum chemotherapy in ovarian cancer patients.

References

[1] Ahsen, M. E., and Vidyasagar, M. Computing various mixing coefficients between random variables. *arxiv* (2012), 1208:1720.
[2] Barash, Y., and Friedman, N. Context-specific bayesian clustering for gene expression data. J. Comput. Biol. 9(2) (2002), 169–191.

[3] Belcastro, V., et al. Reverse engineering and analysis of genome-wide gene regulatory networks from gene expression profiles using high-performance computing. IEEE/ACM Transactions on Computational Biology and Bioinformatics 9(3) (May/June 2012), 668–674.

[4] Biogrid. http://thebiogrid.org/.

[5] Butte, A. J., and Kohane, I. S. Mutual information relevance networks: Functional genomic clustering using pairwise entropy measures. Pacific Symposium on Biocomputing (2000), 418–29.

[6] Chow, C. K., and Liu, C. N. Approximating discrete probability distributions with dependence trees. IEEE Trans. Info. Thy. 14(3) (May 1968), 462–467.

[7] Cover, T. M., and Thomas, J. A. Elements of Information Theory (Second Edition). Wiley Interscience, New York, 2006.

[8] Crick, F. H. C. Central dogma of molecular biology. Nature 227 (08 August 1970), 561–563.

[9] Doukhan, P. Mixing: Properties and Examples. Springer-Verlag, Heidelberg, 1994.

[10] Friedman, N. Inferring cellular networks using probabilistic graphical models. Science 303 (February 2004), 799–805.

[11] Friedman, N., et al. Using bayesian networks to analyze expression data. J. Comput. Biol. 7(3-4) (June-August 2000), 601–620.

[12] Gardner, T. S., et al. Inferring genetic networks and identifying compound mode of action via expression profiling. Science 301, 5629 (2003).

[13] Ibragimov, I. A. Some limit theorems for stationary processes. Theory of Probability and its Applications 7 (1962), 349–382.

[14] Intact. http://www.ebi.ac.uk/intact/.

[15] Koller, D., and Friedman, N. Probabilistic Graphical Models: Principles and Techniques. MIT Press, Cambridge, MA, 2009.

[16] Liu, E. T., Pott, S., and Huss, M. Q&a: Chip-seq technologies and the study of gene regulation. BMC Biology 8:56 (2010), http://www.biomedcentral.com/1741–7007/8/56.

[17] Margolin, A. A., et al. Aracne: An algorithm for the reconstruction of gene regulatory networks in a cellular context. BMC Bioinformatics 7(Supplement 1):S7 (20 March 2008).
[18] McLean, C. Y., et al. Great improves functional interpretation of cis-regulatory regions. *Nature Biotechnology* 28(5) (2010), 495–501.

[19] Mint. http://mint.bio.uniroma2.it/mint/welcome.do.

[20] Qiu, P., Gentles, A. J., and Plevritis, S. K. Reducing the computational complexity of information theoretic approaches for reconstructing gene regulatory networks. *J. Comput. Biol. 17*(2) (February 2010), 169–176.

[21] String. http://thebiogrid.org/.

[22] TCGA. http://cancergenome.nih.gov.

[23] Vidyasagar, M. *Learning and Generalization: With Applications to Neural Networks and Control Systems*. Springer-Verlag, London, 2003.

[24] Wang, K., et al. Genome-wide identification of post-translational modulators of transcription factor activity in human b cells. *Nature Biotechnology* 27(9) (September 2009), 829–839.

[25] Wentao Zhao, E. S., and Dougherty, E. R. Inferring connectivity of genetic regulatory networks using information theoretic criteria. *IEEE/ACM Trans. Comput. Biol. and Bioinf. 5*(2) (April-June 2008), 262–274.