Machine Learning and Genetic Regulatory Networks: A Review and a Roadmap

Christopher Fogelberg[1] and Vasile Palade[1]

Oxford University Computing Laboratory, Wolfson Building, OX1-3QD, UK.
Contact email: christopher.fogelberg@comlab.ox.ac.uk

Summary. Genetic regulatory networks (GRNs) are causal structures which can be represented as large directed graphs. Their inference is a central problem in bioinformatics. Because of the paucity of available data and high levels of associated noise, machine learning is essential in performing a good and tractable inference of the underlying causal structure.

This chapter serves as a review of the GRN field as a whole, as well as a roadmap for researchers new to the field. It describes the relevant theoretical and empirical biochemistry and the different types of GRN inference. It also describes the data that can be used to perform GRN inference. With this biologically-centred material as background, the chapter primarily focuses on previous applications of machine learning techniques and computational intelligence to GRN inference. It describes clustering, logical and mathematical formalisms, Bayesian approaches and their interaction. Each of these is shortly explained theoretically, and important examples of previous research using each are highlighted. Finally, the chapter analyses wider statistical problems in the field, and concludes with a summary of the main achievements of previous research as well as some open research questions in the field.

1 Introduction

Genetic regulatory networks (GRN) are large directed graphs which cause the phenotypic states of biological organisms. Inference of their structure and parameters is a central problem in bioinformatics. However, because of the paucity of the training data and its noisiness, machine learning is essential to good and tractable inference. How machine learning techniques can be developed and applied to this problem is the focus of this review.

Section 2 summarises the relevant biology, and section 3 describes the machine learning and statistical problems in bioinformatic network inference.

Sections 4 discusses biological data types that can be used, and section 5 describes existing approaches to *network inference*. Section 6 describes important and more general statistical concerns associated with the problem of inference, and section 7 provides a brief visual categorisation of the research

in the field. Section 8 concludes the survey by describing several open research questions.

Other reviews of GRN include [25], [19] and [26]. However, many of these are dated. Those that are more current focus on presenting new research findings and do not summarise the field as a whole.

2 The Underlying Biology

A GRN is one kind of regulatory (causal) network. Others include protein networks and metabolic processes[76]. This section briefly summarises the cellular biology that is relevant to GRN inference.

2.1 Network Structure and Macro-Characteristics

GRN have a *messily robust* structure as a consequence of evolution[106]. This subsection discusses the known and hypothesised network-level characteristics of GRNs. Subsection 2.2 describes the micro-characteristics of GRNs.

A GRN is a directed graph, the vertices of this graph are genes and the edges describe the regulatory relationships between genes. GRN may be modeled as either directed[100] or undirected[111] graphs, however the true underlying regulatory network is a directed graph. Recent[6] and historical[56] research shows that GRN are not just random directed graphs. Barabasi and Oltvai [6] also discusses the statistical macro-characteristics of GRN.

The Out-degree (k_{out}), and In-degree (k_{in})

GRN network structure appears to be neither random nor rigidly hierarchical, but *scale free*. This means that the probability distribution for the out-degree follows a power law[6; 56]. I.e., the probability that i regulates k other genes is $p(k) \approx k^{-\lambda}$, where usually $\lambda \in [2, 3]$. Kauffman's [56] analysis of scale free Boolean networks shows that they behave as if they are on the cusp of being highly ordered and totally chaotic. Barabasi and Oltvai [6] claims that "being on the cusp" contributes to a GRN's evolvability and adaptability.

These distributions over k_{in} and k_{out} means that a number of assumptions have been made in previous research to simplify the problem and make it more tractable. For example, the exponential distribution over k_{in} means that most genes are regulated by only a few others. Crucially, this average is not a maximum.

This means that techniques which strictly limit k_{in} to some arbitrary constant (e.g. [100; 108]) may not be able to infer all networks. This compromises their explanatory power.

Modules

Genes are organised into modules. A module is a group of genes which are functionally linked by their phenotypic effects. Examples of phenotypic effects include protein folding, the *cell development cycle*[96], glycolysis metabolism[95], and amino acid metabolism[5].

Evolution means that genes in the same module are often physically proximate and *co-regulated*, even *equi-regulated*. However, a gene may be in multiple modules, such genes are often regulated by different genes for each module[6; 54].

One or two genes may be the main regulators of all or most of the other genes in the module. It is crucial to take these "hub" genes into account, else the model may be fragile and lack biological meaning[96].

Genetic networks are enormously redundant. For example, the `Per1`, `Per2` and `Per3` genes help regulate circadian oscillations in many species. Knocking out one or even two of them produces no detectable changes in the organism[58]. This redundancy is an expected consequence of evolution[6].

Known modules range in size from 10 to several hundred genes, and have no characteristic size[6].

Motifs

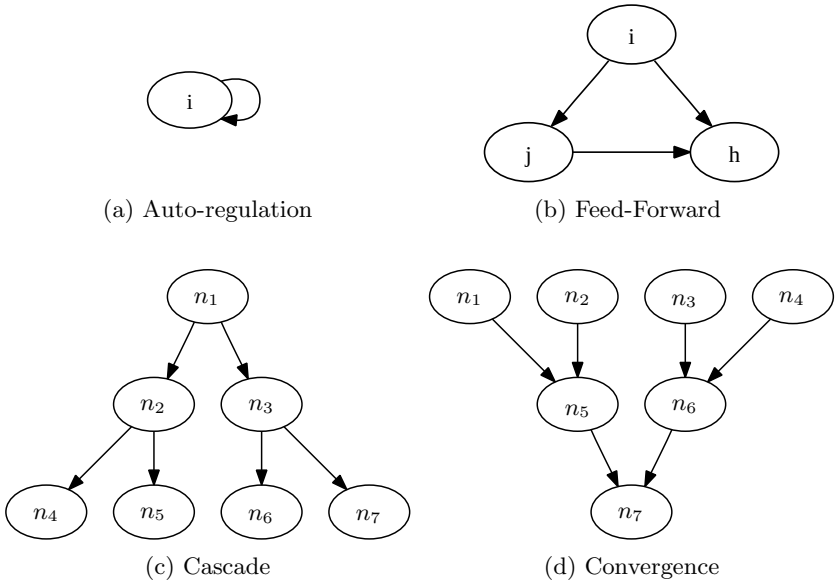
This subsubsection discusses motifs. A motif is a sub-graph which is repeated more times in a GRN than would be expected if a graph with its edge distributions were randomly connected[58]. For example, the feed-forward triangle shown in figure 1(b) frequently occurs with module-regulatory genes, where one module-regulatory gene binds to another and then both contribute to the module's regulation[5].

Auto-regulation (usually self-inhibition[24]) is also over-represented, as are the *cascade* and *convergence* motifs. Each of these three is illustrated in figure 1. The biases in network structures that motifs represent can be used to guide, describe and evaluate network inference.

Like modules, motifs often overlap. In addition, they are strongly conserved during evolution[17; 49].

2.2 Gene-Gene Interactions and Micro-Characteristics

Subsection 2.1 described the graph-level statistical properties of GRN. This subsection considers individual gene-gene interactions. Crudely, i may either up-regulate (*excite*) or down-regulate (*inhibit*) j. Different formalisations model this *regulatory function* to different degrees of fidelity.



(a) Auto-regulation (b) Feed-Forward

(c) Cascade (d) Convergence

Fig. 1. Network motifs in genetic regulatory networks. The auto-regulatory, feed-forward, cascade and convergence motifs.

One-to-One Regulatory Functions

Imagine that i is regulated only by j. The regulatory function, $f_i(j)$, may be roughly linear, sigmoid or take some other form, and the strength of j's effect on i can range from very strong to very weak.

Also consider non-genetic influences on i, denoted ϕ_i. In this situation, $i' = f_i(j, \phi_i)$. Inter-cellular signaling is modeled in [74] and is one example of ϕ. In many circumstances we can assume that $\frac{\delta f}{\delta \phi} = 0$.

It is also possible for one gene to both up-regulate and down-regulate another gene. For example, j might actively up-regulate i when j is low, but down-regulate it otherwise. However, the chemical process underlying this is not immediately clear, and in the models inferred in [89] a previously postulated case of this was not verified. In any case, this kind of especially complex relationship is not evolutionarily robust. For that reason it will be relatively rare.

Wider properties of the organism also have an influence on the kinds of regulatory functions that are present. For example, inhibitors are more common in prokaryotes than in eukaryotes[48].

Many-to-One Regulatory Functions

If a gene is regulated by more than one gene its regulatory function is usually much more complex. In particular, eukaryotic gene regulation can be

enormously complex[28]; regulatory functions may be piecewise threshold functions[18; 96; 115]. Consider the regulatory network shown in figure 1(d). If n_2 is not expressed strongly enough, n_1 may have no affect on n_5 at all.

This complexity arises because of the complex indirect, multi-level and multi-stage biological process underlying gene regulation. This regulatory process is detailed in work such as [19; 26; 115].

Some of the logically possibly regulatory relationships appear to be unlikely. For example, it appears that the *exclusive or* and *equivalent* relationships are biologically and statistically unlikely[67]. Furthermore, [56] suggests that many regulatory functions are *canalised*. A canalised[106] regulatory function is a function that is buffered and depends almost entirely on the expression level of just one other gene.

The Gene Transcription Process

Figure 2 shows how proteins transcribed by one gene may bind to several other genes as regulators and is based on figures in [19; 26]. The transcription process itself takes several steps.

First the DNA is transcribed into RNA. The creation of this fragment of RNA, known as *messenger RNA* (mRNA), is initiated when *promoters* (proteins that regulate transcription) bind ahead of the start site of the gene and cause the gene to be copied. The resulting fragment of RNA is the genetic inverse of the original DNA (i.e. an A in the DNA is transcribed as a T in the RNA). Next, the mRNA is translated into a protein.

The transcribed protein causes phenotypic effects, such as DNA repair or cellular signaling. In addition, some proteins act as promoters and bind to genes, regulating their transcriptional activity. This completes the loop.

Note that the term "motifs" is also used to refer to binding sites, to describe the kinds of promoters which can bind with and regulate a particular gene. For clarity, the term is not used in this way in this chapter. For example, [90] uses prior knowledge of protein-binding site relationships, not prior knowledge of motifs.

In summary, a GRN is a stochastic system of discrete components. However modeling a GRN in this way is not tractable. For that reason we do not consider stochastic systems in this chapter. Instead, a continuous model of GRN is used. In this model a GRN is a set of genes N and a set of functions F, such that there is one function for each gene: $\forall n \in N, \exists f_n : f_n \in F$. Each of these functions would take all or a subset of N as parameters, and ϕ_n as well. Using this sort of model the important features of the regulatory relationships can be inferred and represented.

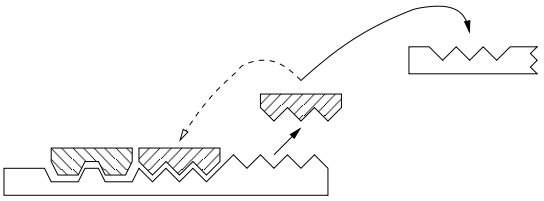


Fig. 2. A protein's-eye view of gene regulation. The complete gene displayed in the figure is regulated by itself and one other gene. We have omitted the mRNA → protein translation step for the sake of clarity. A just-transcribed protein is shown in the figure as well. Right now it cannot bind to the gene which transcribed it, as that site is occupied. It may bind to the gene on the right hand side of the figure, or it may go on to have a direct phenotypic effect.

3 Outstanding Problems

The key machine learning problems in biological inference are very closely related. Machine learning can either be used to infer *epistasis* (determine which genes interact), or create explanatory models of the network.

Epistasis is traditionally identified through *synthetic lethality*[38; 68; 112] and *yeast two-hybrid* (Y2H) experiments. Machine learning is necessary in these situations because the data is often very noisy, and (as with `Per1`–`3`), phenotypic changes may be invisible unless several genes are knocked out. Two recent examples of research are [79; 92]. Synthetic lethality and other perturbations are discussed in more depth in subsection 4.2.

Inferring an explanatory model of the network is often better, with more useful applications to biological understanding, genetic engineering and pharmaceutical design. Types of model and inference techniques are discussed in section 5. The distinction between network inference and epistatic analysis is frequently not made clear in the research; failing to do so makes the consequent publications very difficult to understand.

Interestingly, there has been very little work which has combined different types of models. One example is [50; 111], another is [9], it is also discussed theoretically by D'haeseleer et al. [25].

4 Available Data

To address bioinformatic network problems there are four types of data available. These are:

- Expression data
- Perturbation data
- Phylogenetic data
- Chemical and gene location data

This section describes the accessibility and utility of these for machine learning. Sometimes multiple kinds of data are used, e.g. [5; 44; 118], however this usually makes the inference more time and space complex.

4.1 Expression Data

Expression data measures how active each $n \in N$ is. As transcription activity cannot be measured directly, the concentration of mRNA (which is ephemeral) is used as a proxy.

Because regulatory or phenotypic protein interactions after transcription can consume some of the mRNA before it can regulate another gene[96] this may seem to be an inaccurate measure of gene activity[95]. Furthermore, a protein may bind to a promoter region but actually have no regulatory effect[44].

In addition, most genes are not involved in most cellular processes[54]. This means that many of the genes sampled may appear to vary randomly.

However, if the data set is comprehensive and we are just concerned with inference of the regulatory relationships these influences are not important. Sufficient data or targeted inference obviates the problem of irrelevant genes. Non-genetic, unmodeled influences are analogous to hidden intermediate variables in a *Bayesian network*[44] (BN, subsection 5.4) whose only parent is the regulatory gene and whose only child is the regulated gene. An influence like this does not distort the regulatory relationships or predictive accuracy of the model.

Tegner et al.'s [108]'s inference of a network with known post-transcription regulation and protein interactions using only (perturbed, subsection 4.2) gene expression data provides evidence of this.

Types of Expression Data

There are two kinds of expression data. They are equilibrium expression levels in a static situation and time series data that is gathered during a phenotypic process such as the cell development cycle (e.g. [104]).

Expression data is usually collected using *microarrays* or a similar technology. Time series data is gathered by using temperature- or chemical-sensitive mutants to pause the phenotypic process while a microarray is done on a sample.

A microarray is a pre-prepared slide, divided into cells. Each cell is individually coated with a chemical which fluoresces when it is mixed with the mRNA generated by just one of the genes. The brightness of each cell is used as a measurement of the level of mRNA and therefore of the gene's expression level.

Microarrays can be both *technically noisy*[59] and *biologically noisy*[84]. However, the magnitude and impact of the noise is hotly debated and dependent on the exact technology used to collect samples. Recent research[59, p. 6] (2007) argues that it has been "gravely exaggerated".

New technologies[26] are promising to deliver more and cleaner expression data in the future.

Examples of research which use equilibrium data include [2; 15; 27; 51; 57; 66; 72; 80; 95–97; 99; 104; 105; 107; 110; 111; 116; 124]. Wang et al.'s [116] work is particularly interesting as it describes how microarrays of the same gene collected in different situations can be combined into a single, larger data set.

Work that has been done using time series data includes [115], [89], [100] and [53; 102; 103; 120; 121]. Kyoda et al. [62] notes that time series data allows for more powerful and precise inference than equilibrium data, but that the data must be as noise-free as possible for the inference to be reliable.

Research on accurately simulating expression data includes [3; 7; 19; 30; 84; 102].

4.2 Perturbation Data

Perturbation data is expression data which measures what happens to the expression levels of all genes when one or more genes are artificially perturbed. Perturbation introduces a causal arrow which can lead to more efficient and accurate algorithms, e.g. [62].

Examples of experiments using perturbation data include [26; 36; 62; 108].

4.3 Phylogenetic Data

Phylogenetics is the study of species' evolutionary relationships to each other. To date, very little work has been carried out which directly uses phylogenetic conservation[17; 49] to identify regulatory relationships *de novo*. This is because phylogenetic data is not sufficiently quantified or numerous enough.

However, this sort of information can be used to validate results obtained using other methods. As [60] notes, transcriptional promoters tend to evolve phylogenetically, and as research by Pritsker et al. [90] illustrates, regulatory relationships in species of yeast are often conserved. [28] reaches similar conclusions, arguing that the "evolution of gene regulation underpins many of the differences between species".

4.4 Chemical and Gene Location Data

Along with phylogenetic data, primary chemical and gene location data can be used to validate inference from expression data or to provide an informal prior. Many types of chemical and gene location data exist; this subsection summarises some examples of recent research.

Yamanishi et al. [118] presents a technique and applies it to the yeast *Saccharomyces cerevisiae* so that the protein network could be inferred and understood in more depth than just synthetic lethality allowed. Their technique used Y2H, phylogenetics and a functional spatial model of the cell.

Hartemink et al.'s [44] work is broadly similar to Yamanishi et al.'s [118]. ChIP assays were used to identify protein interactions. In some experiments these regulatory relationships were fixed as occurring and the results compared with the unrestricted inference.

Harbison et al. [43] combined microarrays of the entire genome and phylogenetic insights from four related species of yeast (*Saccharomyces*). Given 203 known regulatory genes and their transcription factors, they were able to discover the genes that these factors acted as regulators for.

Although some research[114; 118] on general and principled techniques for using multiple types of data, most such research aims to answer a specific question about a specific, well known species, rather than to develop more general methods. General machine learning techniques which have potential include multi-classifiers[91] and fuzzy set theory[11] to maximise the information extracted.

5 Approaches to GRN Inference

Having discussed the relevant biology, the bioinformatic problems and the data that can be used to approach these problems, this section reviews different types of approach to network inference.

5.1 Clustering

Clustering[117] can reveal the modular structure[5; 50] of GRN, guide other experiments and be used to preprocess data before further inference.

This subsection discusses distance measures and clustering methods first. Then it gives a number of examples, including the use of clustering to preprocess data. Finally, it summarises *biclustering*.

Overview

A clustering algorithm is made up of two elements: the *method*, and the *distance measure*. The distance measure is how the similarity (difference) of any two data points is calculated, and the method determines how data points are grouped into clusters based on their similarity to (difference from) each other. Any distance measure can be used with any method.

Distance Measures

Readers are assumed to be familiar with basic distance measures such as the Euclidean distance. The *Manhattan distance*[61] is similar to the Euclidean distance. A gene's distance from a cluster is usually considered to be the gene's mean, maximum, median or minimum from genes in that cluster. *Mutual information* (MI), closely related to the Shannon entropy[98], is also used.

The *Mahalanobis distance*[20; 73] addresses a weakness in Euclidean distance measures. To understand the weakness it addresses it is important to distinguish between the real module or cluster underlying the gene expression, and the apparent cluster which an algorithm infers. Dennett [22] has a more in depth discussion of this distinction.

Imagine that we are using the Euclidean distance and that the samples we have of genes in the underlying "real" clusters C and D are biased samples of those clusters. Assume that the method is clustering the gene h, and that h is truly in D. However, because of the way that the samples of C and D are biased it will be clustered into C. Having been clustered into C it will also bias future genes towards C even more. Because microarrays are done on genes and phenotypic situations of known interest this bias is possible and may be common.

This bias comes about because naive distance measures do not consider the covariance or spread of the cluster. The Mahalanobis distance considers this; therefore it may be more likely to correctly cluster genes than measures which do not consider this factor.

Clustering Methods

Readers are assumed to be familiar with clustering methods and know that they can be *partitional* or *hierarchical* and *supervised* or *unsupervised*.

Many classic partitional algorithms, such as k-means[1; 71], work best on hyper-spherical clusters that are well separated. Further, the number of clusters must be specified in advance. For this reason they may not be ideal for gene expression clustering. We expect that *self-organising maps*[41, ch. 7] (SOM) would be have similar problems, despite apparent differences.

Some clustering algorithms which have been successfully used are *fuzzy clustering* algorithms. When fuzzy clustering is used a gene may be a partial member of several clusters, which is biologically accurate[6]. Fuzzy methods are also comparatively robust against noisy data.

Fuzzy (and discrete) methods that allow a gene to have total cluster membership greater than one, i.e. $\sum_j \mu_{ij} > 1$, create *covers*[72] over the data, and don't just *partition* it[25].

Similarly, clustering methods which find hypergraphs[42; 78] allow any one gene to belong to more than one cluster. In a hypergraph an edge can connect to any number of vertices and is analogous to a cover.

It is also important that the clustering algorithms are robust in the face of noise and missing data, [54; 113] discuss techniques for fuzzy and discrete methods.

Previous Clustering Research

Gene expression data clustering has been surveyed in several recent papers (e.g. [2; 25; 27; 97; 124] and others). Zhou et al. [124] compares a range of

algorithms and focuses on combining two different distance measures (e.g. mutual information and fuzzy similarity or Euclidean distance and mutual information) into one overall distance measure. Azuaje [2] describes some freely available clustering software packages and introduces the SOTA algorithm. SOTA is a hierarchical clustering algorithm which determines the number of clusters via validity threshold.

In [124], initial membership of each gene in each fuzzy cluster is randomly assigned[124] and cluster memberships are searched over using *simulated annealing*[77]. While searching, the fuzzy membership is swapped in the same way that discrete cluster memberships would be swapped.

GRAM[5] is a supervised clustering algorithm which finds a cover and is interesting because it combines protein-binding and gene expression data. It uses protein-binding information to group genes which are likely to share promoters together first, then other genes which match the initial members expression profiles closely can also be included in the cluster.

[95] describes another clustering algorithm which uses a list of candidate regulators specified in advance to cluster genes into modules. This algorithm can infer more complex (Boolean AND/OR) regulatory relationships amongst genes, and its predictions have been empirically confirmed.

Multi-stage inference ([9; 25; 111] can make principled inference over larger numbers of genes tractable. Although the underlying network is directed (as described in subsection 2.2) and may have very complex regulatory relationships these factors are conditionally independent of the graphical structure and do not need to be considered simultaneously.

Horimoto and Toh [50] also found that nearly 20% of the gene pairs in a set of 2467 genes were Pearson-correlated at a 1% significance level. This emphasises the modular nature of GRN.

Mascioli et al.'s [75] hierarchical algorithm is very interesting. The validity criterion is changed smoothly, and this means that every cluster has a *lifetime*: the magnitude of the validity criterion from the point the cluster is created to the point that it splits into sub-clusters. The dendogram is cut at multiple levels so that the longest lived clusters are the final result.

This idea is very powerful and selects the number of clusters automatically. However a cluster's lifetime may depend on samples not in the cluster, and this is not necessarily appropriate if intra-cluster similarity is more important.

Shamir and Sharan [97] suggests not clustering genes which are distant outliers and leaving them as singletons.

Selection of the right clustering algorithm remains a challenging and demanding task, dependent on the data being used and the precise nature of any future inference.

Previous Biclustering Research

Biclustering is also known as co-clustering and direct clustering. It involves grouping subsets of the genes and subsets of the samples together at the same

time. A bicluster may represent a subset of the genes which are co-regulated some of the time. Such a model generalises naturally to a cover in which each gene can be in more than one cluster. This kind of *bicovering* algorithm is described in [99] and [107].

Madeira and Oliveira's [72] article is a recent survey of the field. It tabulates and compares many biclustering algorithms.

In general, optimal biclustering is an NP-hard problem[119]. In a limited number of cases, exhaustive enumeration is possible. In other cases, heuristics such as divide-and-conquer or a greedy search may be used[72].

5.2 Logical Networks

This subsection describes research which infer Boolean or other logical networks as a representation of a GRN. Boolean networks were first described by Kauffman[55]. Prior to discussing examples of GRN inference carried out using Boolean networks we define them.

Overview

In a Boolean model of a GRN, at time t, each gene is either expressed or not. Based on a logical function over a gene's parents and their value at time t, its value can be calculated at time $t+1$. Figure 3 is an example of a Boolean network.

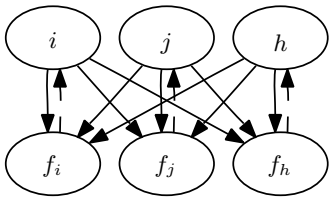


Fig. 3. A Boolean network. For clarity each $f \in F$ has been made into a node. n and n' are connected via these function nodes. Normally the functions are implicit in the edges amongst N. $f_i = (\neg i \vee i) \wedge j \wedge h$, $f_j = \neg i \vee j \wedge h$, $f_h = i \vee j \wedge h$.

Recent research[10; 12] investigates the theoretical properties of *fuzzy logic networks* (FLN) and infers biological regulatory networks using time series expression data. FLN are a fuzzy generalisation of Boolean networks.

Previous Logical Network Research

Silvescu and Honavar's [100] algorithm uses time series data to find *temporal Boolean networks* (TBoN). [67] is older and uses unaugmented Boolean

networks. TBoN were developed to model regulatory delays, which may come about due to missing intermediary genes and spatial or biochemical delays between transcription and regulation. An example of a temporal Boolean network has been presented in figure 4.

A temporal Boolean network is very similar to a normal Boolean network except that the functions $f \in F$ can refer to past gene expression levels. Rather than depending just on N_t to infer N_{t+1}, parameters to f_i can be annotated with an integer temporal delay.

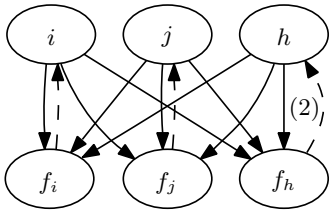


Fig. 4. A temporal Boolean network. Presentation and functions are as in figure 3, but delays are shown in brackets between genes and functions. The default delay if no annotation is present is assumed to be 0.

For example: $h' = f_h(i, j, h) = i_0 \vee j_0 \wedge h_2$ means h is expressed at $t + 1$ if either i at t or j at t is expressed, so long as h also was at time $t - 2$. TBoN can also be reformulated and inferred as decision trees.

Lähdesmäki et al. [63] considered how to take into account the often contradictory and inconsistent results which are obtained from microarray data. The aim of the approach is to find not just one function f_i but a set of functions F_i for each gene i. Each member of F_i may predict the wrong value for i based on the values of N. In those situations though other $f \in F_i$ may predict correctly.

They developed a methodology which would find all functions which made less than ϵ errors on the time series training data for 799 genes. The regulatory functions of only five genes were identified because the search for all consistent or all *best-fit* functions could not be done efficiently.

Boolean networks have a number of disadvantages. Compared to the underlying biology, they create such a simple model that it can only give a broad overview of the regulatory network. In addition, despite a simple model, the algorithms are usually intractable. Typically, they are polynomial or worse in N and exponential in k_{in}^{max}. Furthermore, as k_{in}^{max} increases you need greater quantities of data to avoid overfitting.

However, the simplicity of the functional representation is a strength as well. The very low fidelity means that the models are more robust in the face of noisy data. Attractor basin analysis of Boolean networks can help provide a

better understanding of the stability and causes of equilibrium gene expression levels. Such equilibria are often representative of particular phenotypes[19].

5.3 Differential Equations and Other Mathematical Formalisms

Approaches based on differential equations predict very detailed regulatory functions. This is a strength because the resulting model is more complete. It is also a weakness because it increases the complexity of the inference and there may not be enough data to reliably infer such a detailed model. In addition, the existence of a link and the precise nature of the regulatory function are two inferential steps and the regulatory function can be easily inferred given knowledge of the link.

Because there are so many different ways of doing this kind of high-fidelity inference, this subsection just presents a number of examples, as in subsection 4.4.

The NIR (*N*etwork *I*nference via multiple *R*egression) algorithm is summarised in [26; 36]. It uses gene perturbations to infer ordinary differential equations (ODEs). The method has been applied to networks containing approximately 20 genes.

Kyoda et al. [62] uses perturbations of equilibrium data and a modified version of the Floyd-Warshall algorithm[29] to infer the most parsimonious ODE model for each gene. Although noise and redundancy may make this incorrect it is arguably the most correct network which can be inferred with the training data. Different inferred networks can fit the data equally well, because GRN are cyclic[62].

Kyoda et al.'s method is very efficient ($O(N^3)$) and is not bound by arbitrary k_{in}^{max}. This is a consequence of the fact that it uses perturbation data, which is much more informative than expression data.

Toh and Horimoto [111] used *graphical Gaussian models* (GGM) to find conditional dependencies amongst gene clusters[50]. Information on the regulatory direction from primary literature was used to manually annotate the raw, undirected model.

5.4 Bayesian Networks

This subsection describes and defines Bayesian networks, how they can be learnt and previous research which used them. Readers are referred to [46] for a more detailed introduction.

Bayesian Networks Described and Defined

A Bayesian network is a graphical decomposition of a joint probability distribution, such as the distribution over the state of all genes in a GRN. There is one variable for each gene.

The lack of an edge between two genes i and j means that, given i's or j's *Markov blanket*, i and j are independent: $p(i|j, mb(i)) = p(i|mb(i))$. Loosely and intuitively, the presence of an edge between i and j means that they are "directly" (causally?[88]) dependent on each other. The Markov blanket[70] consists of a variable's parents, children and children's parents as defined by the edges in and out of the variables. See figure 5 for an example.

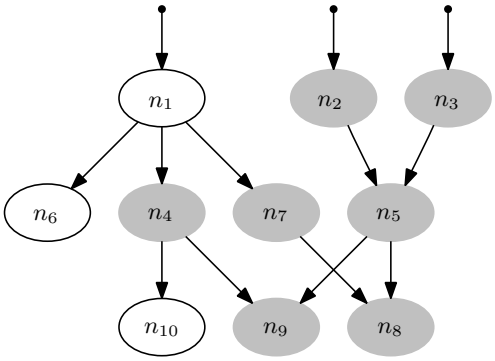


Fig. 5. A Bayesian network and Markov blanket. Genes in the Markov blanket of n_5 are shown with a grey background. Priors for $n_{1..3}$ are denoted by incoming parentless edges.

BN must be acyclic[26]. This is a problem because auto-regulation and feedback circuits are common in GRN[109]. The reason why BN must be acyclic is that a cyclic BN cannot be factorised.

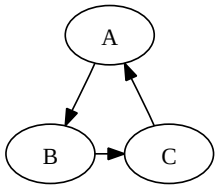


Fig. 6. A cyclic Bayesian network. Impossible to factorise.

Consider the BN shown in figure 6. The value of A depends on the value of B, the value of B depends on the value of C, and the value of C depends on the value of A. Equation 1 shows what happens when we try to factorise the joint distribution by expanding the parents (pa).

$$\begin{aligned} p(A,B,C) &= p(A|pa(A))p(pa(A)) \\ &= p(A|B)p(B|pa(B))p(pa(B)) \\ &= p(A|B)p(B|C)p(C|pa(C))p(pa(C)) \\ &= p(A|B)p(B|C)p(C|A)p(A|pa(A))p(pa(A)) \\ &\qquad\qquad \text{And so on...} \end{aligned} \tag{1}$$

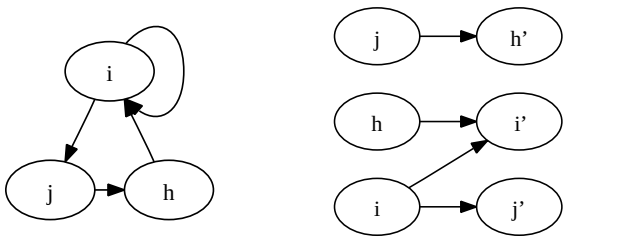
A BN is also a subtly different kind of model than a Boolean network or set of ODEs. While the latter create a definite, possibly incorrect model of the regulatory relationships, a BN model is strictly probabilistic, although it may incorrectly describe some genes as directly dependent or independent when they are not and long run frequency evaluations may suggest that it has incorrect conditional distributions for some genes.

In summary, BN are attractive (their statistical nature allows limited causal inference[88] and is robust in the face of noise and missing data) and unattractive (acyclic only). *Dynamic Bayesian networks*[34; 82] (DBN) are an elegant solution to this problem.

Dynamic Bayesian Networks

A DBN is a Bayesian network which has been temporally "unrolled". Typically we view variables as entities whose value changes over time. If we view them as constant, as they are in HMM, then we would represent i at t and i at $t+1$ with two different variables, say i_t and i_{t+1}.

If we assume that conditional dependencies cannot point backwards or "sideways" in time this means that the graph must be acyclic, even if i auto-regulates. If we also assume that the conditional dependencies are constant over time and that the prior joint distribution[34] is the same as the temporal joint distribution then the network only needs to be unrolled for one time step. A visual illustration of this is provided in figure 7.



(a) A cyclic BN (b) An equivalent, acyclic, DBN

Fig. 7. A cyclic BN and an equivalent, acyclic, DBN. The prior network[34] is not shown in this diagram.

Modern variations of BN have also added new capabilities to them, particularly fuzzy Bayesian networks. These range from specialised techniques designed to reduce the complexity of *hybrid Bayesian network* (HBN) belief

propagation with fuzzy approximations[4; 47; 85; 86] to more general formalisations which allow variables in Bayesian networks to take fuzzy states, with all of the advantages in robustness, comprehensibility and dimensionality reduction[32; 87] this provides.

Learning Bayesian Networks

The problem of learning a Bayesian network can be divided into two subproblems. The simpler problem is learning θ, the conditional distributions of the BN given its edges, η. This can be done with either full training data or training data which is partially covered.

The second and more difficult problem is inference of η and θ simultaneously. This can also be done with either full or incomplete training data.

Bayesian network inference is a large research field, and comprehensively summarising it here is impossible. For such a summary we refer interested readers to [46; 81] and [34; 40]. This subsubsection focuses on just a few algorithms. It considers the case of θ inference first, before showing how many of the same algorithms can be applied to structural inference.

θ Inference

The simplest way of performing θ inference is just to count up and categorise the examples. This is statistically valid in the case of complete data. The result of such a count is a *maximum likelihood* (ML) estimate. The desired result is the *maximum a posteriori* (MAP), which incorporates any prior information. When the prior is uniform then ML = MAP. A uniform prior is common as it also maximises the informativeness of the data.

To avoid certainty in the conditional probability distributions, *pseudocounts* are often used. Pseudocounts were invented by Laplace[65] for the sunrise problem[1] and they can be considered an ad hoc adjustment of the prior distribution. Pseudocounts are invented data values, normally 1 for each entry in each conditional distribution, and their presence ensures probabilities never reach 0 or 1. This is important because $p(i|\cdot) = 0$ or $p(i|\cdot) = 1$ implies certainty, which is inferentially invalid with finite data.

Although pseudocounts are invalid if there is missing data, we speculate that if the available data is nearly complete then using pseudocounts could be accurate enough or a good starting point to search from.

If the data is too incomplete for counts to be used then there is a range of search algorithms which can be used. These include greedy hill climbing with random restarts[93], the EM algorithm[21], *simulated annealing*[77] and *Markov Chain Monte Carlo*[45] (MCMC).

The independence and decomposability of the conditional distributions is a crucial element in the efficiency of the algorithm, as it makes calculation of the likelihood much faster.

[1] *Viz:* What is the probability that the sun will rise tomorrow?

The EM Algorithm

The expectation maximisation (EM) algorithm[21] is shown in algorithm figure 1. A key advantage of EM is that it robust and tractably handles missing (covered) values in the training data. From an initial guess θ_0 and the observed gene expression levels we can calculate N_i, the expected gene expression levels. Holding this expectation constant and assuming that it is correct, θ_{i+1} is set so that N_i is the maximum likelihood gene expression levels.

This process is repeated and θ will converge on a local maxima. Random restarts or using simulated annealing (described next) to determine θ_{i+1} means that the algorithm can also find θ_{ML}.

Algorithm 1: The EM algorithm[21]. p_N is a function that returns the expected expression level of all genes, N_i, given the current parameters θ and the observed training data $N \times M$. ML_N is a function that returns the θ which maximises the likelihood of some state of the genes, N. θ_{best} is the best set of parameters the search has found. If simulated annealing or random restarts are used this will be θ_{ML}.

Input:
$N \times M$, the data to use in the inference
η, the edges of the graph G
Output:
$G = \langle \eta, \theta_{best} \rangle$, the maximum likelihood BN given η.
begin
  $\theta_0 \longleftarrow$ initial guess, e.g. based on counts from noisy data
  repeat
    $N_{i+1} \longleftarrow p_N(\theta_i, N \times M)$
    $\theta_{i+1} \longleftarrow ML_N(N_{i+1})$
  until $p(\theta_i) = p(\theta_{i-1})$
  return θ_{best}
end

Simulated Annealing

Simulated annealing [77] is a generalised Monte Carlo method which was inspired by the process of annealing metal. A Monte Carlo method is an iterative algorithm which is non-deterministic in its iterations. Metal is annealed by heating it to a very high temperature and then slowly cooling it. The resulting metal has a maximally strong structure.

Simulated annealing (SA) is very similar to hill climbing, except that the distance and direction to travel in (Δ and $grad(\theta)$ in hill climbing) are sampled from a probability distribution for each transition. Transitions to better states are always accepted, whilst transitions to worse states are accepted with a probability p, which is lower for transitions to much worse states. This

probability decreases from transition to transition until only transitions to better states are accepted.

In this way simulated annealing is likely to explore a much wider part of the search space at the early stage of the search, but eventually it optimises greedily as hill climbing does[31]. If proposed transitions are compared to current positions based on their likelihood, simulated annealing finds θ_{ML}. The MAP can be found by including a prior in the calculations.

η Inference
The three search algorithms just discussed are naturally applicable to θ inference, but each of them can be used for η inference as well. For example, the EM algorithm has been generalised by Friedman *et al.*[33; 34]. *Structural EM* (SEM) is very similar to EM. The main difference is during the ML step, when SEM updates *theta* and also uses it to search the η-space.

Integrating the Posterior
One common weakness of these algorithms is that they all find a single solution. A better solution is to integrate over the posterior distribution[44]. This is because the MAP solution may be only one of several equi-probable solutions. Ignoring these other solutions when calculating a result means that there is a greater chance it will be in error.

Analytically integrating the posterior distribution is usually impossible. Numerical integration by averaging many samples from the posterior is an alternative. MCMC algorithms[69] are the most common way of drawing samples from the posterior.

Markov Chain Monte Carlo Algorithms
Metropolis-Hastings[45] and Gibbs sampling[37] are commonly used MCMC algorithms. Other more intricate algorithms which explore the solution space more completely (such as Hybrid Monte Carlo[83]) have also been developed.

An MCMC algorithm is very similar to simulated annealing. Starting from an initial state it probabilistically transitions through a solution space, always accepting transitions to better solutions and accepting transitions to worse states with lower probability, as in simulated annealing. Transitions in MCMC must have the *Markov property*. The probability of a transition from γ_t to γ_{t+1} which has the Markov property is independent of everything except γ_t and γ_{t+1}, including all previous states.

Higher order Markov chains can also be defined. An s'th order Markov chain is independent of all states before γ_{t-s}. Markov chains are a type of Bayesian network and are very similar to dynamic Bayesian networks.

Because of the wide range of MCMC algorithms it is difficult to give an informative list of invariant properties. We use the Metropolis-Hastings algorithm as an example instead.

Metropolis-Hastings MCMC
Assume we have a solution γ_t and that we can draw another solution con-

ditional on it, γ_{t+1}, using a proposal distribution q. The normal distribution $N(\theta, \sigma^2)$ is frequently used as the proposal distribution q. Assume also that we can calculate the posterior probability of any solution γ (this is usually much easier than drawing samples from the posterior). The proposed transition to γ_{t+1} is accepted if and only if the *acceptance function* in equation 2 is true.

$$u < \frac{p(\gamma_{t+1})q(\gamma_t|\gamma_{t+1})}{p(\gamma_t)q(\gamma_{t+1}|\gamma_t)}, \text{ where } u \sim U(0,1) \tag{2}$$

Metropolis-Hastings (and Gibbs sampling) converge quickest to the posterior when there are no extreme probabilities in the conditional distributions[40].

Because the probability of a transition being accepted is proportional to the posterior probability of the destination, the sequence of states will converge to the posterior distribution over time. MCMC can be used to take samples from the posterior, by giving it time to converge and by leaving a large enough number of proposed transitions between successive samples to ensure that $\Gamma_t \perp\!\!\!\perp \Gamma_{t+1}$.

The number of transitions needed to converge depends on the cragginess and dimensionality of the search space. Considering 10^5 proposed transitions is usually sufficient, and 10^4 proposed transitions between samples usually means that they are independent. Convergence and independence can be checked by re-running the MCMC algorithm. If the results are significantly different from run-to-run it indicates that one or both of the conditions was not met.

Because the number of samples that are necessary is constant as the dimensionality of the problem grows[70], MCMC are somewhat protected from the *curse of dimensionality*[23; 26].

Scoring Bayesian Networks

An important part of structural inference is comparing two models. The simplest scoring measure is the marginalised likelihood of the data, given the graph: $p(N \times M|G')p(G')$. This scoring measure is decomposable, as shown in equation 3. The log-sum is often used for pragmatic reasons. In this subsection, G refers to the set of all graphs and $G' = \langle \eta', \theta' \rangle \in G$ refers to one particular graph. η, η', θ and θ' are defined similarly.

$$p(N \times M|G') = \prod_{m \in M} \prod_{n \in N} p(n \times m|m, G') \tag{3}$$

However, the marginalised likelihood may overfit the data. This is because any edge which improves the fit will be added, and so the measure tends to make the graph too dense. A complexity penalty can be introduced if graphs are scored by their posterior probability, the *Bayesian Scoring Metric*[120] (BSM). For multinomial BN this scoring measure is commonly referred to as the BDe (*Bayesian Dirichlet equivalent*).

$$\begin{aligned} BSM(G', N \times M) &= \log p(G'|N \times M) \\ &= \log p(N \times M|G') + \log p(G') - \log p(N \times M) \end{aligned} \tag{4}$$

With the BSM, overly complex graphs can be penalised through the prior, or one can just rely on the fact that more complex graphs have more free parameters in θ. Because $p(\sum_{\theta|\eta} \theta) = 1$, the probability of any particular θ given a complex η will be relatively less. Loosely, this is because the probability must be "spread out" over a larger space.

The probability of the data over all possible graphs — $p(N \times M)$, expanded in equation 5 — is difficult to calculate because there are so many graphs. MCMC methods are often used to calculate it.

$$\begin{aligned} p(N \times M) &= \sum_{G' \in G} p(N \times M, G') \\ &= \sum_{G' \in G} p(N \times M|G')p(G') \\ &= \sum_{\eta' \in \eta} \sum_{\theta' \in \theta} p(N \times M|\theta', \eta')p(\theta'|\eta') \end{aligned} \tag{5}$$

When MCMC is too time consuming the posterior can be approximated using the *Bayesian Information Criterion*[94] (BIC, equation 6, where $|\theta_{ML}|$ is the number of parameters in the ML θ for η). This is an asymptotic approximation to the BDe and it is faster to calculate[120]. However, the BIC over penalises complex graphs when the training data is limited. Training data for GRN inference is typically very limited.

$$\log p(N \times M|\eta') \approx BIC(N \times M, G) = \log p(N \times M|\eta', \theta') - \frac{|\theta'|}{2} \log N \tag{6}$$

Other measures include the *minimum description length*[39; 64], the *Local Criterion*[46], the *Bayesian Nonparametric heteroscedastic Regression Criteria* (BNRC)[51; 52] and applications of computational learning theory[16]. Each of these tries to balance a better fitting graph with complexity controls.

Previous Bayesian Network Research

The project at Duke University[53; 102; 103; 120; 121] aimed to understand songbird singing. It integrates simulated neural activity data with simulated gene expression data and infers DBN models of the GRN.

Between 40–90% of the genes simulated were "distractors" and varied their expression levels according to a normal distribution. The number of genes

simulated differed from experiment to experiment and was in the range 20–100 ([121] and [102], respectively). It was claimed that the simulated data showed realistic regulatory time lags, and also between gene expression and neural activity[102].

Singing/not-singing and gene expression levels were updated once each theoretical minute, and the simulated data was sampled once every 5 minutes. It takes approximately 5 minutes for a gene to be transcribed, for the mRNA to be translated into a protein and for the protein to get back to the nucleus to regulate other genes[102].

The simulated data was continuous and a range of normalised hard and fuzzy discretisations were trialled[102; 121]. Interestingly, and contrary to information theoretic expectations[51; 101], the hard discretisations which discarded more information did better than the fuzzy ones. Linearly interpolating 5 data points between each pair of samples gave better recovery and fewer false positives[121].

[121] also developed the *influence score*, which makes the joint distribution easier to understand. If i regulates j then $-1 < I_{ij} < 1$, where the sign indicates down or up regulation and the magnitude indicates the regulatory strength.

The techniques developed could reliably infer regulatory cycles and cascade motifs. However, convergent motifs and multiple parents for a single gene were only reliably identified with more than 5000 data points. [40] and [103] discuss topological factors in more detail.

Nir Friedman and others[34; 51; 82] have also used DBNs for GRN inference. Friedman et al. [34] has extended the BIC and BDe to score graphs in the case of complete data. They have also extended SEM to incomplete data.

Friedman and others have proposed a *sparse candidate* algorithm which is optimised for problems with either a lot of data or a lot of variables. It uses MI to select candidate regulators for each gene and then only searches over networks whose regulators for each gene i come from $cand_i$. This process is iterated until convergence.

Murphy and Mian[82] discuss DBNs with continuous conditional distributions. So far our discussion has only considered multinomial BN. The value of a sample from a continuous θ_i is calculated by drawing a sample from it or by integrating over it. Continuous representations maximise the amount of information that can be extracted from the data, although they are also vulnerable to noise.

The tractability of different kinds of BN is contentious. Jarvis et al. [53, p974] claim that continuous BNs are intractable. Murphy and Mian [82, 5.1] note that exact inference in densely connected discrete BN is intractable and must be approximated. Others focus on the complexity of HBN[4; 85]. In general, Bayesian inference is NP-complete[14; 108].

There is no consensus on the most appropriate search algorithm. Hartemink et al. [44] concluded that simulated annealing was better than both greedy hill climbing and Metropolis-Hastings MCMC. Yu et al. [121] argued that greedy

hill climbing with restarts was the most efficient and effective search method. Imoto et al. [51], used non-parametric regression. Because each algorithm is better for subtly different problems this variation is unsurprising, and [40, sections 3.2.1, 4.2.2] has suggestions on selecting an algorithm.

6 Statistical and Computational Considerations

This section discusses the problems of tractability (subsection 6.1) and a particular statistical problem with GRN inference from microarrays (subsection 6.2).

6.1 Efficiency and Tractability

Almost all of the algorithms described above must limit N, the number of genes, and k_{in}^{max}. Unbounded k_{in}^{max}-network inference is almost always[2] $O(N^k) = O(N^N)$ or worse. Bayesian network inference is NP-hard[14; 108]; DBN inference is even harder[82].

The magnitude of this problem is clear when we consider the number of genes in, e.g., *S. cerevisiae* (approximately 6,265) and compare it to the size of the inferred networks as research has progressed. See table 1 for examples.

Table 1. GRN algorithmic efficiency against the number of genes N. Most of these results also require $k_{in}^{max} \leq 3$. [63] found all explanatory Boolean functions with $\epsilon \lesssim 5$ for the genes it solved for.

Research	$max(N)$
[67] (1998)	50
[74] (1998)	Unspecifiedly "small"
[115] (2001)	≈ 100
[108] (2003)	$\approx$ 10–40
[63] (2003)	5
[102; 121] (2002–2004)	$\approx$ 20–100
[8] (2004)	100, also $\overline{k_{in}} = 10$
[26] (2006)	≈ 20

Three exceptions to this trend are informative. [50; 111] and [9] used clustering to reduce the dimensionality first. However, no "de-clustering" was carried out on the inferred cluster-networks.

Kyoda et al. [62] uses perturbation data and creates a polynomial-time algorithm which is independent of k_{in}^{max}. Bernardo et al.'s [8] work uses a

[2] [62] was better, but it used an interactive style of learning that is not practical with current biochemical technology.

similar approach. This research shows how valuable it is to use all information in the data, however current biotechnology does not allow for the data sets used in this research to be realistically created.

6.2 Microarrays and Inference

Chu et al. [15] identifies a statistical problem with inference from microarrays. The problem is as follows:

- Expression levels obtained from microarrays are the summed expression levels of $10^3 < x < 10^6$ cells.
- Except in limited circumstances, summed conditional dependencies may be different from individual conditional dependencies.

When the regulatory graph is singly connected (i.e. $i \rightarrow j \rightarrow h$, but not $i \rightarrow j \leftarrow h$), or if the noise is Gaussian and the regulatory relationships are all linear, then the factorisation of the sum is identical with the factorisation of the summands.

As neither of these conditions hold in GRN, the authors of [15] are concerned with the apparently successful results of machine learning using microarrays.

However, since the article was published substantially many more results have been biologically verified. This does not indicate that Chu et al. are wrong, but it does suggest that the conditional dependencies of the sum and the summands are similar enough in GRN. Further, it is important to remember that noise also blurs true conditional dependencies and that machine learning has been relatively successful anyway.

7 A Rough Map of the Field

Table 2 visually categorises some GRN research. It excludes results which just cluster the genes into modules and do not predict any regulatory relationships.

8 Conclusion and Directions

Research in GRN inference spans an enormous range of techniques, fields, and technologies. Techniques from other fields are used in new ways and new technology is being continuously developed. Nonetheless the problems of network inference remains open, and machine learning is essential. Fruitful avenues of research include:

- Incorporating cluster information into more detailed GRN inference.
- Combining separately learnt networks. Bayesian networks seem to be an ideal representation to use in this case. Considerations include:

Table 2. A visual categorisation of GRN research. Columns denote kinds of data (sometimes simulated) which can be used and rows denote types of model. The GRN inference in [90] was secondary, and [79; 92] discuss epistatic inference. [96] also cites some work which uses phylogenetic data and ChIP assays. [44] and [43] used equilibrium microarray data as well. [13] used both equilibrium and microarray data.

	ODE etc.	Boolean	BN	Neural
Time series	[24; 89; 116]	[63; 67; 100]	[35; 53; 82; 123]	[115]
Equilib.	[13; 95; 111]		[51]	
Perturb.	[8; 26; 36; 62; 108]	[122]		[79; 92]
Phylo.	[90]			
Chem./Loc.	[43]		[44]	

  - How are two models which disagree about the edges, regulatory functions or both combined?
  - How does model agreement affect the posterior distribution?
  - What does it mean if two models agree but a third disagrees?
  - Can models inferred with different data sets be combined?

  In related research, machine inferred networks have been compared with networks manually assembled from the primary literature [48].
- Algorithmically increasing the value of different data types, as in [62]. Multi-classifiers[91] may also be fruitful.
- Incorporation of fuzzy techniques and clustering to address noise and the curse of dimensionality[32].

References

[1] D. Arthur, S. Vassilvitskii. k-means++: The advantages of careful seeding. Technical Report 2006-13, Stanford University, 2006.
[2] F. Azuaje. Clustering-based approaches to discovering and visualing microarray data patterns. *Brief. in Bioinf.*, 4(1):31–42, Mar. 2003.
[3] Y. Balagurunathan et al. Noise factor analysis for cDNA microarrays. *J. Biomed. Optics*, 9(4):663–678, Jul./Aug. 2004.
[4] J. F. Baldwin, E. Di Tomaso. Inference and learning in fuzzy Bayesian networks. In *FUZZ'03: The 12th IEEE Int'l Conf. on Fuzzy Sys.*, volume 1, pages 630–635, May 2003.
[5] Z. Bar-Joseph et al. Computational discovery of gene modules and regulatory networks. *Nat. Biotech.*, 21(11):1337–1342, Nov. 2003.
[6] A.-L. Barabasi, Z. N. Oltvai. Network biology: Understanding the cell's functional organisation. *Nat. Rev. Genetics*, 5(2):101–113, Feb. 2004.
[7] A. Ben-Dor et al. Clustering gene expression patterns. *J. Comp. Bio.*, 6(3/4):281–297, 1999.

[8] D. Di Bernardo et al. Robust identification of large genetic networks. *Pacific Symp. on Biocomp.*, pages 486–497, 2004.
[9] R. Bonneau et al. The inferelator: An algorithm for learning parsimonious regulatory networks from systems-biology data sets *de novo*. *Genome Bio.*, 7(R36), 2006.
[10] Y. Cao et al. Reverse engineering of NK boolean network and its extensions — fuzzy logic network (FLN). *New Mathematics and Natural Computation*, 3(1):68–87, 2007.
[11] Y. Cao. *Fuzzy Logic Network Theory with Applications to Gene Regulatory Sys.*. PhD thesis, Department of Electrical and Computer Engineering, Duke University, 2006.
[12] Y. Cao et al. S. pombe gene regulatory network inference using the fuzzy logic network. *New Mathematics and Natural Computation*
[13] Zeke S. H. Chan et al. Bayesian learning of sparse gene regulatory networks. *Biosystems*, 87(5):299–306, 2007.
[14] D. M. Chickering. Learning Bayesian networks is NP-Complete. In D. Fisher and H. J. Lenz, eds, *Learning from Data: Artificial Intelligence and Statistics V*, pages 121–130. Springer-Verlag, 1996.
[15] T. Chu et al. A statistical problem for inference to regulatory structure from associations of gene expression measurements with microarrays. *Bioinf.*, 19(9):1147–1152, 2003.
[16] I. Cohen et al. Learning Bayesian network classifiers for facial expression recognition using both labeled and unlabeled data. *cvpr*, 01:595–601, 2003.
[17] G. C. Conant, A. Wagner. Convergent evolution of gene circuits. *Nat. Genetics*, 34(3):264–266, 2003.
[18] Q. Cui et al. Characterizing the dynamic connectivity between genes by variable parameter regression and kalman filtering based on temporal gene expression data. *Bioinf.*, 21(8):1538–1541, 2005.
[19] H. de Jong. Modeling and simulation of genetic regulatory systems: A literature review. *J. Comp. Bio.*, 9(1):67–103, 2002.
[20] A. R. de Leon, K. C. Carriere. A generalized Mahalanobis distance for mixed data. *J. Multivariate Analysis*, 92(1):174–185, Jan. 2005.
[21] A. P. Dempster et al. Maximum likelihood from incomplete data via the EM algorithm. *J. the Royal Statistical Society. Series B (Methodological)*, 39(1):1–38, 1977.
[22] D. C. Dennett. Real patterns. *J. Philosophy*, 88:27–51, 1991.
[23] P. D'haeseleer. *Resconstructing Gene Networks from Large Scale Gene Expression Data*. PhD thesis, University of New Mexico, Albuquerque, New Mexico, Dec. 2000.
[24] P. D'haeseleer, S. Fuhrman. Gene network inference using a linear, additive regulation model. Submitted to Bioinf., 1999.
[25] P. D'haeseleer et al. Genetic network inference: From co-expression clustering to reverse engineering. *Bioinf.*, 18(8):707–726, 2000.

[26] M. E. Driscoll, T. S. Gardner. Identification and control of gene networks in living organisms via supervised and unsupervised learning. *J. Process Control*, 16(3):303–311, Mar. 2006.

[27] M. B. Eisen et al. Cluster analysis and display of genome-wide expression patterns. *Proc. of the National Academy of Sciences USA*, 95(25): 14863–14868, Dec. 1998.

[28] P. C. FitzGerald et al. Comparative genomics of drosophila and human core promoters. *Genome Bio.*, 7:R53+, Jul. 2006.

[29] R. W. Floyd. Algorithm 97: Shortest path. *Communications of the ACM*, 5(6):345, 1962.

[30] C. Fogelberg, V. Palade. GreenSim: A genetic regulatory network simulator. Technical Report PRG-RR-08-07, Computing Laboratory, Oxford University, Wolfson Building, Parks Road, Oxford, OX1-3QD, May 2008.

[31] C. Fogelberg, M. Zhang. Linear genetic programming for multi-class object classification. In S. Zhang and R. Jarvis, eds, *AI 2005: Advances in Artificial Intelligence: Proc. of the 18th Australian Joint Conf. on Artificial Intelligence, LNCS 3809/LNAI 3809*, pages 369–379, Sydney, Australia, Dec. 2005. Springer Verlag. ISBN 3-540-30462-2.

[32] C. Fogelberg et al. Belief propagation in fuzzy bayesian networks. In I. Hatzilygeroudis, ed, *1st Int'l Workshop on Combinations of Intelligent Methods and Applications(CIMA) at ECAI'08*, University of Patras, Greece, 21–22 Jul. 2008.

[33] N. Friedman. Learning belief networks in the presence of missing values and hidden variables. In *Proc. of the 14th Int'l Conf. on Machine Learning*, pages 125–133. Morgan Kaufmann, 1997.

[34] N. Friedman et al. Learning the structure of dynamic probabilistic networks. In *Proc. of the 14th Annual Conf. on Uncertainty in Artificial Intelligence (UAI-98)*, volume 14, pages 139–147, San Francisco, CA, 1998. Morgan Kaufmann.

[35] N. Friedman et al. Using Bayesian networks to analyze expression data. *J. Comp. Bio.*, 7(3):601–620, 2000.

[36] T. S. Gardner et al. Inferring microbial genetic networks. *ASM News*, 70(3):121–126, 2004.

[37] S. Geman, D. Geman. Stochastic relaxation, Gibbs distributions and the Bayesian restoration of images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 6:721–742, 1984.

[38] G. Giaever et al. Functional profiling of the Saccharomyces cerevisiae genome. *Nat.*, 418(6896):387–91, 2002.

[39] P. Grünwald. The minimum description length principle and non-deductive inference. In P. Flach, ed, *Proc. of the IJCAI Workshop on Abduction and Induction in AI, Japan*, 1997.

[40] H. Guo, W. Hsu. A survey of algorithms for real-time Bayesian network inference. In *Joint AAAI-02/KDD-02/UAI-02 workshop on Real-Time Decision Support and Diagnosis Sys.*, 2002.

[41] K. Gurney. *An Introduction to Neural Networks*. Taylor & Francis, Inc., Bristol, PA, USA, 1997. ISBN 1857286731.

[42] E.-H. Han et al. Clustering based on association rule hypergraphs. In *Research Issues on Data Mining and Knowledge Discovery*, page TODO, 1997.

[43] C. T. Harbison et al. Transcriptional regulatory code of a eukaryotic genome. *Nat.*, 431(7004):99–104, 2004.

[44] A. J. Hartemink et al. Combining location and expression data for principled discovery of genetic regulatory network models. *Pacific Symp. on Biocomp.*, pages 437–449, 2002.

[45] W. K. Hastings. Monte Carlo sampling methods using Markov chains and their applications. *Biometrika*, 57(1):97–109, 1970.

[46] D. Heckerman. A tutorial on learning with Bayesian networks. Technical report, Microsoft Research, Redmond, Washington, 1995.

[47] X.-C. Heng, Zheng Qin. Fpbn: A new formalism for evaluating hybrid Bayesian networks using fuzzy sets and partial least-squares. In D.-S. Huang et al., eds, *ICIC (2)*, volume 3645 of *Lecture Notes in Computer Science*, pages 209–217, Hefei, China, Aug. 23–26 2005. Springer. ISBN 3-540-28227-0.

[48] M. J. Herrgard et al. Reconciling gene expression data with known genome-scale regulatory network structures. *Genome Research*, 13(11): 2423–2434, 2003.

[49] V. F. Hinman et al. Developmental gene regulatory network architecture across 500 million years of echinoderm evolution. *Proc. of the National Academcy of Sciences, USA*, 100(23):13356–13361, Nov. 2003.

[50] K. Horimoto, H. Toh. Statistical estimation of cluster boundaries in gene expression profile data. *Bioinf.*, 17(12):1143–1151, 2001.

[51] S. Imoto et al. Estimation of genetic networks and functional structures between genes by using Bayesian networks and nonparametric regression. In *Pacific Symp. on Biocomp.*, volume 7, pages 175–186, 2002.

[52] S. Imoto et al. Bayesian network and nonparametric heteroscedastic regression for nonlinear modeling of genetic network. *J. Bioinf. and Comp. Bio.*, 1(2):231–252, 2003.

[53] E. D. Jarvis et al. A framework for integrating the songbird brain. *J. Comp. Physiology A*, 188:961–80, Dec. 2002.

[54] D. Jiang et al. Cluster analysis for gene expression data: A survey. *IEEE Transactions on Knowledge and Data Engineering.*, 16(11):1370–1386, 2004.

[55] S. A. Kauffman. *The Origins of Order: Self-Organization and Selection in Evolution*. Oxford University Press, 1993.

[56] S. A. Kauffman. Antichaos and adaptation. *Scientific American*, 265 (2):78–84, Aug. 1991.

[57] S. Kim et al. Dynamic Bayesian network and nonparametric regression for nonlinear modeling of gene networks from time series gene expression data. *Biosys.*, 75(1-3):57–65, Jul. 2004.

[58] H. Kitano. Computational systems biology. *Nat.*, 420(6912):206–210, Nov. 2002.

[59] L, Klebanov, A. Yakovlev. How high is the level of technical noise in microarray data? *Bio. Direct*, 2:9+, Apr. 2007.

[60] M. A. Koch et al. Comparative genomics and regulatory evolution: conservation and function of the chs and apetala3 promoters. *Mol. Bio. and Evolution*, 18(10):1882–1891, Oct. 2001.

[61] E. F. Krause. *Taxicab Geometry*. Dover Publications, 1987.

[62] K. M. Kyoda et al. A gene network inference method from continuous-value gene expression data of wild-type and mutants. *Genome Informatics*, 11:196–204, 2000.

[63] H. Lähdesmäki et al. On learning gene regulatory networks under the Boolean network model. *Machine Learning*, 52(1–2):147–167, 2003.

[64] W. Lam, F. Bacchus. Learning Bayesian belief networks: An approach based on the MDL principle. In *Comp. Intelligence*, volume 10, pages 269–293, 1994.

[65] P.-S. Laplace. *Essai philosophique sur les probabilits*. Mme. Ve. Courcier, 1814.

[66] P. P. Le et al. Using prior knowledge to improve genetic network reconstruction from microarray data. *In Silico Bio.*, 4, 2004.

[67] S. Liang et al. REVEAL: a general reverse enginerring algorithm for inference of genetic network architectures. In *Pacific Symp. on Biocomp.*, pages 18–29, 1998.

[68] P. Y. Lum et al. Discovering modes of action for therapeutic compounds using a genome-wide screen of yeast heterozygotes. *Cell*, 116(1):121–137, Jan. 2004.

[69] D. J. C. MacKay. Introduction to Monte Carlo methods. In M. I. Jordan, ed, *Learning in Graphical Models*, NATO Science Series, pages 175–204. Kluwer, 1998.

[70] D. J. C. MacKay. *Information Theory, Inference, and Learning Algorithms*. Cambridge University Press, 2003.

[71] J. B. MacQueen. Some methods for classification and analysis of multivariate observations. In *Proc. of the 5th Berkeley Symp. on Mathematical Statistics and Probability*, pages 281–297. University of California Press, 1967.

[72] S. C. Madeira, A. L. Oliveira. Biclustering algorithms for biological data analysis: a survey. *IEEE/ACM Transactions on Comp. Bio. and Bioinf.*, 1(1):24–45, 2004.

[73] P. C. Mahalanobis. On the generalised distance in statistics. In *Proc. of the National Institute of Science of India 12*, pages 49–55, 1936.

[74] G. Marnellos and E. Mjolsness. A gene network approach to modeling early neurogenesis in drosophila. In *Pacific Symp. on Biocomp.*, volume 3, pages 30–41, 1998.

[75] F. Massimo Frattale Mascioli et al. Scale-based approach to hierarchical fuzzy clustering. *Signal Processing*, 80(6):1001–1016, 2000.

[76] D. C. McShan et al. Symbolic inference of xenobiotic metabolism. In R. B. Altman et al., eds, *Pacific Symp. on Biocomp.*, pages 545–556. World Scientific, 2004. ISBN 981-238-598-3.

[77] N. A. Metropolis et al. Equation of state calculations by fast computing machines. *J. Chemical Physics*, 21:1087–1092, 1956.

[78] E. Mjolsness et al. Multi-parent clustering algorithms from stochastic grammar data models. Technical Report JPL-ICTR-99-5, JPL, 1999.

[79] A. A. Motsinger et al. GPNN: Power studies and applications of a neural network method for detecting gene-gene interactions in studies of human disease. *BMC Bioinf.*, 7:39, 2006.

[80] T. M. Murali, S. Kasif. Extracting conserved gene expression motifs from gene expression data. In *Pacific Symp. on Biocomp.*, pages 77–88, 2003.

[81] K. Murphy. Learning Bayes net structure from sparse data sets. Technical report, Comp. Sci. Div., UC Berkeley, 2001.

[82] K. Murphy and S. Mian. Modelling gene expression data using dynamic Bayesian networks. Technical report, Computer Science Division, University of California, Berkeley, CA, 1999.

[83] R. M. Neal. Probabilistic inference using Markov chain Monte Carlo methods. Technical Report CRG-TR-93-1, University of Toronto, 1993.

[84] M. Nykter et al. Simulation of microarray data with realistic characteristics. *Bioinf.*, 7:349, Jul. 2006.

[85] H. Pan, L. Liu. Fuzzy Bayesian networks - a general formalism for representation, inference and learning with hybrid Bayesian networks. *IJPRAI*, 14(7):941–962, 2000.

[86] H. Pan, D. McMichael. Fuzzy causal probabilistic networks - a new ideal and practical inference engine. In *Proc. of the 1st Int'l Conf. on Multisource-Multisensor Information Fusion*, Jul. 1998.

[87] H.-S. Park et al. A context-aware music recommendation system using fuzzy Bayesian networks with utility theory. In L. Wang et al., eds, *FSKD*, volume 4223 of *Lecture Notes in Computer Science*, pages 970–979, Xi'an, China, Sept. 24–28 2006. Springer-Verlag. ISBN 3-540-45916-2.

[88] J. Pearl. Causal diagrams for empirical research. *Biometrika*, 82(4): 669–709, 1995.

[89] T. J. Perkins et al. Reverse engineering the gap gene network of drosophila melanogaster. *PLoS Comp. Bio.*, 2(5):e51+, May 2006.

[90] M. Pritsker et al. Whole-genome discovery of transcription factor binding sites by network-level conservation. *Genome Research*, 14(1):99–108, Jan. 2004.

[91] R. Ranawana, V. Palade. Multi-classifier systems: Review and a roadmap for developers. *Int'l J. Hybrid Intelligent Sys.*, 3(1):35–61, 2006.

[92] M. D. Ritchie et al. Optimization of neural network architecture using genetic programming improves detection and modeling of gene-gene interactions in studies of human diseases. *BMC Bioinf.*, 4:28, 2003.

[93] S. J. Russell, P. Norvig. *Artificial Intelligence: A Modern Approach (2nd Edition)*. Prentice Hall, Dec. 2002. ISBN 0137903952.

[94] G. Schwarz. Estimating the dimension of a model. *The Annals of Statistics*, 6(2):461–464, 1978.

[95] E. Segal et al. Module networks: identifying regulatory modules and their condition-specific regulators from gene expression data. *Nat. Genetics*, 34(2):166–176, Jun. 2003.

[96] E. Segal et al. From signatures to models: Understanding cancer using microarrays. *Nat. Genetics*, 37:S38–S45, Jun. 2005. By invitation.

[97] R. Shamir, R. Sharan. *Current Topics in Comp. Bio.*, chapter Algorithmic approaches to clustering gene expression data, pages 269–300. MIT press, Cambridge, Massachusetts, 2002. (T. Jiang, T. Smith, Y. Xu, and M. Q. Zhang, eds).

[98] C. E. Shannon. A mathematical theory of communication. *Bell System Technical Journal*, 27:379–423 & 623–656, Jul. & Oct. 1948.

[99] Q. Sheng et al. Biclustering microarray data by Gibbs sampling. *Bioinf.*, 19:ii196–ii205, 2003.

[100] A. Silvescu, V. Honavar. Temporal Boolean network models of genetic networks and their inference from gene expression time series. *Complex Sys.*, 13:54–70, 2001.

[101] D. S. Sivia. *Data Analysis: A Bayesian Tutorial*. Clarendon Press, Oxford, 1996.

[102] V. A. Smith et al. Evaluating functional network inference using simulations of complex biological systems. *Bioinf.*, 18:S216–S224, 2002.

[103] V. A. Smith et al. Influence of network topology and data collection on network inference. In *Pacific Symp. on Biocomp.*, pages 164–175, 2003.

[104] P. T. Spellman et al. Comprehensive identification of cell cycle-regulated genes of the yeast Saccharomyces cerevisiae by microarray hybridization. *Mol. Bio. of the Cell*, 9(12):3273–3297, Dec. 1998.

[105] P. Spirtes et al. Constructing Bayesian network models of gene expression networks from microarray data. In *Proc. of the Atlantic Symp. on Comp. Bio., Genome Information Sys. and Technology*, 2000.

[106] K. Sterelny, P. E. Griffiths. *Sex and Death : An Introduction to Philosophy of Bio. (Science and Its Conceptual Foundations series)*. University Of Chicago Press, Jun. 1999. ISBN 0226773043.

[107] C. Tang et al. Interrelated two-way clustering: An unsupervised approach for gene expression data analysis. *Proc. of the IEEE 2nd Int'l Symp. on Bioinf. and Bioeng. Conf., 2001*, pages 41–48, 4–6 Nov. 2001.

[108] J. Tegner et al. Reverse engineering gene networks: integrating genetic perturbations with dynamical modeling. *Proc. of the National Academy of Sciences, USA*, 100(10):5944–5949, May 2003.

[109] R. Thomas. Laws for the dynamics of regulatory networks. *Int'l J. Developmental Bio.*, 42:479–485, 1998.
[110] R. Tibshirani et al. Clustering methods for the analysis of DNA microarray data. Technical report, Stanford University, Oct. 1999.
[111] H. Toh, K. Horimoto. Inference of a genetic network by a combined approach of cluster analysis and graphical Gaussian modeling. *Bioinf.*, 18(2):287–297, 2002.
[112] A. H. Tong et al. Systematic genetic analysis with ordered arrays of yeast deletion mutants. *Science*, 294(5550):2364–2368, Dec. 2001.
[113] O. Troyanskaya et al. Missing value estimation methods for DNA microarrays. *Bioinf.*, 17(6):520–525, Jun. 2001.
[114] J.-P. Vert, Y. Yamanishi. Supervised graph inference. In L. K. Saul et al., eds, *Advances in Neural Information Processing Sys. 17*, pages 1433–1440. MIT Press, Cambridge, MA, 2005.
[115] J. Vohradskỳ. Neural network model of gene expression. *FASEB Journal*, 15:846–854, 2001.
[116] Y. Wang et al. Inferring gene regulatory networks from multiple microarray datasets. *Bioinf.*, 22(19):2413–2420, 2006.
[117] R. Xu, D. Wunsch II. Survey of clustering algorithms. *IEEE Transactions on Neural Networks*, 16(3):645–678, May 2005.
[118] Y. Yamanishi et al. Protein network inference from multiple genomic data: a supervised approach. *Bioinf.*, 20(1):363–370, 2004.
[119] E. Yang et al. A novel non-overlapping bi-clustering algorithm for network generation using living cell array data. *Bioinf.*, 23(17):2306–2313, 2007.
[120] J. Yu et al. Using Bayesian network inference algorithms to recover molecular genetic regulatory networks. In *Int'l Conf. on Sys. Bio. (ICSB02)*, Dec. 2002.
[121] J. Yu et al. Advances to Bayesian network inference for generating causal networks from observational biological data. *Bioinf.*, 20(18):3594–3603, 2004.
[122] C. H. Yuh et al. Genomic cis-regulatory logic: Experimental and computational analysis of a sea urchin gene. *Science*, 279:1896–1902, 1998.
[123] Y. Zhang et al. Dynamic Bayesian network (DBN) with structure expectation maximization (SEM) for modeling of gene network from time series gene expression data. In H. R. Arabnia, H. Valafar, eds, *BIOCOMP*, pages 41–47. CSREA Press, 2006.
[124] X. Zhou et al. Gene clustering based on clusterwide mutual information. *J. Comp. Bio.*, 11(1):147–161, 2004.