CHAPTER SIXTEEN

Regulatory network inference

Russell Schwartz

Identifying the complicated patterns of regulatory interactions that control when different genes are active in a cell is a challenging problem, but one essential to understanding how organisms function at a systems level. In this chapter, we will examine the role of computational methods in making such inferences by studying one particularly important version of this problem: the inference of genetic regulatory networks from gene expression data. We will first briefly cover some necessary background on the biology of genetic regulation and technology for measuring the activities of distinct genes in a sample. We will then work through the process of how one can abstract the biological problem of finding interactions among genes into a precise mathematical formulation suitable for computational analysis, starting from very simple variants and gradually working up to models suitable for analysis of large-scale networks. We will also briefly cover key algorithmic issues in working with such models. Finally, we will see how one can transition from simplified pedagogical models to the more detailed, realistic models used in actual research practice. In the process, we will learn about some key concepts in computer science and machine learning, consider how computational scientists think about solving a problem, and see why such thinking has come to play an essential role in the emerging field of systems biology.

1 Introduction

Each cell in a biological organism depends on the coordinated activity of thousands of different kinds of proteins occurring in potentially millions of variations. To function properly, the cell must ensure that each of these proteins is present in the specific places

Bioinformatics for Biologists, ed. P. Pevzner and R. Shamir. Published by Cambridge University Press.
© Cambridge University Press 2011.

it is needed, at the proper times, and in the necessary quantities. An exquisitely complicated network of regulatory interactions ensures these conditions are met throughout the cell's lifetime. Such regulatory interactions include mechanisms for controlling when DNA molecules are properly primed to produce RNA, how often RNA molecules are produced from DNA, how long RNA molecules persist in cells, how often the RNA molecules give rise to proteins, how the proteins are shuttled about the cell, how they are chemically modified at any given time, with which other proteins they are associated, and when they are degraded. These various kinds of regulation are carried out and interconnected through an array of specialized regulatory proteins.

In a regulatory network inference problem, one seeks to infer these complex sets of interactions using indirect measurements of the activities of the individual components of the system. Identifying how genes regulate one another is a fundamental problem in basic biological research into how organisms function, develop, and evolve. Regulatory networks also have important practical applications in helping us to interpret large-scale genomic data and to use them to understand how organisms respond to disease, potential treatments, and other environmental influences. While we cannot hope to do justice to such a complicated problem in one chapter, we can look at one special case of the problem that will illustrate the general principles behind a broad array of work in the field. We will specifically examine the problem of how one can infer transcriptional regulatory networks – networks describing regulatory behavior that act by controlling when RNA is transcribed from DNA – using measurements of RNA expression levels.

The problem of regulatory network inference is interesting not only for its intrinsic scientific merit but also as a model for several important themes in how modern computational biology is practiced and how one reasons about computational inferences from complex biological data sets in general. First, regulatory network inference provides an example of how computational biology intersects with another major trend in modern biological research: systems biology. Systems biology arose out of the realization that one cannot hope to understand the complicated networks of interactions typical of real biological systems by looking at just one or a few components at a time, as was long the standard in biological research. Rather, to infer the overall behavior of a system, researchers must build unified models of the interactions of many components, often using large, noisy data sets. This sort of inference critically depends on computer science methods to enumerate over large numbers of possible models of a given system and weigh the plausibility of each model given the available data. Such systems-level thinking increasingly drives research in biology and has vastly increased the need for computer science expertise in the biological world.

More fundamentally, regulatory network inference is a great example of a problem in machine learning, a subdiscipline of computer science concerned with inferring probabilistic models of complex systems from exactly the kinds of large, error-prone data sets one increasingly encounters in biological contexts. Machine learning has thus emerged as one of the key technologies behind modern high-throughput biology. If we want to understand current directions in computational biology, we need to understand how a researcher thinks about a machine learning problem and some of the basic ways he or she poses and solves such a problem.

Furthermore, regulatory network inference is a problem whose solution critically depends on a careful matching of the class of models one wishes to solve with the data one has available to solve them. It therefore provides a great case study for thinking about the general topic of designing mathematical models for problems in the real world, which is really the beginning of any work in computational biology. The network inference problem is perhaps unusual among those covered in this text in that the hardest, and perhaps most interesting, part of solving it is simply formalizing the problem we wish to solve. This chapter will therefore focus primarily on the issue of formulating the problem mathematically and less so on the details of how one actually solves it.

1.1 The biology of transcriptional regulation

Before we can consider computational approaches to regulatory network inference, we need to know something about the biology of transcriptional regulation. At a high level, a transcriptional regulatory network can be understood in terms of the interactions of two elements: transcription factors and transcription factor binding sites. A transcription factor is a specialized protein that controls when a gene is transcribed to produce RNA. A transcription factor binding site is a small segment of DNA recognized by a particular transcription factor. Transcription factor binding sites are usually, but not exclusively, found near a region called a promoter that occurs near the start of each gene. A promoter serves to recruit the polymerase complex that will read the DNA to produce an RNA transcript. When the transcription factor is present, and perhaps appropriately activated, it will physically bind to its transcription factor binding sites wherever they are exposed in the DNA. The presence of the transcription factor then influences how the transcriptional machinery of the cell acts on the corresponding gene. A given transcription factor can facilitate the recruitment of the polymerase, causing the target gene to be transcribed at a higher level when the transcription factor is present, or it can interfere with the recruitment of the polymerase, reducing expression of the target gene. Furthermore, transcription factors may act in groups, with a specific gene's activity level dependent on the levels of several different transcription factors to different degrees. Figure 16.1 illustrates the concept of transcription factor binding.

Transcription factors are themselves proteins transcribed from genes, and a transcription factor may therefore help to control the expression of another transcription

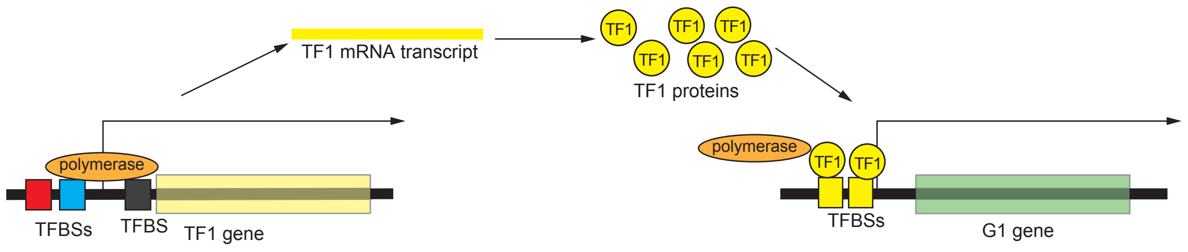

Figure 16.1 Illustration of how transcription factors regulate gene expression. A transcription factor gene (left) produces an mRNA transcript, which in turn produces a protein, TF1, that will bind to transcription factor binding sites (TFBSs) in the promoter regions of other genes, such as the target gene G1 (right). The presence of TF1 is here depicted as blocking recruitment of the RNA polymerase to G1, inhibiting its production of mRNA transcripts.

$$\begin{tikzcd}
 & \text{CAT8} \arrow[r] & \cdots & & \\
 & & \text{RME1} \arrow[r] & \text{IME1} \arrow[r] & \cdots \\
\text{MIG1} \arrow[uur] \arrow[r] \arrow[dr] \arrow[ddr] & \text{SWI5} \arrow[ur] \arrow[r] & \text{ASH1} \arrow[r] & \cdots & \\
 & \text{GAL4} \arrow[r] & \cdots & & \\
 & \text{HAP4} \arrow[r] & \cdots & &
\end{tikzcd}$$

Figure 16.2 Example of a small section of a transcriptional regulatory network from *Saccharomyces cerevisiae* taken from Guelzim *et al.* [1], involved in regulating the response of cell metabolism to stresses such as lack of nutrients. A central "hub" gene, MIG1, responds to the availability of glucose in the cell. It in turn regulates several other transcription factors, including SWI5, which helps to control cell division, and CAT8, GAL4, and HAP4, which regulate various aspects of cell metabolism. SWI5 itself regulates the transcription factors RME1, which helps control meiosis, and ASH1, which regulates genes involved in more specific steps of cell division. RME1 regulates the transcription factor IME1, which regulates its own subset of meiosis-specific genes. Each of these transcription factors regulates various other downstream targets with more specific functions.

factor or even itself. Transcription factors are typically organized into complicated networks of transcription factors regulating other transcription factors, which regulate others, which regulate others, and so forth, before finally activating modules of non-regulatory genes to perform various biological functions. Figure 16.2 shows an example of a small subset of a real regulatory network from the yeast *Saccharomyces cerevisiae* [1].

There are many sources of experimental data by which one might infer a regulatory network and we will primarily confine ourselves to one particular such source of data: gene expression measurements. To date, most such expression data come from *microarrays*. A microarray is a small glass plate covered with thousands or millions of tiny spots, each made up of many copies of a single short DNA strand called a "probe." When one exposes a purified sample of nucleic acid (DNA or RNA) to a microarray, pieces of the nucleic acid from the sample will anneal to those spots whose DNA sequences are complementary to the sample sequences. To use this principle to quantify RNA in a sample, one will typically convert the RNA into complementary DNA strands (called cDNAs) through the process of reverse transcription, break the cDNAs into small pieces, and then fluorescently label the pieces by attaching a small molecule to each piece of cDNA whose presence one can measure by light emissions. When the labeled sample is run over the microarray and then washed away, we expect to find fluorescence only on those spots to which some sample has annealed and roughly in direct proportion to how much sample has annealed there. We can thus use these fluorescence intensities to give us a quantitative measure of how much RNA complementary to each probe was present in the sample. Figure 16.3 shows an example of a microarray. A typical expression microarray may have a few probes each for every known gene in a given organism's genome, as well as potentially others to detect non-coding genes and other non-genic sources of transcribed RNA. For our purposes, we will simplify a bit and assume that a microarray gives us a measure of how much RNA from each gene is present, or *expressed*, in a given sample.

In a typical microarray experiment, one will use several copies of a given microarray and apply them to a collection of samples gathered under different conditions. These conditions may correspond to different time points, different individuals from whom a tissue sample has been taken, different nutrients or drugs that have been applied to samples, or any other sort of variation that might be expected to change the activities of genes. The data from each gene across all samples are commonly normalized relative to some control sample (typically a pooled mixture of all conditions), giving a measure of the expression level of that gene in each condition relative to the control. Thus, we can think of an array as providing us with a matrix of relative expression data, in which we have one column of data for each condition and one row for each gene. We will assume that this matrix of gene expression measurements represents the data from which we wish to infer the regulatory network.

The preceding description of the problem and the data available to solve it omits many details, as one always must in posing a computational problem, but it provides a reasonable beginning for formulating a mathematical model of the network inference problem. In the remainder of this chapter, we will survey the basic ideas behind how one can go from gene expression measurements to inferred regulatory networks. We will seek to build an intuition for the problem by starting with a simple variant and gradually moving toward a realistic model of the problem in practice. We will conclude with some discussion of the further complications that come up in real-world systems and how the interested reader can learn more about these topics.

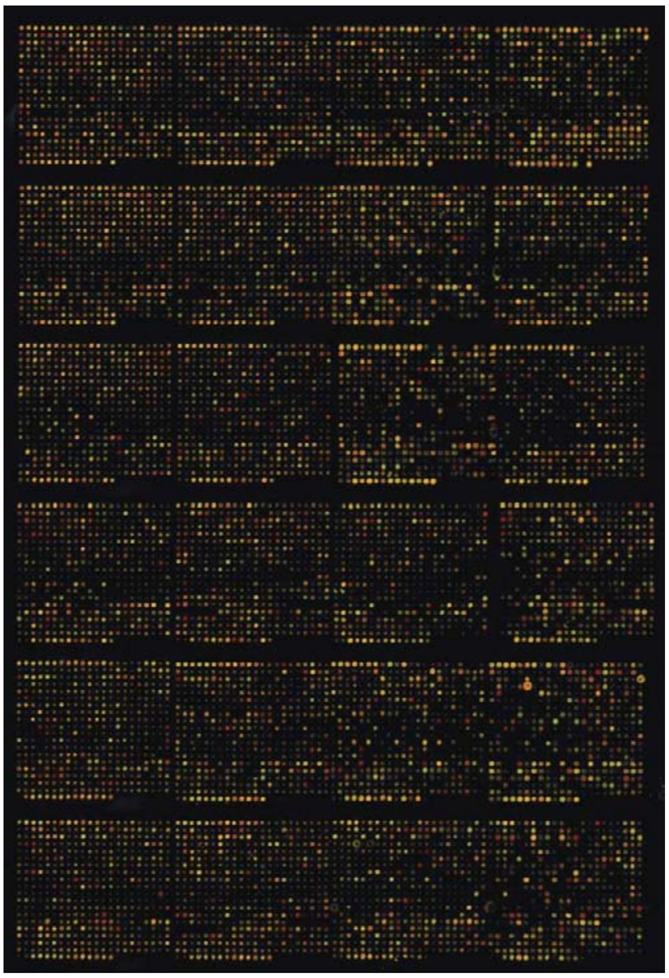

Figure 16.3 A microarray slide showing relative levels of nucleic acid in two samples that are complementary to a set of probes [2]. The two samples are labeled in red and green, producing yellow spots when the samples show similar expression levels and red or green spots when one sample shows substantially different expression than the other.

2 Developing a formal model for regulatory network inference

2.1 Abstracting the problem statement

If we want to develop a computational method for the regulatory network inference problem, we need to begin by developing an abstraction of the problem, i.e. a formal mathematical description of what we will consider the inputs and outputs of the problem to be. Abstracting a problem requires precisely defining what data we assume we have available to us and how we will represent those data, as well as what an answer to

	C1	C2	C3	C4	C5	C6	C7	C8
G1	1	1	0	0	1	1	1	0
G2	0	1	0	1	1	1	1	0
G3	0	0	1	0	0	0	0	1
G4	0	0	0	0	0	1	0	1

Figure 16.4 A toy example of a discretized gene expression data set describing the activities of four genes (G1–G4) in eight conditions (C1–C8). Each row of the matrix (running left to right) describes the activity of one gene under all conditions and each column (running top to bottom) describes the activity of all genes under one condition.

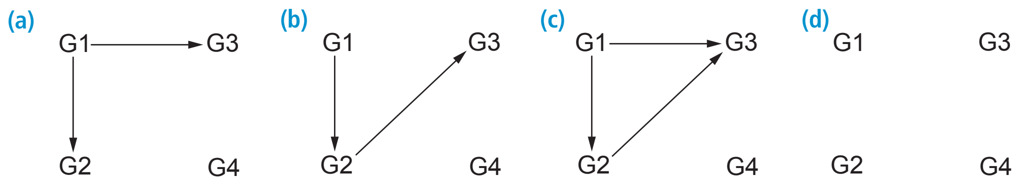

Figure 16.5 A set of possible networks for the expression data of Figure 16.4.

the problem will look like and how we will choose among possible answers. To help us develop an intuition for posing such a problem, we will start with a very simple abstraction of transcriptional regulatory network inference.

We will first develop an abstraction of the input data. We can begin by assuming that the only data we have available to us are a set of microarray measurements comprising a matrix in which each element describes the expression level of one gene in one condition. To keep things simple for the moment, we will further assume that each data point takes on one of two possible values: "1" if the gene is expressed at a higher than average level (informally, that the gene is "on" or "active") and "0" if the gene is expressed at a lower than average level (informally, that the gene is "off" or "inactive"). We are thus making the decision for this level of abstraction to discard the true continuous (real-valued) data that would be produced by the microarray in order to derive a more conceptually tractable model. Figure 16.4 shows a hypothetical example of such an input data set for four genes in eight conditions.

We must also define some formalized statement of the output of a network inference algorithm. In a generic sense, our output should be a model of a network identifying pairs of genes that appear to regulate one another. In this simple version of the problem, we will pick a binary output as well: for any ordered pair of genes, G1 and G2, we will say that either G1 regulates G2 or G1 does not regulate G2. We can represent the output of the inference problem by the set of ordered pairs of genes corresponding to regulatory relationships. This representation of the output can be visualized as a network, also called a *graph*, consisting of a set of vertices with pairs of vertices (or nodes) joined by edges. Here, we create one node for each gene and place a directed edge between any pair of genes Gi and Gj for which Gi regulates Gj. Figure 16.5 shows a few examples of possible networks for the data of Figure 16.4 according to this particular representation of the model.

In choosing this particular representation, we are again making some assumptions about what we will and will not consider important in a model. We are choosing to use a model that represents directionality of regulation; "G1 regulates G2" means something different in our model than "G2 regulates G1." A regulatory network inference algorithm need not distinguish between those possibilities. On the other hand, we are choosing to ignore the fact that regulation can be positive (activation) or negative (repression). We could alternatively have chosen to maintain a sign on each regulatory relationship to distinguish these possibilities, as is typically done in network models. We are similarly ignoring the fact that regulatory relationships could have different strengths (G1 might regulate G2 strongly or weakly), something that is certainly true and which one might denote by placing a numerical weight on each edge. Regulatory relationships could in fact be described by essentially arbitrary functions of expression. We will also assume that genes cannot self-regulate and that we do not have directed cycles, which are paths in the network that lead from a gene back to itself. These assumptions are not, in fact, accurate, but help us establish a more conceptually simple model. Making such trade-offs, in a way that is appropriate to the data available to us and the uses to which we want to put them, is one of the hardest but most important issues in developing a formal model. Our goal in developing the present model is to help us understand the inference problem and so we favor a relatively simple model, but we might favor a very different model if we had some other goal in mind.

The two formalizations defined in this section – a formal representation of the input to the problem and a formal representation of the output to the problem – are two of the main ingredients in a formal problem statement. There is a third component we will need, though: a formal specification of how we will judge any given output for a given input. This measure of the quality of a possible output, known as an *objective function*, is not so easy to define for a complicated problem like this. We will spend the next few subsections showing how to define a precise objective function for the regulatory network inference problem, starting with some intuition behind the problem and building up to a general formulation.

2.2 An intuition for network inference

A good starting point for an objective function is to consider informally how we can reason about the evidence available to us to develop a plausible model.[1] We can see

[1] The terminology here may be confusing to readers previously familiar with mathematical modeling, as the term "model" has a different meaning in the mathematical modeling community than it does in the machine

at an intuitive level how one might evaluate possible regulatory networks for a given data set by closely examining the data of Figure 16.4. We can observe that genes G1 and G2 are generally, although not always, active and inactive in the same conditions. We might therefore guess that G1 regulates G2, and specifically that G1 activates G2. G1 and G3 are generally active in opposite conditions. This, too, might be seen as evidence of regulation, in this case perhaps that G1 represses G3. G4's activity appears unrelated to that of G1, G2, or G3 and we might therefore conclude that it is probably not in a regulatory relationship with any of them. We therefore might conjecture that Figure 16.5a provides a good model of the regulatory network we want to infer.

Intuition can only take us so far, though. The same reasoning that led us to the network of Figure 16.5a could just as easily lead us to Figure 16.5b or Figure 16.5c. For that matter, we do not know if the correlations we think we see in the data are sufficiently well supported by the data that we should believe them. Perhaps Figure 16.5d (no regulation) is the true network and the apparent correlations arose from random chance. If we want to be able to choose among these possibilities, we will need to be a bit more precise about how we we will decide what makes for a "plausible" model.

2.3 Formalizing the intuition for an inference objective function

To go from intuition to a formal computational problem, we will need to come up with a way of specifying precisely how good one model is relative to another. A common way of accomplishing this for noisy data inference problems is to define the problem in terms of probabilities. We will use a particular variant of a probabilistic model, known as a likelihood model, in which we judge a model by how probable we think it is that the observed data could have been generated from that model. This probability is known as the *likelihood* of the model. We then seek the model that gives us the greatest likelihood, known as the *maximum likelihood model*.

To put the intuitive problem into a formal framework, we first need to develop some notation. As in Figure 16.4, we will assume our input is a matrix, which we will call D. We will refer to each row of the matrix, corresponding to a single gene, as a vector $\mathbf{d}_i$. So for example, the row for gene G1 is represented by the vector $\mathbf{d}_1 = [11001110]$. Each element of each row is represented by a single scalar (non-vector) value d_{ij}. For example, the expression of gene G1 in condition C2 is given by $d_{12} = 1$.

We will also need a notation to refer to our output, i.e. the regulatory network we would like to infer. As discussed in the preceding section, our output can be represented

learning community. We will follow machine learning practice in using "model" to refer to a particular output of the network inference problem, i.e. a network modeling the regulatory interactions among the input genes. In mathematical modeling terminology, a "model" of the problem would refer instead to what we have here called the "formal problem statement."

by a graph, which we can call G. Any given G is itself defined by a set of vertices V, with one vertex per gene, and a set of edges E, with potentially one edge for each pair of genes. Thus, for example, we can refer to the model of Figure 16.5a by the graph

$$G = (V, E) = (\{v_1, v_2, v_3, v_4\}, \{(v_1, v_2), (v_1, v_3)\}). \tag{16.1}$$

The vertex set contains four vertices, one for each of the four genes, and the edge set contains two edges, one for each of the two posited regulatory relationships.

We will be working specifically with probability models, which will require that our models include some additional information to let us determine how likely the model is to produce a given set of expression data. We will defer the details of these probabilities for the moment and just declare that we have some additional set P of probability parameters contained in the model. For a maximum likelihood model, we define those additional values contained in P to be whatever will make the likelihood function as large as possible. The exact contents of P will depend on the graph elements V and E, as we will see shortly. For our formal purposes, then, an output model M consists of the elements (V, E, P) defining the proposed regulatory relationships and the probability of outputting any given expression matrix D from that model M. This probability, called the likelihood of the model, is denoted by the probability function $Pr\{D|M\}$, read as "the probability of D given M." Our goal will be to find

$$\max_M Pr\{D|M\},$$

i.e. the maximum likelihood model over all possible models M for a given data set D. We still have more work to do, though, to define precisely what it means mathematically to find the M maximizing $Pr\{D|M\}$.

2.3.1 Maximum likelihood for one gene

We next need to specify how one actually evaluates the function $Pr\{D|M\}$ for a known D and M. We can start by considering just one gene, G1, whose expression is described by the vector $\mathbf{d}_1 = [11001110]$. Since we are now assuming that there is only one gene, we cannot have any regulatory relationships. Therefore, we have only one possible graph G for our model: $G = (V, E) = (\{v_1\}, \{\})$, a vertex set of one node and an empty edge set. To determine the likelihood of the model, we will need to evaluate $Pr\{\mathbf{d}_1|(V, E, P)\}$, the probability that the model $M = (V, E, P)$ would lead to the output vector $\mathbf{d}_1$. It is a universal law of probability that the probability of a pair of independent outcomes is the product of the probabilities of the individual outcomes. Therefore, if we assume that each condition represents an independent experiment then the probability of outputting the complete vector $\mathbf{d}_1$ will be given by the product of probabilities of outputting each element of that vector. Thus, if we knew the probability that G1 was active in a given condition given our model M ($Pr\{d_{1i} = 1|M\}$, which we will call $p_{1,1}$) and the probability that G1 was inactive in a given condition given model M ($Pr\{d_{1i} = 0|M\}$, which we will call $p_{1,0}$) then we could determine the probability of the whole vector as follows:

$$\begin{aligned} Pr\{\mathbf{d}_1 = [11001110]|M\} &= Pr\{d_{11} = 1|M\} \times Pr\{d_{12} = 1|M\} \times Pr\{d_{13} = 0|M\} \\ &\quad \times Pr\{d_{14} = 0|M\} \times Pr\{d_{15} = 1|M\} \times Pr\{d_{16} = 1|M\} \\ &\quad \times Pr\{d_{17} = 1|M\} \times Pr\{d_{18} = 0|M\} \\ &= p_{1,1} \times p_{1,1} \times p_{1,0} \times p_{1,0} \times p_{1,1} \times p_{1,1} \times p_{1,1} \times p_{1,0}. \end{aligned} \tag{16.2}$$

For this particular model, $p_{1,1}$ and $p_{1,0}$ are precisely the additional model parameters P that we need to know to finish formally specifying the model.

As noted above, those additional values contained in P must be whatever will make the likelihood function as large as possible. Fortunately, those maximum likelihood values are easy to determine, at least for this model. The values that will give the maximum likelihood are given by the fractions of observations corresponding to each given probability in the observed data. In other words, we observe that G1 is active in five conditions out of eight, giving a maximum likelihood estimate of $p_{1,1} = 5/8$. G1 is inactive in three conditions out of eight, giving a maximum likelihood estimate of $p_{1,0} = 3/8$. This procedure for learning optimal parameters of P then lets us complete the formal specification of our model M as follows:

$$M = (V, E, P) = \left(\{v_1\}, \{\}, \left\{Pr\{d_{1i} = 1|M\} = \frac{5}{8}, Pr\{d_{1i} = 0|M\} = \frac{3}{8}\right\}\right). \tag{16.3}$$

We also now have all the tools we need to come up with a precise quantitative statement of the likelihood of the data given the model for this simple one-gene case:

$$\begin{aligned} Pr\{\mathbf{d}_1 = [11001110]|M\} &= p_{1,1} \times p_{1,1} \times p_{1,0} \times p_{1,0} \times p_{1,1} \times p_{1,1} \times p_{1,1} \times p_{1,0} \\ &= \frac{5}{8} \times \frac{5}{8} \times \frac{3}{8} \times \frac{3}{8} \times \frac{5}{8} \times \frac{5}{8} \times \frac{5}{8} \times \frac{3}{8} \approx 0.00503. \end{aligned} \tag{16.4}$$

This number is not too useful to us when we only have one model to consider, but will become our measure for evaluating possible models with more complicated examples.

2.3.2 Maximum likelihood for two genes

Now that we know how to evaluate a likelihood function for one gene, we will move on to considering two genes, G1 and G2, simultaneously. There are now three possible hypotheses we can consider: neither G1 nor G2 regulates the other, G1 regulates G2, or G2 regulates G1. Each of these hypotheses can be converted into a formal model using the concepts laid out above. We will want to determine which of these three models maximizes the likelihood of both genes given the model:

$$\max_M Pr\{\mathbf{d}_1 = [11001110], \mathbf{d}_2 = [01011110] | M\}. \tag{16.5}$$

To keep the notation from getting too cumbersome, we will henceforth abbreviate the above likelihood as $Pr\{\mathbf{d}_1, \mathbf{d}_2 | M\}$.

Our first model, which we will call M_1, assumes that neither G1 nor G2 regulates the other. Formally, $M_1 = (V_1, E_1, P_1) = (\{v_1, v_2\}, \{\}, P_1)$, where we will again defer defining P_1 precisely until we see how we will use it. For this model, we can treat the outputs $\mathbf{d}_1$ and $\mathbf{d}_2$ as independent sets of data since we assume neither gene regulates the other. As we noted above, the assumption that two variables are independent means that we can derive their joint probability by multiplying their individual probabilities:

$$Pr\{\mathbf{d}_1, \mathbf{d}_2 | M_1\} = Pr\{\mathbf{d}_1 | M_1\} \times Pr\{\mathbf{d}_2 | M_1\}. \tag{16.6}$$

We can then evaluate each of these two probabilities exactly as we did in the one-gene case. The additional probability parameters P_1 that we will need to know are the probability G1 is active or inactive independently of G2 and the probability G2 is active or inactive independently of G1. Extending our notation from the one-gene case, $P_1 = \{p_{1,1}, p_{1,0}, p_{2,1}, p_{2,0}\}$. We can derive maximum likelihood estimates for these probabilities as above by observing the fraction of outputs that are 1 or 0 for each gene. As before, we can estimate $p_{1,1} = 5/8$ and $p_{1,0} = 3/8$. We similarly observe five 1s and three 0s for G2, so we estimate $p_{2,1} = 5/8$ and $p_{2,0} = 3/8$. We then get the following estimate for the likelihood of G1's outputs:

$$\begin{aligned} Pr\{\mathbf{d}_1 | M_1\} &= p_{1,1} \times p_{1,1} \times p_{1,0} \times p_{1,0} \times p_{1,1} \times p_{1,1} \times p_{1,1} \times p_{1,0} \\ &= \frac{5}{8} \times \frac{5}{8} \times \frac{3}{8} \times \frac{3}{8} \times \frac{5}{8} \times \frac{5}{8} \times \frac{5}{8} \times \frac{3}{8} \approx 0.00503, \end{aligned} \tag{16.7}$$

and the following for G2's outputs:

$$\begin{aligned} Pr\{\mathbf{d}_2 | M_1\} &= p_{2,0} \times p_{2,1} \times p_{2,0} \times p_{2,1} \times p_{2,1} \times p_{2,1} \times p_{2,1} \times p_{2,0} \\ &= \frac{3}{8} \times \frac{5}{8} \times \frac{3}{8} \times \frac{5}{8} \times \frac{5}{8} \times \frac{5}{8} \times \frac{5}{8} \times \frac{3}{8} \approx 0.00503. \end{aligned} \tag{16.8}$$

Thus,

$$Pr\{\mathbf{d}_1, \mathbf{d}_2 | M_1\} = \left(\frac{5}{8}\right)^5 \times \left(\frac{3}{8}\right)^3 \times \left(\frac{5}{8}\right)^5 \times \left(\frac{3}{8}\right)^3 \approx 2.53 \times 10^{-5}. \tag{16.9}$$

Things get trickier when we move to a model assuming some regulation. We will now consider the possibility that G1 regulates G2. For this model, $M_2 = (V_2, E_2, P_2) = (\{v_1, v_2\}, \{(v_1, v_2)\}, P_2)$. That is, the model assumes a single regulatory edge running from v_1 to v_2 representing the assumption that G2's expression is a function of G1's expression. As before, we can assume G1's expression is an independent random variable:

$$\begin{aligned} Pr\{\mathbf{d}_1|M_2\} &= p_{1,1} \times p_{1,1} \times p_{1,0} \times p_{1,0} \times p_{1,1} \times p_{1,1} \times p_{1,1} \times p_{1,0} \\ &= \frac{5}{8} \times \frac{5}{8} \times \frac{3}{8} \times \frac{3}{8} \times \frac{5}{8} \times \frac{5}{8} \times \frac{5}{8} \times \frac{3}{8} \approx 0.00503. \end{aligned} \tag{16.10}$$

We must, however, assume that G2's expression depends on G1's. More formally, our likelihood function will need a term of the form $Pr\{\mathbf{d}_2|M_2, \mathbf{d}_1\}$, which we read as "the probability of $\mathbf{d}_2$ given M_2 and $\mathbf{d}_1$." This function will depend on a model of how likely it is that d_{2i} is 1 when d_{1i} is 1 as well as how likely it is that d_{2i} is 1 when d_{1i} is 0. We will therefore need to specify four probability parameters:

- $p_{2,0,0}$: the probability $d_{2i} = 0$ when $d_{1i} = 0$
- $p_{2,0,1}$: the probability $d_{2i} = 0$ when $d_{1i} = 1$
- $p_{2,1,0}$: the probability $d_{2i} = 1$ when $d_{1i} = 0$
- $p_{2,1,1}$: the probability $d_{2i} = 1$ when $d_{1i} = 1$

P_2 is defined by the probabilities we need to evaluate $Pr\{\mathbf{d}_1|M_2\}$ and those we need to evaluate $Pr\{\mathbf{d}_2|M_2, \mathbf{d}_1\}$, so $P_2 = \{p_{1,1}, p_{1,0}, p_{2,0,0}, p_{2,0,1}, p_{2,1,0}, p_{2,1,1}\}$. As before, we can derive maximum likelihood estimates of these parameters by counting the fraction of times we observe each value of G2 for each value of G1. We have five instances in which G1 is 1 and four of these five also have G2 = 1. Thus, $p_{2,1,1} = 4/5$ and $p_{2,0,1} = 1/5$. Similarly, we have three instances in which G1 = 0 and two of these three have G2 = 0. Thus, $p_{2,0,0} = 2/3$ and $p_{2,1,0} = 1/3$. Therefore,

$$\begin{aligned} Pr\{\mathbf{d}_2|M_2, \mathbf{d}_1\} &= p_{2,0,1} \times p_{2,1,1} \times p_{2,0,0} \times p_{2,1,0} \times p_{2,1,1} \times p_{2,1,1} \times p_{2,1,1} \times p_{2,0,0} \\ &= \frac{1}{5} \times \frac{2}{3} \times \frac{1}{3} \times \frac{4}{5} \times \frac{4}{5} \times \frac{4}{5} \times \frac{4}{5} \times \frac{2}{3} \approx 0.0121. \end{aligned} \tag{16.11}$$

The complete likelihood for this model is then given by

$$Pr\{\mathbf{d}_1, \mathbf{d}_2|M_2\} = Pr\{\mathbf{d}_1|M_2\}Pr\{d_2|\mathbf{d}_1, M_2\} \approx 0.00503 \times 0.0121 \approx 6.10 \times 10^{-5}.$$

We can therefore conclude that M_2 is a more likely explanation for the data than M_1.

Evaluating the final model for two genes, $M_3 = (V_3, E_3, P_3) = (\{v_1, v_2\}, \{(v_2, v_1)\}, P_3)$, proceeds analogously to the evaluation of M_2:

$$Pr\{\mathbf{d}_1, \mathbf{d}_2|M_2\} = Pr\{\mathbf{d}_2|M_3\}Pr\{\mathbf{d}_1|\mathbf{d}_2, M_2\}, \tag{16.12}$$

i.e. the model is the product of a term accounting for the independent likelihood of G2 and the likelihood of G1 given that it is a function of G2. We can evaluate $Pr\{\mathbf{d}_2|M_3\}$

as we did for M_1:

$$Pr\{\mathbf{d}_2|M_3\} = p_{2,0} \times p_{2,1} \times p_{2,0} \times p_{2,1} \times p_{2,1} \times p_{2,1} \times p_{2,1} \times p_{2,0}$$
$$= \frac{3}{8} \times \frac{5}{8} \times \frac{3}{8} \times \frac{5}{8} \times \frac{5}{8} \times \frac{5}{8} \times \frac{5}{8} \times \frac{3}{8} \approx 0.00503. \tag{16.13}$$

We can also evaluate $Pr\{\mathbf{d}_1|\mathbf{d}_2, M_3\}$ as we did for $Pr\{\mathbf{d}_2|\mathbf{d}_1, M_2\}$. We define a new set of parameters:

- $p_{1,0,0}$: the probability $d_{1i} = 0$ when $d_{2i} = 0$
- $p_{1,0,1}$: the probability $d_{1i} = 0$ when $d_{2i} = 1$
- $p_{1,1,0}$: the probability $d_{1i} = 1$ when $d_{2i} = 0$
- $p_{1,1,1}$: the probability $d_{1i} = 1$ when $d_{2i} = 1$

We estimate the parameters by identifying all occurrences of G2 = 0 and G2 = 1 and, for each, counting how often G1 = 0 and G1 = 1: $p_{1,0,0} = 1/3$, $p_{1,1,0} = 2/3$, $p_{1,0,1} = 4/5$, $p_{1,1,1} = 1/5$. These probabilities collectively define $P_3 = \{p_{2,1}, p_{2,0}, p_{1,0,0}, p_{1,0,1}, p_{1,1,0}, p_{1,1,1}\}$. Then,

$$Pr\{\mathbf{d}_2|M_3, \mathbf{d}_1\} = p_{1,1,0} \times p_{1,1,1} \times p_{1,0,0} \times p_{1,0,1} \times p_{1,1,1} \times p_{1,1,1} \times p_{1,1,1} \times p_{1,0,0}$$
$$= \frac{1}{5} \times \frac{2}{3} \times \frac{1}{3} \times \frac{4}{5} \times \frac{4}{5} \times \frac{4}{5} \times \frac{4}{5} \times \frac{2}{3} \approx 0.0121. \tag{16.14}$$

Putting it all together gives us the full model likelihood

$$Pr\{\mathbf{d}_1, \mathbf{d}_2|M_2\} \approx 0.00503 \times 0.0121 \approx 6.10 \times 10^{-5}. \tag{16.15}$$

Thus, M_3 has the same likelihood as M_2.

If we had just the two genes to consider then we could run through these possibilities and come to the final conclusion that M_1 is a poorer model of the data, while M_2 and M_3 are better models than M_1 and equally good to one another.

It is worth noting that it is not a coincidence that M_2 and M_3 yield identical likelihoods. In fact, the problem as we posed it guarantees that the likelihood of any model will be identical to that of a mirror image model, in which the directionality of all edges is reversed. We might therefore conclude that our formalization of the problem was, in this respect, poorly matched to our data and that we should have posed the problem in terms of finding undirected networks. Alternatively, we might consider ways of adding additional information by which we might disambiguate the directions of regulatory edges, a topic we will consider later in the chapter. For now, however, we will ignore this issue and continue working through the problem as we have formalized it.

2.3.3 From two genes to several genes

The mathematics became fairly complicated when we moved from one to two genes, so one might expect that moving to three or four will be much harder. In fact, though, it is not much more difficult to reason about four genes, or forty thousand, than it is to reason about two. The number of models one can potentially consider goes up rapidly with increasing numbers of genes, but evaluating the likelihood of any given model is not that much harder conceptually. To see why, let us consider just three of the possible models of all four genes from Figure 16.4.

One model we might wish to consider is that no gene regulates any other. We can call this model M_1', which would correspond to the assumption that

$$Pr\{\mathbf{d}_1, \mathbf{d}_2, \mathbf{d}_3, \mathbf{d}_4 | M_1'\} = Pr\{\mathbf{d}_1 | M_1'\} \times Pr\{\mathbf{d}_2 | M_1'\} \times Pr\{\mathbf{d}_3 | M_1'\} \times Pr\{\mathbf{d}_4 | M_1'\}. \tag{16.16}$$

We can evaluate each of these terms just as we did when we considered two genes. For example, to evaluate $Pr\{\mathbf{d}_1 | M_1'\}$, we define variables $p_{1,0}$ and $p_{1,1}$ representing the probabilities G1 is 0 or 1, estimate these probabilities by counting the fraction of occurrences of G1 = 0 and G1 = 1, and multiply probabilities across conditions:

$$\begin{aligned} Pr\{\mathbf{d}_1 | M_1'\} &= p_{1,1} \times p_{1,1} \times p_{1,0} \times p_{1,0} \times p_{1,1} \times p_{1,1} \times p_{1,1} \times p_{1,0} \\ &= \frac{5}{8} \times \frac{5}{8} \times \frac{3}{8} \times \frac{3}{8} \times \frac{5}{8} \times \frac{5}{8} \times \frac{5}{8} \times \frac{3}{8} \approx 0.00503. \end{aligned} \tag{16.17}$$

Similarly,

$$\begin{aligned} Pr\{\mathbf{d}_2 | M_1'\} &= p_{2,0} \times p_{2,1} \times p_{2,0} \times p_{2,1} \times p_{2,1} \times p_{2,1} \times p_{2,1} \times p_{2,0} \\ &= \frac{3}{8} \times \frac{5}{8} \times \frac{3}{8} \times \frac{5}{8} \times \frac{5}{8} \times \frac{5}{8} \times \frac{5}{8} \times \frac{3}{8} \approx 0.00503, \\ Pr\{\mathbf{d}_3 | M_1'\} &= p_{3,0} \times p_{3,0} \times p_{3,1} \times p_{3,0} \times p_{3,0} \times p_{3,0} \times p_{3,0} \times p_{3,1} \\ &= \frac{6}{8} \times \frac{6}{8} \times \frac{2}{8} \times \frac{6}{8} \times \frac{6}{8} \times \frac{6}{8} \times \frac{6}{8} \times \frac{2}{8} \approx 0.0111, \\ Pr\{\mathbf{d}_4 | M_1'\} &= p_{4,0} \times p_{4,0} \times p_{4,0} \times p_{4,0} \times p_{4,0} \times p_{4,1} \times p_{4,0} \times p_{4,1} \\ &= \frac{6}{8} \times \frac{6}{8} \times \frac{6}{8} \times \frac{6}{8} \times \frac{6}{8} \times \frac{2}{8} \times \frac{6}{8} \times \frac{2}{8} \approx 0.0111. \end{aligned} \tag{16.18}$$

The formal statement of the model is, then,

$$M_1' = (V_1', E_1', P_1') = (\{v_1, v_2, v_3, v_4\}, \{\}, \{p_{1,0}, p_{1,1}, p_{2,0}, p_{2,1}, p_{3,0}, p_{3,1}, p_{4,0}, p_{4,1}\}) \tag{16.19}$$

and the likelihood of the whole model is

$$Pr\{\mathbf{d}_1, \mathbf{d}_2, \mathbf{d}_3, \mathbf{d}_4 | M'_1\} \approx 0.00503 \times 0.00503 \times 0.0111 \times 0.0111 \approx 3.00 \times 10^{-9}. \tag{16.20}$$

We might alternatively consider a model M'_2 in which G1 regulates G2, G2 regulates G3, and nothing regulates G1 or G4. M'_2 corresponds to the assumption that

$$Pr\{\mathbf{d}_1, \mathbf{d}_2, \mathbf{d}_3, \mathbf{d}_4 | M'_2\} = Pr\{\mathbf{d}_1 | M'_2\} \times Pr\{\mathbf{d}_2 | \mathbf{d}_1, M'_2\} \times Pr\{\mathbf{d}_3 | \mathbf{d}_2, M'_2\} \times Pr\{\mathbf{d}_4 | M'_2\}. \tag{16.21}$$

The G1 and G4 terms can be evaluated just as with model M'_1:

$$\begin{aligned}
Pr\{\mathbf{d}_1 | M'_2\} &= p_{1,1} \times p_{1,1} \times p_{1,0} \times p_{1,0} \times p_{1,1} \times p_{1,1} \times p_{1,1} \times p_{1,0} \\
&= \frac{5}{8} \times \frac{5}{8} \times \frac{3}{8} \times \frac{3}{8} \times \frac{5}{8} \times \frac{5}{8} \times \frac{5}{8} \times \frac{3}{8} \approx 0.00503, \qquad (16.22) \\
Pr\{\mathbf{d}_4 | M'_2\} &= p_{4,0} \times p_{4,0} \times p_{4,0} \times p_{4,0} \times p_{4,0} \times p_{4,1} \times p_{4,0} \times p_{4,1} \\
&= \frac{6}{8} \times \frac{6}{8} \times \frac{6}{8} \times \frac{6}{8} \times \frac{6}{8} \times \frac{2}{8} \times \frac{6}{8} \times \frac{2}{8} \approx 0.0111.
\end{aligned}$$

The G2 term can be handled just as when we considered G1 and G2 alone:

$$\begin{aligned}
Pr\{\mathbf{d}_2 | M'_2, \mathbf{d}_1\} &= p_{2,0,1} \times p_{2,1,1} \times p_{2,0,0} \times p_{2,1,0} \times p_{2,1,1} \times p_{2,1,1} \times p_{2,1,1} \times p_{2,0,0} \\
&= \frac{1}{5} \times \frac{2}{3} \times \frac{1}{3} \times \frac{4}{5} \times \frac{4}{5} \times \frac{4}{5} \times \frac{4}{5} \times \frac{2}{3} \approx 0.0121.
\end{aligned} \tag{16.23}$$

Finally, the G3 term can be handled analogously to the G2 term:

$$\begin{aligned}
Pr\{\mathbf{d}_3 | \mathbf{d}_2, M'_2\} &= p_{3,0,0} \times p_{3,0,1} \times p_{3,1,0} \times p_{3,0,1} \times p_{3,0,1} \times p_{3,0,1} \times p_{3,0,1} \times p_{3,1,0} \\
&= \frac{1}{3} \times \frac{5}{5} \times \frac{2}{3} \times \frac{5}{5} \times \frac{5}{5} \times \frac{5}{5} \times \frac{5}{5} \times \frac{2}{3} \approx 0.148.
\end{aligned} \tag{16.24}$$

We thus get the complete likelihood:

$$Pr\{\mathbf{d}_1, \mathbf{d}_2, \mathbf{d}_3, \mathbf{d}_4 | M'_2\} = 0.00503 \times 0.0121 \times 0.148 \times 0.0111 \approx 1.00 \times 10^{-7}. \tag{16.25}$$

We can therefore conclude that M'_2 has a substantially higher likelihood than M'_1.

We can also consider models in which a given gene is a function of more than one regulator. For example, suppose we consider a model M'_3 in which G1, G2, and G4 are unregulated but G3 is regulated by both G1 and G2. For this model, we assume that

$$Pr\{\mathbf{d}_1, \mathbf{d}_2, \mathbf{d}_3, \mathbf{d}_4 | M'_3\} = Pr\{\mathbf{d}_1 | M'_3\} \times Pr\{\mathbf{d}_2 | M'_3\} \times Pr\{\mathbf{d}_3 | \mathbf{d}_1, \mathbf{d}_2, M'_3\} \times Pr\{\mathbf{d}_4 | M'_3\}. \tag{16.26}$$

We can evaluate the G1, G2, and G4 terms exactly as with model M_1' above:

$$Pr\{\mathbf{d}_1|M_3'\} = p_{1,1} \times p_{1,1} \times p_{1,0} \times p_{1,0} \times p_{1,1} \times p_{1,1} \times p_{1,1} \times p_{1,0} \approx 0.00503. \tag{16.27}$$

Similarly,

$$Pr\{\mathbf{d}_2|M_3'\} = p_{2,0} \times p_{2,1} \times p_{2,0} \times p_{2,1} \times p_{2,1} \times p_{2,1} \times p_{2,1} \times p_{2,0} \approx 0.00503,$$
$$Pr\{\mathbf{d}_4|M_3'\} = p_{4,0} \times p_{4,0} \times p_{4,0} \times p_{4,0} \times p_{4,0} \times p_{4,1} \times p_{4,0} \times p_{4,1} \approx 0.0111. \tag{16.28}$$

To evaluate the G3 term, however, we will need to consider its dependence on states of both G1 and G2. We can capture this dependence with the following set of probability parameters:

- $p_{3,0,0,0}$: the probability $d_{3i} = 0$ when $d_{1i} = 0$ and $d_{2i} = 0$
- $p_{3,1,0,0}$: the probability $d_{3i} = 1$ when $d_{1i} = 0$ and $d_{2i} = 0$
- $p_{3,0,0,1}$: the probability $d_{3i} = 0$ when $d_{1i} = 0$ and $d_{2i} = 1$
- $p_{3,1,0,1}$: the probability $d_{3i} = 1$ when $d_{1i} = 0$ and $d_{2i} = 1$
- $p_{3,0,1,0}$: the probability $d_{3i} = 0$ when $d_{1i} = 1$ and $d_{2i} = 0$
- ...

We can then say

$$Pr\{\mathbf{d}_3|\mathbf{d}_1, \mathbf{d}_2, M_3'\} = p_{3,0,1,0} \times p_{3,0,1,1} \times p_{3,1,0,0} \times p_{3,0,0,1} \times p_{3,0,1,1} \times p_{3,0,1,1} \times p_{3,0,1,1} \times p_{3,1,0,0}. \tag{16.29}$$

To estimate the probability parameters, we need to count values of G3 for each combination of values of G1 and G2. For example, to evaluate $p_{3,0,1,1}$ (the probability G3 $=0$ given that G1 $=1$ and G2 $=1$), we note that there are four conditions in which G1 $=1$ and G2 $=1$ and all four have G3 $= 0$. Thus, $p_{3,0,1,1} = 4/4$. Similarly, we estimate $p_{3,0,1,0} = 1/1$ and $p_{3,1,0,0} = 2/2$. We would then conclude that

$$Pr\{\mathbf{d}_3|\mathbf{d}_1, \mathbf{d}_2, M_3'\} = \frac{1}{1} \times \frac{4}{4} \times \frac{2}{2} \times \frac{1}{1} \times \frac{4}{4} \times \frac{4}{4} \times \frac{4}{4} \times \frac{2}{2} = 1. \tag{16.30}$$

Putting together all of the terms, we get

$$Pr\{\mathbf{d}_1, \mathbf{d}_2, \mathbf{d}_3, \mathbf{d}_4|M_3'\} \approx 0.00503 \times 0.00503 \times 1 \times 0.0111 \approx 2.81 \times 10^{-7}. \tag{16.31}$$

Thus, this new model M_3' has the highest likelihood of the three we have considered. We could repeat the analysis above for every possible model of the four genes G1–G4 and thereby find the maximum likelihood model.

2.4 Generalizing to arbitrary numbers of genes

The above examples cover essentially all of the complications we would encounter in evaluating the likelihood of any network model for these genes or any set of genes for the present level of abstraction. In particular, if we understand the three examples in the preceding section, we understand all of the concepts we need to evaluate networks of arbitrary complexity, at least at a simple level of abstraction. We will now see how to complete the generalization to arbitrary numbers of genes.

Suppose now that instead of four genes assayed in eight conditions, we have n genes assayed in m conditions. We can then represent our input matrix D as the set of vectors $\mathbf{d}_1, \ldots, \mathbf{d}_n$, each of length m. Any given model M will still have the form (V, E, P), where $V = \{v_1, \ldots, v_n\}$ now contains one element for each of the n genes and $E \subset V \times V$, i.e. the set of edges is a subset of the set of pairs of genes. (In reality, E will generally be much smaller than $V \times V$ due to the restriction that the graph does not contain directed cycles.) Defining P is a bit more complicated, as we require one probability parameter for each gene, each possible expression level of that gene, and each possible expression level of each of its regulators. More formally, for any given gene i regulated by a set of genes $R_i = \{j|(v_j, v_i) \in E\}$ (read as "the set of values j such that (v_j, v_i) is in set E") of size $m_i = |R_i|$ (the number of elements in set R_i), we require a model variable $p_{i,b_i,b_{i1},\ldots,b_{im_i}}$ for each $b_i, b_{i1}, \ldots, b_{im_i} \in \{0, 1\}$. This results in a set of 2^{m_i+1} parameters in P for gene i defining the probability of each possible state of gene i given each possible state of the genes that regulate it. Collectively, these sets $p_{i,b_i,b_{i1},\ldots,b_{im_i}}$ over all genes i define the probability parameter set P. We can find the maximum likelihood estimate for each such parameter $p_{i,b_i,b_{i1},\ldots,b_{im_i}}$, just as we did in the previous cases, by finding the observations in which genes $i_1, \ldots, i_{m_i}$ have values $b_{i_1}, \ldots, b_{i_{m_i}}$ and determining the fraction of those observations for which gene i has value b_i.

Evaluating the probability of an input matrix D given any particular model $M = (V, E, P)$ then follows analogously to the derivations for fixed n in the preceding sections. We can evaluate the likelihood of any particular expression vector $\mathbf{d}_i$ given the model M and the remaining expression matrix $D/\mathbf{d}_i = [\mathbf{d}_1, \mathbf{d}_2, \ldots, \mathbf{d}_{i-1}, \mathbf{d}_{i+1}, \ldots, \mathbf{d}_n]$ (i.e. the portion of D remaining when we remove $\mathbf{d}_i$) by taking the product over the probabilities of the observed output values:

$$Pr\{\mathbf{d}_i|D/\mathbf{d}_i, M\} = \prod_{j=1}^{m} p_{i,d_{ij},d_{r_{i1},j},\ldots,d_{r_{im_i},j}} \tag{16.32}$$

where the indices $r_{i1}, \ldots, r_{im_i}$ come from the set R_i of inputs to gene i. While the notation gets complicated, intuitively this product simply expresses the idea that we can evaluate the probability of the gene's observed output vector by multiplying independent contributions from each condition.

Similarly, we can accumulate the likelihood function across all output genes i to get the full likelihood of input data D given model M:

$$Pr\{D|M\} = \prod_{i=1}^{n} Pr\{\mathbf{d}_i|D/\mathbf{d}_i, M\} = \prod_{i=1}^{n}\prod_{j=1}^{m} p_{i,d_{ij},d_{r_{i1},j},\ldots,d_{r_{im_i},j}} \tag{16.33}$$

where the r_{ik} values are again drawn from the set R_i. While the notation is again complex, the concept is simple. We can evaluate the probability of the entire data set by accumulating a product across all data points, evaluating each data point by the conditional probability of its observed value given the observed values of all of its input genes. Manually evaluating the likelihood of such a model for more than a few variables would be tedious but it is easily handled by a computer program.

3 Finding the best model

The astute reader might notice that we have not yet mentioned any algorithms in this chapter. We know how to compare different models, but we may have a very large number of possible models to consider. Finding the best of all possible models will therefore require a more sophisticated approach than simply evaluating the likelihood for every possibility and picking the best one. Finding the best of all possible models is an example of a machine learning problem. Machine learning problems like this are very different from standard discrete algorithm problems in that we do not generally have a library of problem-specific algorithms with definite run times from which to draw. Rather, there are a host of generic learning methods that work broadly for problems posed with this sort of probabilistic model. Solving a machine learning problem often involves selecting some such generic algorithm and then tuning it to work especially well given the details of the particular inference being conducted. Actually solving real-world versions of the regulatory network inference problem is not trivial and requires expertise in statistics and machine learning beyond what we assume for readers of this text. In this section, though, we will very briefly consider some general strategies we can use to find a reasonable solution in practice.

For relatively small data sets, a variety of simple solutions are available. For the simplest instances of such a problem, one can try a brute-force search of all possible solutions. The four-gene example we considered, for instance, has a few thousand possible models and we could run through all of them in a reasonable time, evaluating the likelihood of each and finding the global maximum likelihood model. We could extend that brute-force approach to perhaps five or six genes, but not much farther. One alternative for larger networks is to use a *heuristic*, which is a method that provides no guarantees of good performance but tends to give at least a pretty good answer in a reasonable amount of time in practice. One such heuristic strategy is *hill-climbing*. With a hill-climbing heuristic, we start with an initial guess as to the network (perhaps assuming no regulation or using a best guess derived from the literature) and then pick a random potential edge to examine. If that edge is present in the network, we remove it, and if it is not present, we add it. We then evaluate the likelihoods of both the original and the modified networks; whichever network has a higher score is retained. (Note that if we wish to keep the restriction that the network has no cycles then we must test for cycles after each proposed change and assign likelihood zero to any network that has a cycle.) This process continues until we find a network whose likelihood cannot be improved by adding or removing any single edge. Many other generic optimization heuristics like hill-climbing can also be adapted to this problem.

There are also various heuristics specific to the network inference problem. For example, the guilt-by-association (GBA) method [3] suggests that we shrink the universe of possible models by only allowing edges between genes when there is a strong correlation between those genes' expression vectors. This improvement greatly reduces the search space of possible models and allows us to extend other optimization heuristics to much larger gene sets.

For more challenging data sets, a standard approach is to use a *Markov chain Monte Carlo method*, which is essentially a randomized version of the hill-climbing approach. The most widely used such method is the Metropolis–Hastings algorithm [4]. With a Metropolis–Hastings approach to the network inference problem, we can begin just as with hill–climbing, choosing a random edge and creating a version of the model in which that one edge is added if it was not present or removed if it was present. We then again evaluate the likelihood of the model in the original form, which we will call L_1, and in the modified form, which we will call L_2. If $L_2 > L_1$ then we make the change, just as with hill-climbing. If, however, $L_2 < L_1$, we still allow some chance of making the change, with probability L_2/L_1. While this may seem like a minor difference, it actually makes for a far more useful algorithm. We can use this Metropolis–Hastings approach to explore possible models and pick the best, but it also gives us quite a bit of useful information about distributions of models that we can use to assess confidence in the model chosen or specific features of that model. A similar alternative to Metropolis–Hastings is Gibbs sampling [5], which uses essentially the same algorithm for this problem except that on each step one either keeps the modified model with probability $L_2/(L_1 + L_2)$ or the original model with probability $L_1/(L_1 + L_2)$. There is an enormous literature on more sophisticated variants on Markov chain Monte Carlo methods and such methods are often effective for quite difficult problem instances.

For the most difficult data sets, we are likely to need more advanced methods than we can reasonably cover in this text. There is now a large literature on optimization methods for machine learning to which one can refer for solving the hardest problems. Some references to this literature are provided in the concluding section below.

4 Extending the model with prior knowledge

We have now seen a very basic version of how to evaluate possible models of a regulation of a genetic regulatory network, but what we have seen so far is still not likely to lead to accurate inferences from real data. There are simply too many possible models and too little data from which to learn them to hope that such a naïve approach will work well. If we want a genuinely useful method, the most important missing piece to our initial approach is some way of using what is already known or suspected about the system to constrain our inferences. This sort of external knowledge about a problem is generally encoded in a *prior probability*, also known simply as a *prior*. A prior probability is an estimate of how plausible we believe a variable or parameter of the model is independent of the data from which we are formally learning the model. It gives us a way to incorporate into our analysis whatever we know, or think we know, about the system being modeled.

To see how one can use a prior probability, let us suppose we already have a general idea of what the network we are inferring looks like. Perhaps we have referred to prior literature on the genes of interest to us and seen several papers reporting that G1 regulates G2 and a single paper reporting that G2 regulates G3. We might, on that basis, have some prior expectation that our model should include those regulatory relationships. Perhaps we decide that we are 90% confident that G1 regulates G2 and 50% confident that G2 regulates G3. We might also have some prior expectation that our network should be sparse, i.e. that most edges for which there is no literature support should not be present. We might then decide on a generic confidence of 10% that any other given regulatory relationship not mentioned in the literature is present. A prior probability gives us a rigorous way of building these estimates into our inferences. For example, let us consider model M'_1 from Section 2.3 with the following likelihood function:

$$Pr\{\mathbf{d}_1, \mathbf{d}_2, \mathbf{d}_3, \mathbf{d}_4|M'_1\} = Pr\{\mathbf{d}_1|M'_1\} \times Pr\{\mathbf{d}_2|M'_1\} \times Pr\{\mathbf{d}_3|M'_1\} \times Pr\{\mathbf{d}_4|M'_1\}. \tag{16.34}$$

We can incorporate our prior expectations into the network inference problem by changing our objective function from the above likelihood to the probability

$$Pr\{\mathbf{d}_1, \mathbf{d}_2, \mathbf{d}_3, \mathbf{d}_4|M'_1\} \times Pr\{M'_1\},$$

where $Pr\{M'_1\}$ is a probability function over possible models that provides an estimate of how intrinsically plausible we believe each model to be independent of the data. To evaluate that prior probability, we need to consider each edge that might be present in M_1'. If we define $\overline{e}$ to mean the event that edge e is not present in the model, then

$$Pr\{M_1'\} = Pr\{\overline{(v_1, v_2)}\} \times Pr\{\overline{(v_1, v_3)}\} \times Pr\{\overline{(v_1, v_4)}\} \times Pr\{\overline{(v_2, v_1)}\} \times Pr\{\overline{(v_2, v_3)}\} \times \cdots \tag{16.35}$$

Since we believe that (v_1, v_2) is present with confidence 90%, we would say $Pr\{\overline{(v_1, v_2)}\} = 1 - 0.9 = 0.1$. Similarly, since we have 50% confidence that (v_2, v_3) is present, $Pr\{\overline{(v_2, v_3)}\} = 1 - 0.5 = 0.5$. For all other edges (v_i, v_j), $Pr\{\overline{(v_i, v_j)}\} = 1 - 0.1 = 0.9$. There are a total of 12 possible edges for models of 4 genes, so

$$Pr\{M_1'\} = 0.1 \times 0.5 \times (0.9)^{10} \approx 0.0174. \tag{16.36}$$

Adding in this prior knowledge, we can revise our estimate of the plausibility of model M_1' to:

$$Pr\{\mathbf{d}_1, \mathbf{d}_2, \mathbf{d}_3, \mathbf{d}_4 | M_1'\} Pr\{M_1'\} \approx 3.00 \times 10^{-9} \times 0.0174 \approx 5.23 \times 10^{-11}. \tag{16.37}$$

We can similarly incorporate this prior knowledge into our consideration of the alternative models. For M_2', we proposed that G1 regulates G2, which we believe with confidence 90%; G2 regulates G3, which we believe with confidence 50%; and that there are no other edges, which we believe each with confidence 90%. Thus, the prior probability for M_2' is

$$Pr\{M_2'\} = 0.9 \times 0.5 \times (0.9)^{10} \approx 0.141 \tag{16.38}$$

and therefore

$$Pr\{\mathbf{d}_1, \mathbf{d}_2, \mathbf{d}_3, \mathbf{d}_4 | M_2'\} Pr\{M_2'\} \approx 0.141 \times 1.00 \times 10^{-7} \approx 1.41 \times 10^{-8}. \tag{16.39}$$

For M_3', we proposed that G1 does not regulate G2, an event we believe has probability 10%; that G1 does regulate G3, which we also believe has probability 10%; that G2 regulates G3, which we believe has probability 50%; and that no other genes regulate one another, which we believe with probability 90% for each such possible edge. Thus, we derive the prior probability

$$Pr\{M_3'\} = 0.1 \times 0.1 \times 0.5 \times 0.9^9 \approx 1.94 \times 10^{-3}. \tag{16.40}$$

Therefore, our complete objective value for that model is

$$Pr\{\mathbf{d}_1, \mathbf{d}_2, \mathbf{d}_3, \mathbf{d}_4 | M_3'\} Pr\{M_3'\} \approx 2.81 \times 10^{-7} \times 1.94 \times 10^{-3} \approx 5.44 \times 10^{-10}. \tag{16.41}$$

By comparing the three models, we can see that adding prior knowledge can substantially change our assessments about the relative merits of the models. We previously concluded that M_3' was the best of the three models we considered. M_3' shows poor agreement with our prior expectations, though, while M_2' shows very good agreement. With this prior knowledge, M_2' now stands out as the best of the models. This kind of use of prior knowledge is one of the most important factors in effectively handling complex model-inference problems in practice. There is an enormous amount of information available in the biological literature and making good use of that information is one of the key features likely to distinguish an accurate from an inaccurate inference.

Even when we lack real knowledge about a problem, some generic prior probabilities can be very helpful in achieving good results. One important special case of this is the use of prior probabilities to penalize model complexity. One might note that before we started considering prior knowledge, the more complicated models we considered generally outperformed the simpler ones. That phenomenon will occur even when the added complexity has no real biological basis because a maximum likelihood model will exploit every chance correlation occurring in the data to achieve a slightly better fit. In model inference, this phenomenon is known as *overfitting* and needs to be controlled. Prior probabilities provide a way to control for overfitting, by allowing us to specifically penalize more complicated models. Our decision above to assign a 10% prior probability to regulatory edges for which there was no prior evidence is a crude example of an anti-complexity prior. That assumption will tend to favor models having fewer regulatory relationships unless those additional relationships lead to significant improvements in the likelihood of the data being generated from the model. Some more mathematically principled ways to set an anti-complexity prior have also been developed. One such method is the Bayesian information criterion (BIC) [6], in which we set the prior probability of each inferred edge to be the inverse of the number of observed data points. Thus, we would penalize each edge by a factor of $1/8$ in our example.

5 Regulatory network inference in practice

We have now covered the major concepts one needs in order to pose and solve a basic version of the regulatory network inference problem, but there are still quite a few details that separate the methods above from the methods likely to be encountered in the current scientific literature. In this section, we will briefly consider a few extensions of the problem that will bring it much closer to those in use for challenging problem instances in practice. We will first consider how we can drop the assumption of discretization we made at the beginning of the chapter, making full use of real-valued expression data. We will then examine how the model can be extended to allow for additional sources of data beyond gene expression levels, as is commonly done in practice. While we cannot cover these extensions in detail, we can see how these

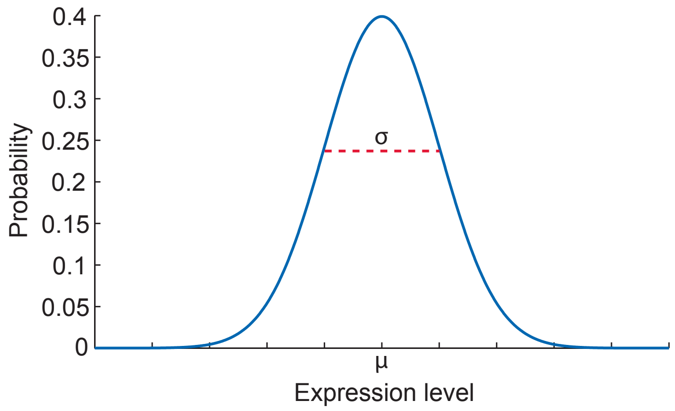

Figure 16.6 Example of a Gaussian curve commonly used as a model of real-valued expression data.

seemingly large changes to the problem actually follow straightforwardly from the principles we have already covered.

5.1 Real-valued data

One of the most dramatic simplifications we made in our toy model was the decision to discretize the data, taking data that are generally real-valued and converting them to binary active/inactive data. It is a minor change to use a more complex discretization – for example, having three labels to represent normal, overexpressed, and underexpressed genes – and we should be able to work out how to extend the concepts we have already covered to any discretized data set. It is possible, however, to work directly with continuous data by adding an assumption about the probability distributions from which data are generated.

It is common to assume that data are *normally distributed*, i.e. described by a Gaussian bell curve as in Figure 16.6. This curve is one example of a probability density function, which describes how likely it is for a given random variable to take on any given possible value. The density curve is highest around the value μ, indicating that the random variable will often be near μ, and is low for values far from μ, indicating that the random variable will rarely be much higher or lower than μ. For a Gaussian random variable, the peak value μ is the average value of the random variable, also known as its *mean*. The width of the bell is controlled by a parameter called its standard deviation (denoted σ). The Gaussian probability density is described by the function

$$Pr\{G = g\} = \frac{1}{\sqrt{2\pi}\sigma} e^{-(g-\mu)^2/(2\sigma^2)} \tag{16.42}$$

where G is the random variable (e.g. expression of gene G1) and g is a particular instance of that random variable (e.g. expression of gene G1 in condition C2).

We can convert our discretized approach above into an approach for real-valued data by using that Gaussian function in place of our previous discrete probability parameters. That is, if we know that the actual real expression value measured by the microarray for some gene i has mean μ_i and standard deviation σ_i, then we can say a given observed value d_{ij} of that gene has likelihood

$$Pr\{d_{ij}|M\} = \frac{1}{\sqrt{2\pi}\sigma_i} e^{-(d_{ij}-\mu_i)^2/(2\sigma_i^2)}. \tag{16.43}$$

The likelihood of a full expression vector $\mathbf{d}_i$ over m different conditions would then be given by

$$Pr\{\mathbf{d}_i|M\} = \prod_{j=1}^{m} \frac{1}{\sqrt{2\pi}\sigma_i} e^{-(d_{ij}-\mu_i)^2/(2\sigma_i^2)}. \tag{16.44}$$

To evaluate this likelihood for a specific data set, though, we need to know μ_i and σ_i. For a gene with no regulators, we will commonly pre-normalize the expression vector $\mathbf{d}_i$ by the formula

$$\hat{d}_{ij} = (d_{ij} - \mu_i)/\sigma_i, \tag{16.45}$$

which will produce a vector of $\hat{d}_{ij}$ values with mean 0 and standard deviation 1. We can then use this normalized vector in place of the raw d_{ij} values. For regulated genes, we will generally assume that μ is a function of the expression levels of its regulators. The most common assumption is that the mean μ_{ij} of a regulated gene i in condition j is a linear function of the expression levels of its regulators in that condition. That is, if we have a gene i regulated by genes $1, \ldots, k$, then we would assume that

$$\mu_{ij} = a_{i1}d_{1j} + a_{i2}d_{2j} + \ldots + a_{ik}d_{kj} \tag{16.46}$$

where each a_{ij} value is a constant that is part of our model.

Finding the maximum likelihood set of a_{ij} values is known as a *regression* problem, and specifically a *linear regression* problem for a linear model like that above. In the interest of space, we will not attempt to explain regression here, only note that finding the maximum likelihood a_{ij} values is a problem we can solve with some basic linear algebra.

5.2 Combining data sources

Another big difference between our toy model above and a real-world method is that an effective method in practice is likely to make use of far more data than just gene expression levels.

Some data sets will inherently have additional information we might use to improve the model. For example, if the data come from experiments at different points in time, we may be able to make a more effective model by assuming expression is a function of time. If the data come from samples subjected to drug treatments, then we may get a more accurate inference by assuming expression is a function of the concentration of drug applied to a given sample. More complicated models are often needed, specialized to the specific kind of data available, but the basics of evaluating and learning those models are not substantially different from what we covered above.

Making accurate predictions will often involve reference to an entirely different data set than the expression data we considered above. For example, we may have DNA sequence data available for the promoters of our genes, which we can examine for likely transcription factor binding sites. We may have direct experimental measurements of which transcription factors bind to which genes. We could treat such data as prior knowledge, building it into our model priors in an *ad-hoc* fashion. A more general approach, however, is to extend the likelihood model to account for multiple experimental measures.

To illustrate this approach, suppose that in addition to the expression data D, we also have a matrix of binding data B, in which an element b_{ij} is 1 if the product of gene i is reported to bind to the promoter of gene j. We can augment our prior likelihood formula for the expression data D to create one evaluating the model as a source for both D and B. If we assume the expression and binding data are independent outputs of a common model, then we can say

$$Pr\{D, B|M\}Pr\{M\} = Pr\{D|M\}Pr\{B|M\}Pr\{M\}.$$

We can evaluate $Pr\{D|M\}$ and the model prior $Pr\{M\}$ just as before.

The same concepts we used to derive a probabilistic model of D can then be used to derive a probabilistic model of B. To account for the possibility of errors in B, we can propose that data in B is a probabilistic function of the regulatory relationships in M. We can use four probability parameters to capture the possible relationships between B and M: $p_{b,0,0}$, the probability B reports no binding given that there is no binding; $p_{b,0,1}$, the probability B reports no binding given that there is binding; $p_{b,1,0}$, the probability B reports binding given that there is no binding; and $p_{b,1,1}$, the probability B reports binding given that there is binding. These four parameters would then augment the probability parameters P for our model $M = (V, E, P)$. Given some such model M we can then say:

$$Pr\{B|M\} = (p_{b,0,0})^{n_{0,0}} (p_{b,0,1})^{n_{0,1}} (p_{b,1,0})^{n_{1,0}} (p_{b,1,1})^{n_{1,1}} \tag{16.47}$$

where $n_{0,0}$ is the number of pairs of genes i and j for which $b_{ij} = 0$ and $(v_i, v_j) \notin E$, $n_{0,1}$ is the number of pairs of genes i and j for which $b_{ij} = 0$ and $(v_i, v_j) \in E$, $n_{1,0}$ is the number of pairs of genes i and j for which $b_{ij} = 1$ and $(v_i, v_j) \notin E$, and $n_{1,1}$ is the number of pairs of genes i and j for which $b_{ij} = 1$ and $(v_i, v_j) \in E$.

The same general ideas can be extended to much more complicated data sets. We can similarly add in any other independent data sources we want by adding an additional multiplicative term to the likelihood for each such data source. Matters get somewhat more complicated if we assume that some data sources are related to one another; for example, if we want to combine two different measures of gene expression. In such cases, we cannot assume distinct measures are independent of one another and therefore cannot simplify our likelihood functions as easily. Nonetheless, similar concepts and methods to those covered above will still apply even if the likelihood formulae are somewhat more complicated.

DISCUSSION AND FURTHER DIRECTIONS

We conclude this chapter with a brief summary and a discussion of where interested readers can go to learn more about the topics covered here. We have seen in this chapter how one can reason about the problem of regulatory network inference. Starting with a simple variant of the problem, we have seen how one can take the real biological problem and abstract it into a precise mathematical framework. In particular, we explored how maximum likelihood inference can be used to frame the regulatory network inference problem. We have further seen some basic methods one can use to find optimal models for that framework. We have, finally, seen how we can take this initial simplified view of the problem and extend it to yield sophisticated models that are not far from those used in practice for difficult real-world network inference problems.

In the process of learning a bit about how regulatory network inference is solved, we have also encountered some of the major paradigms by which computational biologists today think about hard inference problems in general. For example, we saw how to reason about model design, and in particular how one can think about the issue of abstraction in modeling and the kinds of trade-offs different abstractions involve. We saw how probabilistic models, and likelihood models in particular, can provide a general framework for inferring complex models from large, noisy data sets. In the process, we saw an example of how one conceptualizes a problem through the lens of machine learning, for example through reasoning about prior probabilities. These basic concepts in posing and solving for models of large data sources are central to much current work in high-throughput and systems biology. It does not take much imagination to see how the same basic ideas can apply to many other inference problems in biology.

In the space of one chapter, we can only receive a brief exposure to the many techniques upon which the regulatory network inference problem draws; we will therefore conclude with a short discussion of where the interested reader can learn more about the issues discussed here. The specific problem of analyzing gene expression microarrays has been intensively studied and several good texts are available. The beginning reader might refer to Causton *et al.* [7] while those looking for a more advanced treatment might refer to Zhang [8]. More generally, though, the methods described here are fundamental to the fields of statistical inference and machine learning; anyone looking to do advanced work in computational biology would be well advised to seek a strong grounding in those areas. There are numerous texts to which one can refer for statistics training. Wasserman [9, 10] provides a very readable introduction for the beginner. Mitchell [11] provides an excellent introduction to the fundamentals of machine learning and Hastie *et al.* [12] to more advanced topics in statistical machine learning. The specific kind of model we covered here is known as a Bayesian model (or Bayesian network model or Bayesian graphical model). There are many treatments one can reference on that class of statistical model specifically, such as Congdon [13], Gelman *et al.* [14], and Neapolitan [15]. We largely glossed over here the details of algorithms for solving for difficult Bayesian models. The above texts will provide more in-depth coverage of the general algorithmic techniques outlined above. For a deeper coverage of Markov chain Monte Carlo methods, one may refer to Gilks *et al.* [16]. We did not provide any coverage here of more advanced methods in optimization, an important area of expertise for those working on state-of-the-art methods. Optimization is a big field and no one text will do the whole area justice, but those looking for training on advanced optimization might consider Ruszczyński [17] and Boyd and Vandenberghe [18]. Curious readers may also refer to the primary scientific literature for seminal papers that introduced some of the major concepts sketched out there [19, 20].

QUESTIONS

(1) Construct a graph describing the regulatory relationships among four genes, one of which is the sole regulator of the other three.

(2) Provide a likelihood function for regulation of the genes described in Question 1.

(3) How might we change a likelihood function to model a more error-prone expression data source versus a less error-prone expression data source?

(4) How would we need to modify the likelihood function for expression of a single unregulated gene if we assume three different expression levels (high, medium, and low) instead of two (on and off)?

REFERENCES

[1] N. Guelzim, S. Bottani, P. Bourgine, and F. Képès. Topological and causal structure of the yeast transcriptional regulatory network. *Nature Genet.*, 31:60–63, 2002.

[2] National Human Genome Research Institute. Image provided for free public use through the US National Institutes of Health Image Bank as NHGRI press gallery photo 20018.

[3] M. B. Eisen, P. T. Spellman, P. O. Brown, and D. Botstein. Cluster analysis and display of genome-wide expression patterns. *Proc. Natl. Acad. Sci. U S A*, 95:14,863–14,868, 1998.

[4] N. Metropolis, A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller, and E. Teller. Equation of state calculation by fast computing machines. *J. Chem. Phys.*, 21:1087–1092, 1953.

[5] S. Geman and D. Geman. Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images. *IEEE Trans. Pattern Anal. and Machine Intell.*, 6:721–741, 1984.

[6] G. E. Schwarz. Estimating the dimension of a model. *Ann. Stat.*, 6:461–464, 1978.

[7] H. Causton, J. Quackenbush, and A. Brazma. *Microarray Gene Expression Data Analysis: A Beginner's Guide*. Blackwell Science, Malden, MA, 2003.

[8] A. Zhang. *Advanced Analysis of Gene Expression Microarray Data*. World Scientific Publishing, Toh Tuck Link, Singapore, 2006.

[9] L. Wasserman. *All of Statistics*. Springer, New York, 2004.

[10] L. Wasserman. *All of Non-Parametric Statistics*. Springer, New York, 2006.

[11] T. M. Mitchell. *Machine Learning*. WCB/McGraw-Hill, Boston, MA, 1997.

[12] T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer-Verlag, New York, 2001.

[13] P. Congdon. *Applied Bayesian Modelling*. John Wiley and Sons, Chichester, 2003.

[14] A. Gelman, J. B. Carlin, H. S. Stern, and D. B. Rubin. *Bayesian Data Analysis*. CRC Press, Boca Raton, FL, 2004.

[15] R. E. Neapolitan. *Learning Bayesian Networks*. Pearson Prentice Hall, Upper Saddle River, NJ, 2004.

[16] W. R. Gilks, S. Richardson, and D. J. Spiegelhalter. *Markov Chain Monte Carlo in Practice*. Chapman and Hall/CRC, Boca Raton, FL, 1996.

[17] A. Ruszczyński. *Nonlinear Optimization*. Princeton University Press, Princeton, NJ, 2006.

[18] S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, New York, 2004.

[19] P. Dhaseleer, S. Liang, and R. Somogyi. Genetic network inference: From co-expression clustering to reverse engineering. *Bioinformatics*, 16:707–726, 2000.

[20] N. Friedman, M. Linial, I. Nachman, and D. Pe'er. Using Bayesian networks to analyze expression data. *J. Comp. Biol.*, 7:601–620, 2000.