Inference of gene regulatory network from large scale expression data: getting started

Olivier Bernardi

October 8, 2002
Chapter 1

Introduction

As biology enters its post genomic era, and new high throughput technology provides us with genomic scale expression data, one of the most challenging tasks today is to unravel the regulatory process underlying gene expression. Following the central dogma of modern biology, all active processes of living organisms are determined, on the molecular level, by gene expression patterns. Gene expression is regulated by proteins and results in specific protein production. We can therefore consider the set of proteins along with regulatory relations as a large network, the state of which represents the state of a cell. Such a representation is a good way of thinking about regulatory processes in order to devise predictive models. At the very end, understanding such a network should enable us to reveal regulation events involved in differentiation, and disease progression and pinpoint potential targets for drug discovery and diagnostics ([3]). Some advances have been made by focusing on a limited number of genes ([4, 5, 6]). However, as a new class of technology emerges, allowing a global monitoring of gene expression, there is an increasing need for methods that can handle large-scale data in order to unravel the underlying regulatory network ([1]). Though one cannot hope to achieve the same biochemical accuracy with these methods as with gene specific studies, it is the authors' point of view that they are a necessary step toward a global understanding of regulary processes.

This work aims at guiding new researchers to get familiar with the issue and its formalism and to provide a large, though not extensive, review of the algorithms proposed to extract the regulatory network from expression data. We hope to adress the main problems of the modeling task and to give helpful pointers in order to overcome them.

The field is very active and a number of network models have been proposed in the litterature. Among them, the Boolean network model introduced by ([14]) offers the simplest conceptual framework. In this model, each gene is considered either ON or OFF at any time, depending on wether its expression level is above or below a fixed threshold, and a gene regulation rule is given as a boolean function. For example, consider the regulation of genes A, B and C, where A and B both need to be expressed to induce expression of gene C that, in turn, represses the activity of gene A. You can model this stucuture by the network illustrated in figure 1.1

\[\text{Figure 1.1: Boolean abstraction of regulation of genes A, B and C}\]
Despite its apparent simplicity, this mathematical abstraction presents a rich structure for understanding the basic features of a dynamic network ([15]). Furthermore, the abstraction’s transparency allow a good understanding of the question one has to answer when constructing a model or an algorithm. Already our example highlights the concepts of input (e.g. $A$ and $B$ are inputs of $C$) and rules (e.g. NOT, AND), as well as dynamic properties, such as steady state (e.g. $(A,B,C) = (1,0,0)$ is a steady state), periodic behaviour (e.g $(1,1,0) \rightarrow (1,1,1) \rightarrow (0,1,1) \rightarrow (0,1,0) \rightarrow (1,1,0) \rightarrow \ldots$), or even transient state (e.g. $(0,0,1) \rightarrow (0,0,0) \rightarrow (0,0,0) \rightarrow \ldots$).

We will review a number of modeling methodologies, including boolean, continuous, qualitative and bayesian models, and their corresponding algorithms. Also, we will focus on models allowing one to handle large-scale networks. For this reason, models integrating high molecular details will not be described. Furthermore, we will not review clustering algorithms, but instead discuss how they can be used as a first preprocessing of data. A good review of clustering strategies employed in this field is given in [16]. We will also reason on the basis of eukaryote regulation. This should not be too limiting since the mechanisms involved are usually common to both types of organisms, though some more tangled versions are available in eukaryotes. However, some modeling choices might be dependent on the type of organism one has to consider.

The paper is organized into five sections. In the first section, we summarize basic molecular biology in eukaryotes and report known regulatory processes in order to introduce some ideas of the set of rules we should incorporate in our network model. In the second section, we discuss the required inputs and outputs and the way to obtain them. In the third section, we discuss the strengths and weaknesses of usual abstractions. In the fourth section, we review the main algorithms available for the reverse engineering of regulatory networks. We conclude in the last section by giving some perspectives for future work.
Chapter 2

Biological tools of gene regulation in Eukaryotes

2.1 From DNA to proteins

Proteins are the macromolecules on which every function in the living cell depends. Proteins are synthesized from DNA in a two step process:

![Gene to Protein Diagram](image)

Figure 2.1: From the gene to the protein

The mRNA is a template of the DNA formed during the process of transcription. In turn, mRNA serves as a blueprint for the construction of the protein during the process of translation.

2.1.1 Transcription process

Each protein is encoded within a particular subsequence of DNA called gene. Protein-encoding genes contain coding sequences (exons) that will effectively be mapped into proteins, but also non-coding sequences that are useful for the process of transcription. More precisely we shall find:

![Gene Structure Diagram](image)

Figure 2.2: Some elements of a gene

- **exons** whose sequence encodes the protein
- **introns** that will be removed from the mRNA before it is translated
- the **start site** where the transcription of the gene into mRNA begins
- a **basal promoter**
Transcription will occur in the nucleus according to the following scheme:

- A protein complex binds to the *basal promoter* (this step will be widely used for regulation).
- An enzyme (RNA polymerase) binds to the protein complex.
- Working together, they open the DNA double helix.
- RNA polymerase moves along the DNA. As it does so, it assembles a strand of mRNA mapping the DNA.
- mRNA is then processed while still within the nucleus. In particular, the parts corresponding to introns are then removed from the mRNA strand (*splicing)*.

### 2.1.2 Translation process

The mRNA, once synthesized, is released outside the nucleus. It is then binded by ribosomes (*initiation*). Each ribosome, moving along the mRNA strand, translates it into a polypeptide sequence (*elongation*). Finally the mRNA is released from ribosome (*termination*).

This translation process is repeated until the mRNA strand is degraded (Typically 10 proteins will be produced from one RNA strand).

Some proteins may require geometrical reshaping in order to accomplish their function. This requires further processing. (*folding*)

### 2.2 Main mechanisms of regulation

#### 2.2.1 Regulating transcription rate

Altering the rate of gene transcription is the most important and widely-used strategy for regulation. Transcription is regulated by the binding of proteins (which are called *transcription factors*) to the DNA strand. This binding is specific, that is to say, only a few transcription factors will be able to bind to a given binding site.

We shall look at this more closely.

As previously seen, the gene contains *exons*, *introns*, a *start site*, and a *basal promoter*. But one will also find several transcription control elements called *cis-regulatory elements*.

![Figure 2.3: Mechanism of transcript regulation](image)

For the transcription to begin, the *start site* and *basal promoter* must be bound by a complex of proteins. The complex controls the gene activation. However the *start site* and *basal promoter* sequences are very similar from one gene to another, so that this mechanism cannot regulate gene expression specifically.

In contrast, the cis-regulatory system has a more elaborate structure with associated transcription factors that differ from gene to gene. The binding of the cis-elements by transcription factors control the transcription rate. This mechanism thus constitutes a gene-specific regulation system.
Here we introduce some remarks that could be helpful for further analysis.

1. The \textit{cis-regulatory} system contains a specific arrangement of non-specific elements. The transcription will thus need the presence of a particular combination of transcription factors.

2. As cis-regulatory elements and transcription factors are not specific to the gene they control, most of them have been determined by genetic analysis. Hence, the proteins we will have to consider as the input of our network represent a small part of the whole proteome.

3. Most \textit{cis-regulatory elements} are enhancers, that is, their binding by proteins favour transcription, but some have been found to inhibit it.

4. It has been found that several different combinations of \textit{cis-elements} (and thus the corresponding transcription factors) can be sufficient for transcription ([4]).

5. Proteins can bind to each other. The complex will have different binding properties, so that one protein can prevent the binding of another protein on \textit{cis-elements}. Conversely, one \textit{cis-element} binding might require the formation of a complex.

When thinking of the kind of rules needed in our network, remark 3 introduces the need of \textit{NOT}-like rules, remark 4 the need of \textit{OR}-like rules, and remark 5 the need of \textit{AND}-like rules.

\subsection*{2.2.2 Other regulation strategy}

As stated above, transcription regulation is the main tool for regulation and later in our discussion we will reason mainly on this part. However, it is important to notice that eukaryotes supplement transcriptional regulation with several other methods including:

- altering the rate at which RNA transcripts are processed while still within the nucleus (e.g. altering splicing)
- altering the stability of mRNA molecules; that is, the rate at which they are degraded
- altering the efficiency at which the ribosomes translate the mRNA into a polypeptide (e.g. by controlling the number of ribosomes)
- altering protein folding

We should also keep in mind that many external factors, like temperature, PH, light or nutrient abundancies, might have some effect on the chemical reactions and thus influence the regulatory process.

\subsection*{2.3 An example of identified sub-network}

Davidson \textit{et al} ([4]) made a precise analysis of the cis-regulatory system of gene Endo16 in sea urchins. This case study, though genome oriented, might offer some insight of the rules likely to be present in a regulatory network. The model was built and tested iteratively until a complete concordance between experimental and predicted results was achieved. The experiments done were time-course observation after site mutation. A mutation corresponds to setting the activity of the corresponding transcription factor to its null limit. Hence, some inputs can be seen as boolean abstraction of transcription factor abundance, that is, they are considered either \textit{ON} or \textit{OFF}.

The network derived from that study takes cis-regulatory elements \textit{CG1}, \textit{CG2}, \textit{CG3}, \textit{CG4}, \textit{P}, \textit{Z}, \textit{CD}, \textit{E}, \textit{F} as boolean inputs and cis-regulatory element activation \textit{Ot} as continuous inputs. The output \textit{R} should be thought of as a multiplicative factor by which the transcriptional activity of the basal promoter apparatus is multiplied as a result of regulation.

The regulation activity can be summarized as \( R = f(CG_1, CG_2, CG_3, CG_4, P, Z, CD, E, F, Ot, B, G) \), where \( f \) is a function we can represent as the decision tree of figure 2.4 (B).
We can deduce from this result the boolean abstraction of regulation taking transcription factor as input (we will identify the name of the cis-element and its transcription factor for the sake of clarity), $B + G$ will become $B \lor G$, and $CG_1$, $CG_2$, $CG_3$, $CG_4$ will collapse into a single variable $CG$ since they are binded by the same transcription factor. If we do so, we obtain the relation represented in figure 2.5.

The network is highly complex. In practice, most genes should be regulated with a simpler process implying no more than 4 transcription factors. Nevertheless, we should keep in mind that some regulatory processes might imply up to 10 transcription factors.
Chapter 3

Determining the inputs and outputs of the regulatory network

3.1 Inputs and outputs needed

We have to identify the relevant inputs needed to understand the regulatory process. We are assuming that the main regulation occurs during the transcription process, and will depend on protein bindings on the cis-regulatory elements. Thus, we take as a fundamental hypothesis that gene regulation can be usefully studied with the concentration of transcription factors taken as fundamental inputs and the transcription rate, or equivalently mRNA levels, taken as output. However, as noted before, many other factors, that we shall refer to as \textit{environmental variables}, can influence the gene-expression process. Temperature, PH, light or nutrient abundance are clear examples of parameters that can affect the chemical process. One could thus consider them as an input of the regulatory process.

Simply put, the regulatory process we consider is represented in figure 3.1, where protein levels and environmental variables are our inputs, and mRNA levels are our outputs.

![Figure 3.1: Simplified dynamic system of transcript regulation](image)

Once we have determined the set of inputs we choose to take into account, another problem is to collect the corresponding data. The prediction of protein level from microarray data may not be straightforward. In fact the correlation between mRNA and protein level is still under study ([3]), and we shall discuss it in more detail later. Some other possible inputs, including nutrient abundance, might be difficult or even impossible to measure. However we might consider this information as redundant with respect to protein levels. Indeed, dealing with Eukaryotes, we can expect a multiple-step pathway in the regulation of gene expression in response to nutrient presence. Thus, in most cases, the information about the abundance of nutrients should already be contained in protein level records.
3.2 From microarray data to RNA and protein level

When trying to obtain protein levels (our inputs), one would naturally think to make proteomic measures. The most efficient implementation of proteomic analysis is 2D-PAGE. Basically, this technique consists of using a 2 dimensional gradient in order to separate the proteins for further identification and quantification. However, this technology remains very expensive, and especially when trying to work on a genomic scale, and shows poor resolution for low expression level ([2]). Therefore, we shall deduce protein abundance from mRNA levels, a much more easily measured quantity.

Several technologies are now generating experimental records of mRNA levels. The main available techniques are cDNA microarrays, Oligonucleotide chips, RT-PCR and Serial Analysis of Gene Expression (SAGE). We will focus on cDNA microarray, as we think it offers a representative illustration of very general problems. Some data sets produced with those technologies are publicly available. A data set from a series of 7 time point measurements of the entire yeast genome (approximately 6400 measurements per time point) was produced by Brown et al and is available on the internet ([9]). More recently, a time series of 17 samples has been produced by Cho et al. The corresponding data can be found in [10].

3.2.1 From Microarray data to mRNA level

Microarray technology Microarray technology provides us with a way to measure mRNA level within the cell. A microarray is a glass slide onto which single stranded DNA molecules are attached at fixed locations. Exploiting the preferential binding properties of RNA to the corresponding DNA strand, the microarray will allow a simultaneous extraction of the mRNA.

Actually, like many other experimental technologies, microarrays measure the target quantity (mRNA abundance) by converting it into a more tractable physical quantity. Here, the technique is to compare the level of expression in the sample with the expression in a control. To do so, RNA from the sample and control cells are labelled with two different labels, e.g. a red dye for the RNA from the sample and a green dye for that of the control population. Both extracts are washed over the microarray. Then, the relative abundance is measured by exciting the array with a laser. If a spot is red, it means the RNA from the sample is in abundance; if it appears green, it means that the level is low.

Limits When looking at the data, it can be useful to know the standard error. Unfortunately, this value is largely unknown and can vary from spot to spot. Indeed, measurement uncertainty may be influenced by many factors, including spot intensity and shape ([2]). Moreover, transforming the raw data (which is an image) into the gene expression matrix requires a non-trivial processing so that some noise may be added at this step.

Moreover, the obtained data are relative by nature and will need some normalization for further use. This can represent a difficulty if we have to compare expression level between two genes.

Another difficulty is identifying each spot with the respective gene. The task is made harder by the fact that the same gene can appear in more than one spot, either with the same or with different sequences. One could then ask what expression level is to be attributed to the gene. For the reverse engineering problem, it introduces new rules to consider when trying to unravel the network. For example, if we choose that the expression level should be the sum of each different level, then we will have to take the “sum function” as a possible rule in the network.

3.2.2 From mRNA to protein level

Roughly speaking, microarray data provide us with mRNA abundancy. As the process we seek to unravel is the regulatory process from protein level to mRNA level, we will have to make some assumption about the relationship from mRNA to protein level.

This is one important step of modeling and we should pay attention to it if we want to understand what we are doing.

As we have seen, many regulatory processes may occur after transcription. Therefore, the correlation between
RNA and protein level might not be straightforward. Some comparisons between proteomic and genomic data have shown a very poor correlation between the RNA and protein abundance in some cases ([?]). This can be explained by the multiple post-transcription regulatory process. However, the comparison was done between mRNA and protein level at the same time point. Therefore, this study only tells us about the limits of the assumption that protein and RNA level are linearly related without delay. Hopefully, we will not have to rely on regulatory processes to explain these mis-correlations, but rather to take delay into account for the modeling. Thus we should continue to assume that the abundance of one protein can be deduced from the RNA concentration without consideration of other proteins. A typical equation relating the protein level \( p_i \) to the mRNA level \( r_i \) of that protein is:

\[
\frac{\delta p_i}{\delta t}(t) = K \cdot r_i(t - T) - f \cdot p_i(t)
\]

where \( T \) is a time delay representing molecular transport within the cell (typically around the order of 10 minutes), \( \frac{K}{T} \) is a constant of amplification and the term \( f \cdot p_i(t) \) accounts for the protein leakage. Figure 3.2 illustrates the meaning of these parameters.

![Figure 3.2: Qualitative meaning of parameters](image)

Once a model is chosen, the next step is to convert it to fit our discrete-time data. For example, we could choose to consider that the protein abundance at time \( t \) is well approximated by a weighted sum of mRNA levels at time \( t, t - 1 \) and \( t - 2 \), the weight being deduced from the parameter chosen in our model. It could also tell us about the proper time interval across which the measurements need to be acquired and interpreted. Note that, when collecting data, we are not picking one precise time point, but rather a time interval.
Chapter 4

Modeling methodology

Various models of the regulation process have been proposed in the literature. They differ widely on the level of biochemical details they offer. The level of abstraction ranges from the deterministic boolean model ([14, 17]) to the stochastic kinetics modeling approach ([8, 20]). Some have thought to introduce intermediate products ([25, 22]), hidden environmental variables ([27]), asynchronicity ([19]), or even spaciality ([6]). Those models might be more or less appropriate, depending on how well their rules capture basic features of the real process. Yet this faithfulness to the biological phenomenon should be put in balance with computational tractability, transparency (or equivalently its analysis simplicity) and data requirement.

Indeed, given the limited amount of data available (typically 20 measurements per gene), introducing too many parameters will result in a highly underconstrained model, so that the approximation will only be removed from the modeling step to the computational step.

Another limitation is the computational tractability. For models with high levels of biochemical details the computational cost tends to critically depend on the number of genes involved, hence being unappplicable for large-scale study. In fact, the true problem of modeling is not to produce a fully descriptive framework, but to introduce well chosen heuristics in order to fit our computational power.

The third criterion for model choice should be its ability to interact with experimental devices. It seems that a mutual interaction between experiments and computation is one of the keys for an efficient modeling. From that point of view, a highly transparent model is more likely to be efficiently tested and iteratively improved.

A model is inherently a simplifying assumption. That simplification will have its advantages and drawbacks, and its choice should depend mainly on the goals of the modeling effort and the experimental possibilities. Eventually, several models can be applied successively and work in a complementary fashion, as done in [34].

4.1 Boolean Vs Continuous

One big controversy in the field of genetic network modeling is knowing if a continuous representation of expression level is necessary to capture basic properties of gene regulation. Without consideration of biological rules, the boolean model is appreciated for its high transparency and its computational tractability. Moreover, the kind of experiment we can produce is typically boolean, that is, we can either disrupt or overexpress one given gene. On the other hand, continuous models are sometime thought to be more realistic. For an efficient discussion of this matter, one should note that gene levels of expression are indeed continuous and that idealizing them as being either ON or OFF is merely a starting point. But, we emphasize that the question is not to know if the gene expression levels spend lot of their time at intermediate values, but rather to know if the boolean rules will be consistent with the boolean abstraction.

This remark might be highlighted by a simple example:

Suppose, hypothetically, that we are observing a network whose rules are arithmetic expressions constructed with the operators + and *. Solving the boolean abstraction (considering our variable either = 0 or > 0) will allow us to determine correctly the set of inputs to each node and the correct precedence. Note that knowing if expression levels take intermediate values is totally irrelevant and this consideration will just make
the identification task harder. Note, conversely, that if our set of operators contain $\neg$, then we will miss interesting properties. Thus, the real problem is to know if biological rules have a good boolean abstraction. This analysis done, we can consider valuable critiques made to the boolean model:

1. Boolean network model does not provide a suitable framework to study important concepts of control theory: in particular amplification, subtraction and addition. ([16])

2. The model cannot capture the stabilizing effect of feedback control with moderate negative gain since boolean negative feedback always causes oscillation rather than increased stability. ([16])

3. Boolean representation ignores those genes that have different biological regulatory effects at expression level intermediate between their basal and their maximal expression level. ([27])

4. Discretization is undesirable, because it can enforce erroneous relationships which are sensitive to slight variation in the used threshold value ([33]).

5. Discrete representation does not provide an improvement in an automated inference process ([33]).

6. Some techniques for inferring the structure of continuous models scale better than methods for discrete models ([28]).

Those critiques should be considered seriously. However, they can be addressed as follows.

1. Amplification and addition of signals find a good abstraction in boolean representation as seen in the example discussed above. However, it is true that subtraction is not well captured in the boolean model. If this rule is indeed required (and eventually how often it is) remains unclear. One situation where this could appear is represented in figure 4.1 where it might be interesting to consider the quantity $|T - A|$. ($A$ and $T$ are concentrations of proteins $A$ and $T$). The problem might be partially alleviated using discrete values instead of boolean values and including the subtraction rule.

![Figure 4.1: Binding of transcription factor T is needed to induce transcription. In case (A), protein A is absent and transcription occurs. In case (B), A binds to T and prevents its binding to the cis-element, hence stopping transcription.](image)

2. Critique 2 points to dynamic properties of networks. It seems that oscillation is more directly related to synchrony than to boolean representation. Moreover, stability should not be thought the same way when considering boolean network. Either, one should study the characteristic of basin of attraction (see [15] for definition of those concepts) and see to what extent a perturbation of variables, when in one basin, lead to flip the dynamic into another basin. Furthermore, efficient computational techniques have been developed to analyze the behavior of boolean networks. This allows one to have some prior knowledge about the general characteristics our networks should have, or to analyze the results found by our algorithm.

3. For critique 3, the question is to know whether there are intermediate thresholds that will determine the qualitative outcome of regulation. It is indeed known that such properties exist, and the problem needs to be addressed. What would be interesting is knowing how general that property is. We severely lack statistical knowledge in this domain. Once again, the problem might be alleviated using discrete values.
4. Critique 4 overlaps partially with 3 but implicitly introduces the problem of noise in data. It is true that if a threshold exists, making discretization will mask the uncertainty in the data and might convert small perturbations into qualitative errors. However, experimental studies suggest that all continuous algorithms proposed so far are extremely sensitive to noise ([33]). This can be related to the need of extrapolating data to cope with underconstrainment.

5. Critique 5 suggests that one should not discretize data without good algorithmic reasons. It is true that some computational methods are intrinsically continuous and that efficient algorithms might be working without discretization. However, it is also shown in ([33]) that until now, continuous algorithms are still highly inefficient.

6. For critique 6 the question is also algorithmic. The discrete techniques somehow separate the task of wiring (finding the inputs) and the task of ruling (finding the rules at each node). In a continuous representation, you can naturally convert back and forth between representing the structure graphically and numerically.

It is clear from this discussion that boolean models will not be suited to capture every mechanism of regulation. It remains to be seen, if continuous models allow to incorporate the missing feature efficiently. Some solutions might be found in discrete models.

### 4.2 Deterministic Vs Stochastic

Another important question to be addressed is whether our model needs to be stochastic. It is clear that stochastic models include as particular cases the deterministic one. A more interesting remark is that many of the algorithms proposed for deterministic models can be discussed and analyzed with the formalism developed for dynamic bayesian networks ([28]).

There are 3 main reasons for choosing a stochastic modeling.

- The first one addresses the biological part: It has been shown that some processes in molecular biology, and in particular in differentiation, involve a small number of molecules. In those cases the stochastic representation of molecular interaction has a stronger theoretical basis than deterministic kinetics ([20, 8]). However, these effects have been mainly observed in prokaryotes. It is uncertain that they play a substantial role in eukaryotes where the regulation is thought to be more efficient and where a very low concentration of protein is more rare.

- The second one addresses the experimental acquisition process. Even supposing the underlying process is deterministic, the data will contain some noise. A purely logical inference of rules will not be able to cope with the inconsistencies due to noise, whereas a stochastic modeling allows a natural handling of noisy data. However, stochastic modeling might appear to be a heavy method and could advantageously be replaced by a simplified analysis of the variance on the expected result.

- The last reason concerns the computational part. It can be argued that stochasticity provides a good framework for incorporating hidden variables. This could be interesting when some important inputs or intermediate products are known to be missing.

If one chooses to turn to stochastic modeling, the field of dynamic boolean networks investigated by Pearl in [12] might be the most appropriate [28]. Also, if the chemical processes are known, it is possible to write them as stochastic kinetic equations (e.g. [8]). Those equations can be converted naturally and simulated efficiently using the formalism of Stochastic Petri Net [20].

However, stochasticity adds a whole new level of complexity to the models, so that until now, all stochastic approaches have been restricted to the study of a very limited number of genes. It remains very uncertain that the corresponding algorithms scale up efficiently.
4.3 Time representation

Some models, in particular those based on differential equations, are continuous time networks. This can be a good way of thinking to design the model. However, one should remember that our data is time discrete, as they represent discrete “snapshots” of gene expression at various time points and environmental conditions. Thus the model will have to be eventually transferred to discrete time. Conversely, discrete time models will need to have a time continuous interpretation.

A more important issue is to know how to deal with time delays occurring in transcription and translation and the decay rate of protein and mRNA. It is known that those quantities will vary from one gene to another but little is known about the statistical properties of those quantities.

In order to cope with the variance of those temporal parameters, some models have proposed asynchronous updating of the network state. It is the case of qualitative models ([19]) and some stochastic models [8, 20]. Some others have proposed to include several time steps (or, equivalently, several nodes) per gene as inputs (e.g. [31]). This could be an interesting idea if we are able to introduce efficient relations between those time steps, but we should also be aware this method introduces some redundancy in our inputs.

The most radical approach is to consider only steady state. This is equivalent to make models with no time delay. This approach simulates the synchronous case if we separate the network into two sub-networks representing the two time points or, equivalently, the inputs and the outputs.

4.4 Linear Vs Non-linear

The process of regulation is known to be highly non-linear. It is not unusual, for example, to find that in the presence of A or B alone, the transcription of C never exceeds 10% of the rate which is achieved when both A and B are present. Three main methods have been proposed to take into account this non-linear behavior.

The first is to make qualitative (i.e. discrete) models. The second, is to consider neuron-like behavior, where the activation of a given gene is linear, but the transcription rate is a non linear function of the activation ([6, 27]). The third possibility is to consider non linear equations. In particular, power law models are well appreciated ([22, 34]). All those solutions have proved their efficiency in other fields. However, until now, the actual rules of regulation and their statistical properties remain largely unknown.
Chapter 5

Algorithms

When designing an algorithm for inferring functional relationships between genes, we should keep in mind some specific aspects of reverse engineering in the field of genetic networks ([33]). In particular,

- The number of measurements is very small compared to the number of genes.
- Unlike in most machine learning fields, we are more interested in parameter fitting (that hopefully have biological meaning), than about predictive power on gene expression level.
- The data tends to be very noisy.
- The number of inputs per gene is small.

We present here a representative sample of the algorithms available. Those examples have been selected because they are quite popular and incorporate the main ideas and concepts of the field.

5.1 Discrete Algorithms

5.1.1 REVEAL [17]

<table>
<thead>
<tr>
<th>Type of model</th>
<th>Boolean, Deterministic, Synchronous</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type of rules</td>
<td>Indegree bounded by $k_{max}$</td>
</tr>
<tr>
<td>Type of data</td>
<td>Transition state with random inputs</td>
</tr>
</tbody>
</table>

Principle of the algorithm and mathematical concept used: REVEAL is a very popular algorithm in the field. The algorithm tries to infer the inputs of a given output without reference to the rule existing between those inputs and the output. This is made by measuring the information contained in one set of input/output data.


For our matter, all we need to know is that the amount of information given by the observation of a set of variables $X$ (in our case, the expression level of a set of genes) can be quantified as the Shannon entropy of $X$, $H(X)$. $H(X)$ is calculated from probability law of the set of variables $X$. This probability law, and thus $H(X)$, can be approximated by statistical observation of the state of variables $X$. Moreover, the approximation obtained by observing $t$ time points can be calculated in a time linear in $t$.

To get an intuitive idea of $H$ one should refer to figure 5.1, where the information contained in the set of variable $X$ (resp $Y$) is represented as a subset of $\mathbb{R}^2$ and its entropy $H(X)$ (resp $H(Y)$) is its area. The idea of REVEAL to know which inputs are controlling output $Y$, is to approximate $H(X \cup \{Y\})$ and $H(X)$ for subset $X$ of less than $k_{max}$ inputs. If $H(X \cup \{Y\}) = H(X)$, it means that $Y$ is totally determined by $X$, and we have found our inputs. Then, it is easy to find boolean function taking $X$ as input which is consistent with our time series.
Algorithm:

```plaintext
for all Y output do
  for k = 0 to k_{max} do
    for all X = \{X_1, \ldots, X_k\} set of input do
      if H(X_1, \ldots, X_k, Y) = H(X_1, \ldots, X_k) then
        Find a boolean rule with \{X_1, \ldots, X_k\} as input consistent with our input and attribute it to Y;
      end if
    end for
  end for
end for
```

Results and discussion: The computational complexity of the algorithm for inferring a Network of N genes from a data of t transition states is \(\Omega(N \cdot t \cdot C_N^{k_{max}})\).

From computational experiments, the data requirement seems to be of the form of \(f(k_{max}) \cdot \log(N)\) transition states the inputs are uniformly randomly distributed. So for fixed \(k_{max}\), the computational complexity is \(\Omega(N^{k_{max}+1} \cdot \log(N))\).

The method has been tested on a sample of artificial boolean networks of N=50 genes with indegree bounded to 3. It required about 100 transition states (with uniformly distributed inputs) to recover the networks correctly.

5.1.2 ID3 [31]

<table>
<thead>
<tr>
<th>Type of model</th>
<th>boolean, deterministic, including T time steps as input</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type of rules</td>
<td>Indegree bounded by (k_{max})</td>
</tr>
<tr>
<td>Type of data</td>
<td>Time series</td>
</tr>
</tbody>
</table>

Principle of the algorithm and mathematical concept used: The model is a boolean network in which the expression level of a given gene at time \(t\) is a boolean function of the state of other genes at time \((t, t-1, \ldots, t-T)\). This generalization of classical boolean network is motivated by the will to address interaction between genes that span for more than one unit of time. This could occur because of delay in translation, formation of complex, or an intermediate factor not being monitored.

The algorithm The algorithm of inference, called ID3, has been described in the machine learning literature. It can be thought of as a greedy version of REVEAL. The rules are represented as decision trees of length at most \(k_{max}\), and ID3 makes a greedy search through the space of decision trees of length at most \(k\), in order to identify a compact tree consistent with the data. This search is guided by the Shannon entropy reduction.

More precisely, for each output, a tree is constructed from the root to the leaves. At each step of that
construction, the input being the most informative on the output (in the sense of Shannon entropy) is added as a new node of the decision tree.

**Results and discussion:** The performance of the method was tested on artificial networks of 16 genes, with \(k_{max}\) varying from 2 to 10 and \(T\) varying from 3 to 8. The data used for each target network was 20 time series of 100 time points each. The result are good in term of sensitivity, that is most of the real wirings were predicted. However, the specificity, i.e. the probability that a predicted wiring is real, is poor. Anyway, the data requirement is too high for practical cases.

An idea for improvement would be to increase the number of time step iteratively for each gene until a good approximation is found. We should also restrict the rules to be applied. For example, the same gene should not be found as inducer in one time step and as inhibitor in another.

### 5.1.3 **Predictor-Chooser** [29]

<table>
<thead>
<tr>
<th>Type of model</th>
<th>Boolean, Deterministic, without time delay</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type of rules</td>
<td>No restriction</td>
</tr>
<tr>
<td>Type of data</td>
<td>Observation under perturbation</td>
</tr>
<tr>
<td></td>
<td>(in each experiment, any number of protein level may be forced either high or low)</td>
</tr>
</tbody>
</table>

**Principle of the algorithm and mathematical concept used:** The method is designed to create an interaction between the experimental step and the computational step. At the computational step, the *Predictor* determines a set of Boolean networks consistent with our data, favouring the one with the most parsimonious wiring. Then, the *Chooser* makes a scoring based on Shannon entropy to choose between a set of further experiments. This choice is done so as to discriminate optimally among the networks proposed by the *Predictor*.

**Algorithm:**

The *Predictor*:

```java
for all Y output do
    (1) for all (e,f) experiments where Y has not been forced high or low do
        if the expression level of Y differs in those experiments then
            Construct the set \(S_{e,f}\) of all inputs whose expression level changed between the 2 experiments;
        done;
    (2) Find the smallest set of inputs \(S_{min}\) such that at least one element of \(S_{min}\) is present in each \(S_{e,f}\)
    (3) Find a boolean rule with \(S_{min}\) as input consistent with our input and attribute it to Y;
    done;
```

The task (2) called *minimum set covering* is known as NP-complete. However, efficient algorithms exist to find an approximate solution. In the original algorithm it has been chosen to bind the maximum indegree, which allows to solve the problem in polynomial time. At this stage, several sets of consistent inputs and boolean rules might be proposed. For that reason several proposals will be made for \(Y\).

Now, we suppose that the set \(R\) of boolean networks have been proposed by the predictor, and we want to find which experiment in a set \(E\) will best discriminate between the networks in \(R\).

The *Chooser*:

```java
for all e in E do
    (1) Calculate the entropy of networks outputs under experiment \(e\), the probability measure on the networks space being taken uniformly over \(R\).
    (2) Choose the experiment \(e\) that maximizes this quantity.
done;
```

16
The task (1) can be rewritten as follow. Find the set $S$ of all patterns of output of the networks in $R$ under the experiment $e$. Find, for each $s \in S$, the number $l_s$ of networks in $R$ having output $s$. Then calculate

$$H_e = - \sum_{s \in S} \frac{l_s}{|R|} \log \left( \frac{l_s}{|R|} \right)$$

**Results and discussion:** The authors tested the *Predictor* and the *Chooser* separately on representative samples of artificial boolean networks. Those target networks were restricted to be acyclic and of bounded indegree. The acyclicity makes the response of a given network unique under any experiment but is not likely to reflect biological networks.

The *Predictor* was given the output in the wild state and in each single input perturbation. For a network of $N = 100$ genes of maximum indegree $k_{\text{max}} = 2$, the *Predictor* found 77% of the actual wiring and was wrong concerning 12% the wiring he had predicted. For a network of $N = 20$ and $k_{\text{max}} = 4$, the *Predictor* correctly found 51% of the actual wiring and was wrong concerning 16% the wiring it had predicted.

The *Chooser* was tested on a network of $N = 20$ genes and maximum indegree $k_{\text{max}} = 4$. First, all single input perturbations were done and the result was fed to the *Predictor*. Then, the *Chooser* was iteratively asked to choose between every double gene perturbation. It required 16 double gene perturbations to select a single model that turned out to be the good one. More generally, experimental results tend to show that the average number of experiments necessary to infer the network is in the form $f(k) \cdot \log(n)$.

These results are not very impressive. In particular the data requirement proposed for the *predictor* alone is linear in the number of genes which is too high.

There might be several ideas for improvement. First the choice of Shannon entropy for the criterion is very trendy in the field but it seems to us that we seek to minimize the expectation of the number of networks after experiment. In this case, the criterion we should seek to maximize is $-\Sigma_{s \in S} \left( \frac{l_s}{|R|} \right)^2$. Moreover the scoring of an experiment could incorporate its cost. Also, the predictor could be given more latitude so that the chooser could work on a more representative sample of networks. Furthermore, the chooser could be used from the first experiment.

### 5.2 Continuous Algorithms

A comparative study of some of the algorithms described below have been conducted in [33].

#### 5.2.1 D’Haeseleer99 [23]

<table>
<thead>
<tr>
<th>Type of model</th>
<th>Continuous, Deterministic, Synchronous</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type of rules</td>
<td>Linear</td>
</tr>
<tr>
<td>Type of data</td>
<td>Measurement at time point during development and injury</td>
</tr>
</tbody>
</table>

**Principle of the algorithm and mathematical concept used:** The expression of gene $i$, $X_i$, at time $t + \Delta t$ is modelled as a weighted sum of the expression of all other genes at time $t$. A constant bias term $C_i$ is added to represent the activation level of the gene in absence of any input.

$$X_i(t + \Delta t) = \sum_j W_{ij} X_j(t) + C_i$$

The authors attempted to fit this model on a data set, obtained from a study on rat, consisting of 65 genes by 28 time points including,

1. measurement at 9 different stages during cervical spinal cord development
2. measurement at 9 different stages during hippocampus development
3. measurement at 10 more time points following hippocampus injury by injection with Kainate.

The influence of Kainate on gene $i$ is written as $K_i \cdot \text{Kainate}$. Furthermore, in order to simulate environmental variables that may differ from spinal cord to hippocampus, the value of the constant bias $C_i$ differs from experiments 1 to experiments 2 and 3. The final expression becomes:

$$X_i(t + \Delta t) = \sum_j W_{ij} X_j(t) + K_i \cdot \text{Kainate} + C_i(\text{experiment})$$

**Algorithm:** There are 68 parameters per gene against only 28 time points. The solution chosen to cope for underconstrainment is the interpolation of the data points. More precisely, a cubic interpolation of the log of expression level was calculated. Given those time series, finding the parameters requires solving a least-squares system of linear equations or equivalently, performing a multiple regression of each gene to all other genes.

**Results and discussion:** The method could not be evaluated directly since the regulation of the 65 genes monitored is largely unknown. The statistical properties of weight $W_{ij}$ derived by this method seems in agreement with our biological knowledge. Nevertheless, the conclusion seems rather unreliable since the coefficients $K_i$ are rather small, which is in contradiction with the dramatic and almost instantaneous change in gene expression levels caused by kainate injection. It remains unclear that interpolation can resolve the dimensionality problem.

### 5.2.2 Chen99a [24]

<table>
<thead>
<tr>
<th>Type of model</th>
<th>Continuous, Deterministic, Time continuous</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type of rules</td>
<td>Individual activation or inhibition</td>
</tr>
<tr>
<td>Type of data</td>
<td>Time series</td>
</tr>
</tbody>
</table>

**Principle of the algorithm and mathematical concept used:** The model does not really specify the type of rule involved, but rather try to identify the inputs for each output. This is done in a two time process:

- First they create the graph of interaction, with edges labeled either $I$ for inhibitor or $A$ for activator. This graph is found by comparing the occurrences of peaks in the expression data. For example if a peak in expression level of $X$ occurs slightly before a peak in expression level of $Y$, then $X$ is a candidate activator of gene $Y$, and the edge $(X, Y)$ is added to the graph and labeled $A$.

- Then, excess edges of the graph are deleted with the following criterion. Each gene is assigned either an activator or an inhibitor label. That is, the remaining edges from each vertex are either all activator or all inhibitor. The simplification made here is that transcription factors are either inhibitor or activator but not both. This assumption is biologically reasonable even if exception have been reported. The assigning is made so as to maximize the number of vertices with both $A$ and $I$ inputs. We also seek to obtain the sparsest possible wiring so that some extra edges are deleted.

**Algorithm:** The original algorithm is unclear and some of the scoring are doubtful. For that reason, small differences might appear between the algorithm reported here and the original one.
(1) for all $X$ gene do
  Find the peaks in the time serie of $X$;
  \% (A peak is defined as a set of consecutive data points with greater than average expression, \%)
  Define each peak $P$ by its start point $P_{\text{start}}$, maximum point $P_{\text{max}}$, end point $P_{\text{end}}$ and its height $P_h$;
(2) for all $X, X'$ genes do
  (2.1) Align the peaks $X$ and $X'$
          and match together the peaks $P$ of $X$ directly preceding the peaks $P'$ of $X'$;
  (2.2) Evaluate the activation power of gene $X$ on gene $X'$ as:
        \[ G_a(X, X') = \sum_{P, P'} G_a(b(P, P') - U(P) - U(P')) \]
        where \( G_a(b(P, P') = C_a/\exp(P_{\text{start}} - P_{\text{start}} + |P_{\text{max}} - P_{\text{max}}|/2) \) evaluates the activation
        of $P$ on $P'$, and $U(P) = P_{\text{end}} - P_{\text{start}}$ evaluates the importance of $P$.
  (2.3) Use a similar function to evaluate the inhibition power of gene $X$ on gene $X'$;
  (2.4) If the inhibition or activation power of gene $X$ exceeds $\theta$ then
        Add the edge $(X, X')$ labeled with its meaning (A or I) and its numerical weight;
(3) Delete some of the edges and make a consistent labeling of the vertices;
    \% (A vertex labeled $A$ (resp I) must have all remaining edges labeled $A$ (resp I)) \%

The optimizer used here makes a simulated annealing seeking the subgraph $G$ that maximize
\[ f(G) = \sum_X \text{gene}(\max(A) \cdot \max(I)) - K \cdot \text{number of edges} \]
where $\max(A)$ (resp $\min(A)$) is the maximal weight of the activator (resp inhibitor) in-edge.

Results and discussion: The authors tested the method on biological data. In fact, instead of reasoning
on the genes, they choose to make a clustering and reason uniquely on clusters. The interpretation of an
activation or inhibition by a cluster is not given. On the whole, the results are very uncertain.

5.2.3 TReMM [27]

<table>
<thead>
<tr>
<th>Type of model</th>
<th>Continuous, deterministic, synchronous</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type of rules</td>
<td>Neuron like</td>
</tr>
<tr>
<td>Type of data</td>
<td>Time serie</td>
</tr>
</tbody>
</table>

Principle of the algorithm and mathematical concept used: The regulatory input to gene $i$, $r_i$, is
modeled as a weighted sum of the expression of all other genes at time $t$. A constant bias term $C_i$ is added
to represent the activation level of the gene in absence of any input.

\[ r_i(t + \Delta t) = \sum_j W_{ij} X_j(t) + C_i \]

In turn, the expression of gene $i$ at time $t + \Delta$ is modelled as a “squashing” function of the regulatory
input to gene $i$, at time $t$.

\[ r_i(t + \Delta t) = m_i f(r_i(t)) \]

The “squashing” function $f$ is defined by $f(r) = \frac{1}{1 + e^{-r}}$ (see figure 5.2), and $m_i$ is the maximum level of
expression of gene $i$, and is supposed to be known.

Algorithm:

(1) Put the data set in the form of a matrix \( X = (X_i(t))_d \)
(2) Desquash the time series of $X_i$ and create the matrix \( R = (r_i(t))_d \)
    \% (The desquashing operation is made by $r_i(t) = -\ln(\frac{n_i}{X_i} - 1)$) \%

At this stage, they try to find the matrix \( W = (W_{ij})_d \) such that \( R = WX \). In practical cases, the number of
genes exceed by far the number of data points, and the problem is underconstrained; that is, the matrix $W$
is not uniquely defined by this equation. Thus, the problem is not to find the inverse of \(X\) but the correct inverse of \(X\).

To cope with underconstrainment the authors rely on 2 ideas:

- Our data can be split into a training set, used to determine \(W\), and a test set, on which to test our predicted \(W\).
- \(W\) is expected to be very sparse, so that they can iteratively force some coefficients \(W_{ij}\) to be 0.

Those ideas are underlying step (3).

(3) \textbf{for all gene i do}

\[
M_i := X;
\]

\textbf{While } \(M \neq 0\) \textbf{ do}

- Calculate \(M^{-1}\);
- Calculate \(V = R_i \ast M^{-1}\). and evaluate its predictive power on the test set;

\%\% \textit{by using Moore-Penrose pseudo inverse or singular value decomposition}\%

\textbf{If } \(V\) has better predictive power than \(W_i\) \textbf{ then } \(W_i := V\); \%\% \textit{\(W_i\) is the \(i\)th row of } \(W\) \%

\textbf{done};

\textbf{done};

\textbf{Results and discussion:} The authors tested the algorithm on artificial networks of 20 and 50 genes with indegree from 2 to 20. The algorithm seems to perform well, even with noisy data, when the number of data point is of the same order as the number of genes. However, this is not the situation for which this algorithm was designed, neither a practical situation.

Some limits specific to this model come from the sensitivity of the desquashing function when the level of expression of the gene is near its maximal or minimal expression.

### 5.2.4 AIGNET [34]

This system makes a former analysis under a boolean model, and a latter under a continuous model.

<table>
<thead>
<tr>
<th>Former Analysis</th>
<th>Type of model</th>
<th>Boolean, deterministic, without time delay</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type of rules</td>
<td>Individual activation or inhibition</td>
<td></td>
</tr>
<tr>
<td>Type of data</td>
<td>Observation under perturbation</td>
<td></td>
</tr>
<tr>
<td></td>
<td>(in each experiment, 1 protein level may be forced either high or low)</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Latter Analysis</th>
<th>Type of model</th>
<th>continuous, deterministic, continuous time</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type of rules</td>
<td>Power-law differential equations</td>
<td></td>
</tr>
<tr>
<td>Type of data</td>
<td>time course observation</td>
<td></td>
</tr>
</tbody>
</table>

20
Principle of the algorithm and mathematical concept used:
Under the boolean model, the authors make a topological analysis of gene interaction so as to reduce the search space for the second stage.
Under the continuous model, they model interaction between gene by S-System, a power-law formalism. The expression level of a given gene $X_i$ is thought to be governed by a differential equation of type:

$$\frac{d}{dt}X_i = \alpha_i \prod_{j=1}^{n} X_j^{g_{i,j}} - \beta_i \prod_{j=1}^{n} X_j^{h_{i,j}}$$

Hence, the number of parameters to determine, for a network of $N$ genes, is $N(2N+2)$.

Algorithm:
Under the boolean model:
They observe the expression pattern of the wild state as well as each state resulting from the perturbation of a single gene. We note $p_0$ the wild state and $p_i$ the experiment resulting of perturbation of gene $i$. We further note $p_i(j)$ the level of gene $j$ under experiment $p_i$. The topological analysis of gene interaction is made as follows:

(1) for all $(i,j)$ genes do  
   If $|\log \left( \frac{p_0(j)}{p_i(j)} \right) | > \theta$ then  
      Add arc $(i,j)$ to the graph-of-influence;  
      \%\% The graph-of-influence is represented as a matrix \%\%  
   done;

(2) Make the transitive clojure of the graph-of-influence;  
      \%\% By multiplication of the matrix \%\%

(3) Determine the stongly connected components (cliques);

(4) Determine the graph of influence between the cliques;

Under the continuous model:
The former analysis have allowed to restrict the number of possible inputs for a given $Y$. The inputs to consider are only the genes in the same cliques or the one directly affecting the cliques. Still the model is highly underconstrained, so they use a GA for investigating the search space.
The parameters to fit are the $\alpha_i, \beta_i, g_{i,j}$ and $h_{ij}$. The evaluation of a set of parameters is made by measuring its predictive power. More precisely, they try to find the least square fits between experimental values $X_{i,exp}$ and predicted values:

$$\sum_{i=1}^{N} \sum_{t=1}^{T} \left( \frac{X_{i,calc,t} - X_{i,exp,t}}{X_{i,exp,t}} \right)^2$$

Results and discussion: The authors tested their method on an artificial network of 30 genes formulated in the S-System formalism. The network showed very parsimonious wiring since the indegree was bounded 2 (except for one gene of indegree 3) and averaged 1.3. 31 time course were generated. The topological analysis was performed correctly. The S-System inference was tested on 3 genes of indegree 1. The computational cost was extremely high as was the data requirement. At the end, the standard error between calculated and experimental value was 0.86% per time point.
This result is poor. However, the idea to refine the model at each step of the computation should be kept.
Chapter 6

Constraining the model

It appears from the results of the algorithms above that a gap remains between the experimental possibilities and the data requirement. This problem is often referred to as the dimensionality problem. The number of genes exceed by far the number of data points. Some theoretical bounds have been established for the discrete case [26]. We report them since they provide some insight into what can be hoped for on the computational side.

In the discussion below the number of genes will be denoted by \( N \) and the maximum indegree by \( K \).

**Theorem 6.1** [26] If \( O(2^K \cdot (2K + a) \cdot \log(N)) \) input patterns are given uniformly randomly, then with probability \( \geq 1 - \frac{1}{N} \), there exists at most one boolean network of \( N \) nodes with maximum indegree \( \leq K \) consistent with given input/output pairs.

Conversely, a simple information theoretic argument shows that:

**Theorem 6.2** At least \( \Omega(2^K + K \cdot \log(N)) \) input/output pairs are necessary to identify the network in the worst case.

The dependency in \( N \) for discrete model is thus well established, and is better than in most continuous models where the number of required data points is in \( \Omega(N) \). However, there is a great uncertainty in the constant factor involving \( K \). From experimental results, the real factor would be close to \( K \cdot 2^K \cdot \log(N) \).

In the case of continuous models, the results are less clear. In most cases, the number of parameters, thus the needed number of data points, is in \( \Omega(N) \). For sparse additive regulation models, we can speculate the data requirement to be similar to the discrete case that is in \( \Omega(\log(N)) \).

There are several ways to reduce the complexity, hence the data requirement. First we can try to reduce the dependency in \( N \). A good way, is to reduce \( N \) itself, that is, to reason on sets of genes rather than on genes themselves. This reduction make sense, since it is a common observation that many genes show highly correlated expression patterns.

The method is called *clustering* and the resulting sets of genes are referred as *clusters*. The principle is to choose a distance to quantify how close two expression patterns are. Then, we join in the same cluster the closest genes with respect to the distance. The process is illustrated in figure 6.1

![Figure 6.1: Clustering](image-url)
Various distance choices and clustering algorithms have been applied to gene expression patterns [16]. The results have interest of their own, as genes showing closely correlated expression patterns are likely to be involved in the same biological process. For the reverse engineering problem, clustering might be an interesting preprocessing as it artificially reduces \( N \). However, once clustering is done, the problem remains to discover the regulation interactions that controls the behavior of clusters. Thus, we will have to unravel the regulatory network of clusters just as if they were genes. For that reason, we have chosen not to include the clustering step in the description of algorithms. If we want to include clustering as the first step, we will have to address at least two questions. First, what expression pattern is to be taken as representing a cluster? Second, and most importantly, how should we interpret the regulation of one cluster on another cluster? For example, if we think that cluster \( A \) induces cluster \( B \), we need to know which gene of \( A \) acts on which gene of \( B \). This question is usually well ignored.

We can also try to reduce the constant factor involving \( K \), the maximum indegree. This should be done by the use of additional constraints. Those constraints can be derived either from biological knowledge or computational experiments. The principle to obtain the latter is to set some statistical parameters and to generate random networks according to these parameters. Then, we simulate the networks and evaluate how well they mimic biological behavior. In return, it provides some insight into the statistical properties we should expect from the network we are trying to unravel.

We can report here some of the ideas that might be exploited:

- Intuitively, the rules that appear in regulation should be monotonic. For a given output, the same input should not be found as both inhibitor and activator, depending on the level of other proteins.

- Most transcription factors are either activators or inhibitors, so that the monotonicity can somehow be extended to the set of all outputs.

- Some simple statistical properties of expression patterns seems to be linked to the way regulation is achieved ([22]). Hence, one can have some idea of the rules that might control a given output independently of any input.

- We could also imagine to include clustering as the first step but continue to reason on genes themselves. For instance, clustering the expression patterns of outputs might allow one to transfer some results found for one output to its closest neighbors. The clustering of the inputs may help to reduce redundancy in our data. In particular, if the same gene is present several time on the microarray chips, we might find it this way.

- Some genomic analysis could be introduced to complete our data. If we are able to identify some of the most important cis-elements and their corresponding transcription factors, we can extract some statistical knowledge. We can also try to obtain gene specific information. In particular, if we are able to discover the cis-elements of a given gene, we can use the binding strength between transcription factors and binding sites to discover the relevant inputs for that output. The main limitation of that perspective is the weakness of the current methods for the cis-elements prediction.

These are some proposals, among many, to constrain the model by including external informations. In the future, we should try to include as much information as possible in our model building. Doing so, we can hope to achieve higher efficiency in reverse engineering methods.
Bibliography


[9] The complete data set is available at cmgm.stanford.edu/pbrown/explore/index.html


