Modeling the Activity of Single Genes[1]

Michael Andrew Gibson[2] and Eric Mjolsness[3]

1 Introduction

1.1 Motivation – the challenge of understanding gene regulation

The central dogma of molecular biology states that information is stored in DNA, transcribed to messenger RNA (mRNA) and then translated into proteins. This picture is significantly augmentated when we consider the action of certain proteins in regulating transcription. These *transcription factors* provide a feedback pathway by which genes can regulate one another's expression as mRNA and then as protein.

To review: DNA, RNA and proteins have different functions. DNA is the molecular storehouse of genetic information. When cells divide, the DNA is replicated, so that each daughter cell maintains the same genetic information as the mother cell. RNA acts as a go-between from DNA to proteins. Only a single copy of DNA is present, but multiple copies of the same piece of RNA may be present, allowing cells to make huge amounts of protein. In eukaryotes (organisms with a nucleus), DNA is found in the nucleus only. RNA is copied in the nucleus then translocates(moves) outside the nucleus, where it is transcribed into proteins. Along the way, the RNA may be spliced, i.e., may have pieces cut out. RNA then attaches to ribosomes and is translated to proteins. Proteins are the machinery of the cell – other than DNA and RNA, all the complex molecules of the cell are proteins. Proteins are specialized machines, each of which fulfils its own task, which may be transporting oxygen, catalyzing reactions, or responding to extracellular signals, just to name a few. One of the more interesting functions a protein may have is binding directly or indirectly to DNA to perform transcriptional regulation, thus forming a closed feedback loop of gene regulation.

The structure of DNA and the central dogma were understood in the 50s; in the early 80s it became possible to make arbitrary modifications to DNA and use cellular machinery to transcribe and translate the resulting genes; more recently, genomes (i.e., the complete DNA sequence) of many organisms have been sequenced. This large-scale sequencing began with simple organisms, viruses and bacteria, progressed to eukaryotes such as yeast, and more recently (1998) progressed to a multi-cellular animal, the nematode *Caenorhabditis elegans*. Sequencers have now moved on to the fruit fly *Drosophila melanogaster*, whose sequence is slated for completion by the end of 1999. The human genome project is expected to determine the complete sequence of all 3 billion bases of human DNA within the next five years. In the wake of genome-scale sequencing, further instrumentation is being developed to assay gene expression and function on a comparably large scale.

Much of the work in computational biology focuses on computational tools used in sequencing, finding genes that are related to a particular gene, finding which parts of the DNA code for proteins and which do not, understanding what proteins will be formed from a given length of DNA, predicting how the proteins will fold from a one-dimensional structure into a three dimensional structure, and so on. Much less computational work has been done regarding the function of proteins. One reason for this is that different proteins function very differently, and so

[1] Supported in part by ONR grants N00014-97-1-0293 and N00014-97-1-0422, the NASA Advanced Concepts program, and by a JPL-CISM grant.

[2] Department of Computation and Neural Systems, Caltech 136-93, Pasadena, CA 91125; gibson@paradise.caltech.edu

[3] Machine Learning Systems Group 126-347, Jet Propulsion Laboratory, California Institute of Technology, 4800 Oak Grove Drive, Pasadena, CA 91109; mjolsness@jpl.nasa.gov

work on protein function is very specific to certain classes of proteins. There are, for example, proteins such enzymes that catalyze various intracellular reactions, receptors that respond to extracellular signals and ion channels that regulate the flow of charged particles into and out of the cell. In this chapter, we will consider a particular class of proteins called *transcription factors(TFs)*, which are responsible for regulating *when* a certain gene is expressed in a certain cell, *which cells* it is express in, and *how much* is expressed. Understanding these processes will involve developing a deeper understanding of transcription, translation, and the cellular processes that control those processes. All of these elements fall under the aegis of *gene regulation* or more narrowly *transcriptional regulation.*

Some of the key questions in gene regulation are: *What genes are expressed in a certain cell at a certain time? How does gene expression differ from cell to cell in a multicellular organism? Which proteins act as transcription factors, i.e., are important in regulating gene expression?* From questions like these, we hope to understand which genes are important for various macroscopic processes.

Nearly all of the cells of a multicellular organism contain the same DNA. Yet this same genetic information yields a large number of different cell types. The fundamental difference between a neuron and a liver cell, for example, is which genes are expressed. Thus understanding gene regulation is an important step in understanding development. Furthermore, understanding the usual genes that are expressed in cells may give important clues about various diseases. Some diseases, such as sickle cell anemia and cystic fibrosis, are caused by defects in single, non-regulatory genes; others, such as certain cancers, are caused when the cellular control circuitry malfunctions - an understanding of these diseases will involve pathways of multiple interacting gene products.

There are numerous challenges in the area of understanding and modeling gene regulation. First and foremost, biologists would like to develop a deeper understanding of the processes involved, including which genes and families of genes are important, how they interact, etc. From a computation point of view, there has been embarrassingly little work done. In this chapter there are many areas in which we can phrase meaningful, non-trivial computational questions, but questions that have not been addressed. Some of these are purely computational (what is a good algorithm for dealing with a model of type X) and others are more mathematical (given a system with certain characteristics, what sort of model can one use? How does one find biochemical parameters from system-level behavior using as few experiments as possible?). In addition to biological and algorithmic problems, there is also the ever-present issue of theoretical biology – what general principles can be derived from these systems, what can one do with models other than just simulate time-courses, what can be deduced about a class of systems without knowing all the details? The fundamental challenge to computationalists and theorists is to add value to the biology – to use models, modeling techniques and algorithms to understand the biology in new ways.

2 Understanding the biology

There are many processes involved in gene regulation. In this section, we discuss some of the most important and best understood processes. In addition to biological processes, there are numerous physical processes that play a role in gene regulation in some systems. Additionally, we shall discuss some of the key differences between prokaryotes and eukaryotes, which are important for including details in models, and also for determining which kind(s) of models are appropriate.

2.1 Biochemical Processes

The biological process of gene expression is a rich and complex set of events that leads from DNA through many intermediates to functioning protein. The process starts with the DNA for a given gene. Recall that in eukaryotes, the DNA is located in the nucleus, whereas prokaryotes have no nucleus, so the DNA may inhabit any part of the cell.

2.1.1 The Basics of Transcription

Transcription is a complicated set of events that leads from DNA to messenger RNA (mRNA). Consider a given gene, as shown in Figure 1a. The gene consists of a coding region and a regulatory region. The coding region is the part of the gene that encodes a certain protein, i.e., this is the part that will be transcribed into mRNA and

translated into a finished protein. The regulatory region is a part of the DNA that contributes to the control of the

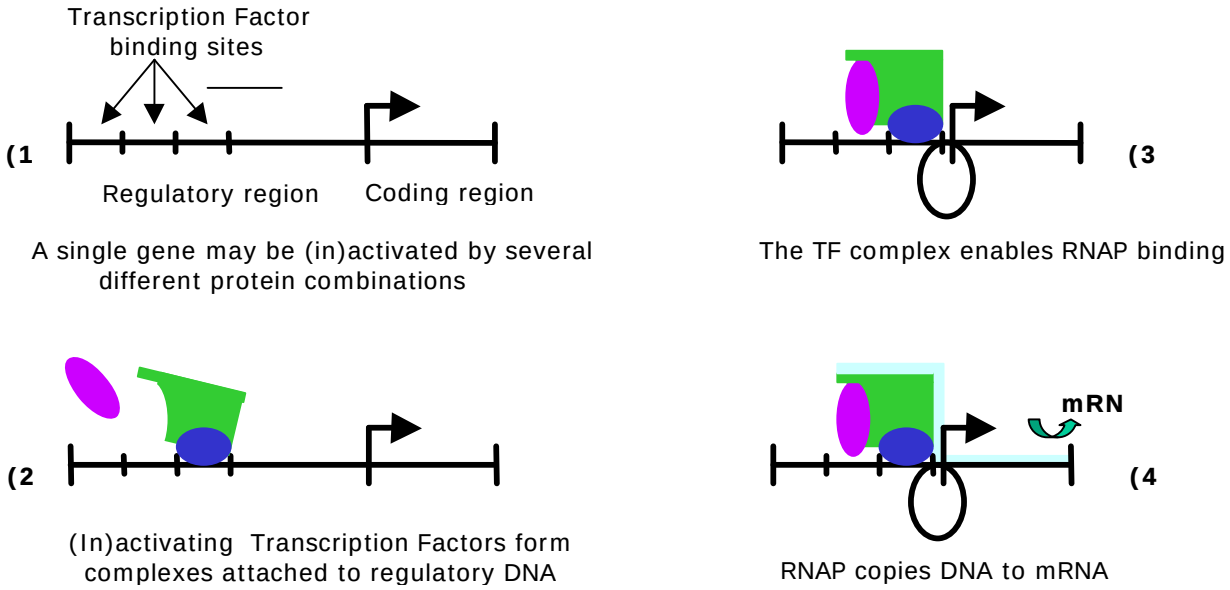

Figure 1 – Processes involved in gene regulation: transcription factor binding, formation of transcriptional complex, RNA Polymerase binding, and transcription initiation.

gene. In particular, it contains binding sites for transcription factors, which act by binding to the DNA (directly or with other transcription factors in a small complex) and affecting the initiation of transcription. In simple prokaryotes, the regulatory region is typically short (10-100 bases) and contains binding sites for a small number of TFs. In eukaryotes, the regulatory region can be very long (up to 10,000 or 100,000 bases), and contains binding sites for multiple TFs. TFs may act either positively or negatively, i.e., an increase in the amount of transcription factor may lead to either more or less gene expression, respectively. Another input mechanism is phosphorylation or dephosphorylation of a bound TF by other proteins.

Typically, TFs do not bind singly, but rather in complexes as in Figure 1b. Once bound to the DNA, the TF complex allows RNA Polymerase (RNAP) to bind to the DNA upstream of the coding region, as in Figure 1c. RNAP forms a transcriptional complex that separates the two strands of DNA, thus forming an *open complex*, then moves along one strand of the DNA, step by step, and transcribes the coding region into mRNA, as in Figure 1d. The rate of transcription varies depending on experimental conditions (Davidson, Watson *et al.*). See von Hippel for a more detailed discussion of transcription.

A bit of terminology: TFs are sometimes called *trans*-regulatory elements, and the DNA sites where TFs bind are called *cis*-regulatory elements. The collection of *cis*-regulatory elements upstream of the coding region can be called the *promoter* (e.g. Small et al. 92), though many authors (e.g. Alberts et al 94) reserve that term just for the sequence where RNA polymerase binds to DNA and initiates transcription. We will follow the former useage.

2.1.2 Cooperativity

Also, there may be extensive cooperativity between binding sites, even in prokaryotes – for example one dimer may bind at one site and interact with a second dimer at a second site. If there were no cooperativity, the binding at the two sites would be independent; cooperativity tends to stabilize the doubly-bound state. Competition is also possible, particularly for two different transcription factors binding at nearby sites.

2.1.3 Exons & Introns, Splicing

In prokaryotes, the coding region is contiguous, but in eukaryotes, the coding region is typically split up into several parts. Each of these coding parts is called an *exon*, and the parts in between the exons are called *introns* (Figure 2 top). Recall that in eukaryotes, transcription occurs in the nucleus. For both types of organisms, translation occurs in the cytosol. In between transcription and translation, in eukaryotes, the mRNA must be moved physically from inside the nucleus to outside. As part of this process, the introns are edited out, which is called *splicing*. In some cases, there are alternative splicings, i.e., the same stretch of DNA can be edited in different ways to form different proteins (Figure 2 bottom). At the end of the splicing process, or directly after the transcription process in prokaryotes, the mRNA is in the cytosol and ready to be translated.

2.1.4 Translation

In the cytosol, mRNA binds to *ribosomes*, complex macromolecules whose function is to create proteins. A ribosome moves along the mRNA three bases at a time, and each three-base combination, or *codon*, is translated into one of the 20 amino acids (Figure 3). As with transcription, the rate of translation varies depending on experimental conditions (Davidson, Watson *et al.*).

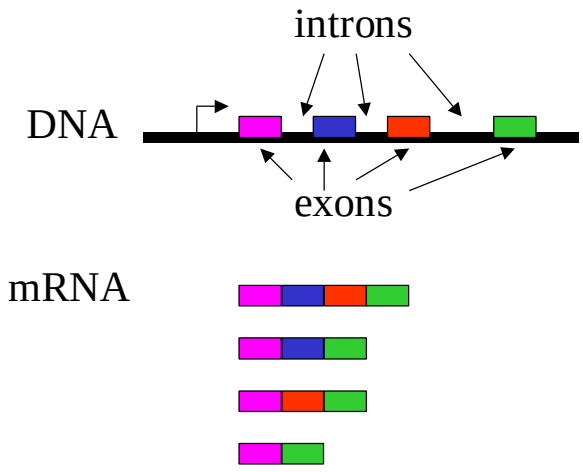

Figure 2– Eukaryotic DNA consists of exons and introns. Exons code for proteins, introns are spliced out of the mRNA. Alternative splicings are possible.

2.1.5 Post-translational modification

The function of the ribosome is to copy the one-dimensional structure of mRNA into a one-dimensional sequence of amino acids. As it does this, the one-dimensional sequence of amino acids folds up into a final three dimensional protein structure. A protein may fold by itself, or it may require the assistance of other proteins, called *chaperones*.
As previously mentioned, the process of protein folding is currently the subject of a large amount of computational work; we shall not discuss it further here.

At this point, we have summarized the flow of information from DNA to a protein. Several more steps are possible. First, proteins may be modified after they are translated. As an example of this post-translational modification, certain amino acids near the N-terminus are frequently cleaved off (Varshavsky, 1995). Even if there is no post-translational modification, proteins may conglomerate, e.g., two copies of the same protein (monomers) may combine to form a homo-dimer. Multimers – complex protein complexes consisting of more than one individual protein - are also common. In particular, many TFs bind to DNA in a multimeric state, e.g., as *homodimers* (two copies of the same monomer) or as *heterodimers* (two different monomers). These same proteins can exist as monomers, perhaps even stably, but only the multimer forms can bind DNA actively.

2.1.6 Degradation

DNA is a stable molecule, but mRNA and proteins are constantly being degraded by cellular machinery and recycled. Specifically, mRNA is degraded by a ribonuclease (RNase), which competes with ribosomes to bind to mRNA. If a ribosome binds, the mRNA will be translated, if the RNase binds, the mRNA will be degraded.

Proteins are degraded by cellular machinery including proteasomes signaled by ubiquitin tagging. Protein degradation is regulated by a variety of more specific enzymes (which may differ from one protein target to another). For multimers, the monomer and multimer forms may be degraded at different rates.

2.1.7 Other mechanisms

Eukaryotic DNA is packaged by complexing with histones and other chromosomal proteins into *chromatin*. The structure of chromatin includes multiple levels of physical organisation such as DNA winding around nucleosomes consisting of histone octamers. Transcriptionally active DNA may require important alterations to its physical organization such as selective uncoiling. Appropriately incorportating this kind of organization, and other complications we have omitted, will pose further challenges to the modeling of gene regulation networks.

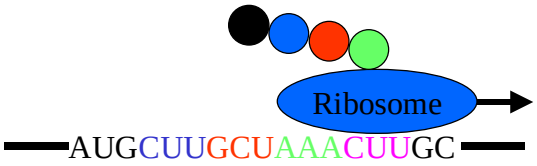

Figure 3– Translation consists of a ribosome moving along the mRNA one codon (three bases) at a time and translating each codon of the mRNA into an amino acid of the final protein.

2.2 Biophysical Processes

In addition to the chemical processes described above, numerous physical processes can be important in gene regulatory systems. One may need to include a detailed model of the physical processes in some systems; in others, these processes may have only minor effects and can be ignored without a significant degradation of model performance.

Most of the models that follow deal with the amount of protein in a cell. In the simplest case, we may be able to assume that the cell is *well-mixed*, i.e., that the amount of protein is uniform across the cell. For more complicated systems, in particular for large systems, that is not a good approximation, and we must consider explicitly the effect of diffusion or of transport. Diffusion will be the most important physical effect in the models we consider, but in other systems active transport could be as important or more important.

Diffusion is a passive spreading out of molecules due to thermal effects. The distance a molecule can be expected to diffuse depends on the square root of time, so molecules can be expected to move small distances relatively quickly (this is why we can ignore this effect in small systems) but will take much longer to diffuse longer distances. The diffusion distance also depends on a constant specific to the molecule and to the solvent. Larger molecules tend to diffuse more slowly than do smaller ones, and all molecules diffuse more slowly in solvents that are more viscous.

In addition to the passive diffusion process, cells have active machinery that moves proteins from one part to another. In humans, for example, some neurons can be up to a meter long, yet the protein synthesis machinery is concentrated in the cell body, the part of the cell where the nucleus is located. Cells have developed an elaborate

system of active transport from the site of protein synthesis to the most distal parts of the cell. It may be necessary to model active transport more completely in some systems than has heretofore been done.

Depending on the type of model and the time scale, cell growth may be an important effect. The rate of a chemical reaction of second or higher order (discussed in Section 3.1) depends on the volume in which the reaction occurs. This consistent change of volume, due to growth, may be an important effect to include in models of some systems.

DNA and most DNA-binding proteins are charged. Some cells, especially neurons, but also muscle cells, heart cells, and many others, change their electrical properties over time. This has no significant effect on gene regulation in any system we will discuss, but that is not to say that it will not have an effect in *any* gene regulatory system.

2.3 Prokaryotes vs. Eukaryotes

The key difference between prokaryotes and eukaryotes is that eukaryotes have a nucleus and prokaryotes do not. That difference belies the difference in complexity between these two types of organisms. Fundamentally, prokaryotes are much simpler organisms – the lack of nucleus is just one example of that. Other differences are in the complexity of the transcription complex, mRNA splicing, and the role of chromatin.

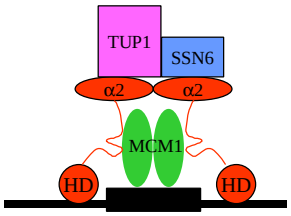

Figure 4– Transcriptional regulation at a-specific gene (asg) operator in budding yeast, for ? and a/? cells. Redrawn from (Johnson 1995). Note multiple protein-protein and protein-DNA interactions in a complex. DNA is at the bottom and includes the asg opera

2.3.1 Transcription

The eukaryotic transcription complex is much more complicated than the prokaryotic one. The latter consists of a small number of proteins that have been isolated, and this small number is sufficient for transcription. For that reason *in vitro* (in a test-tube) measurements of prokaryotes are thought to be more related to *in vivo* (in the living organism) processes than the corresponding measurements in eukaryotes are.

Eukaryotic promoters may have large numbers of binding sites occurring in more or less clustered ways, whereas prokaryotes typically have a much smaller number of binding sites. For N binding sites a full equilibrium statistical mechanics treatment (possibly oversimplified) will have at least 2^N terms in the partition function (discussed later), one for each combination of bound and unbound conditions at all binding sites. The most advantageous way to simplify this partition function is not known, because there are many possible interactions between elements of the transcription complex (some of which bind directly to DNA, some bind to each other). In the absence of all such interactions the partition function could be a simple product of N independent two-term factors, or perhaps one such sum for each of a global “active” and “inactive” state.

The “specific” transcription factors are proteins that bind to DNA and interact with one another in poorly understood ways to regulate transcription of specific, large sets of genes. These protein-protein interactions inside the transcription complex really cloud the subject of building models for eukaryotic gene expression. An example of transcription factor interaction in budding yeast is shown in Figure 4.

A further complication is the "general" transcription factors such as TFIID that assemble at the TATA sequence of eukaryotic transcription complexes, building a complex with RNA Polymerase II which permits the latter to associate with DNA and start transcribing the coding sequence. Finally, signal transduction (e.g. by MAP kinase cascades (Madhani and Fink 1998) may act by phosphorylating constitutively bound transcription factors, converting a repressive transcription factor into an enhancing one.

2.3.1.1 Modules

Transcription factors work by binding to the DNA and affecting the rate of transcription initiation. However, severe complications ensue when interactions with other transcription factors in a large "transcription complex" become important. For simple prokaryotes, it is sometimes possible to write out all possible binding states of the DNA, and to measure binding constants for each such state. For more complicated eukaryotes, it is not. It has been hypothesized that transcription factors have three main functions – some are active in certain cells and not others and provide *positional* control, others are active at certain times and provide *temporal* control, still others are present in response to certain extracellular signals (Figure 5).

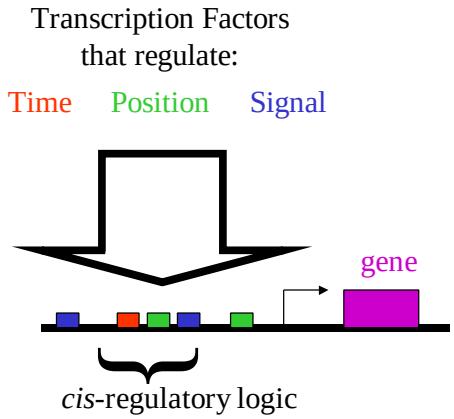

Figure 5– Some transcription factors act at certain times, some in certain cells, and others in response to signals.

Many binding sites occur in spatial and functional clusters called enhancers , promoter elements, or regulatory modules. For example, the 480 base pair *even-skipped (eve)* stripe 2 "minimal stripe element" in *Drosophila* (Small et al. 1992) has five activating binding sites for the *bicoid (bcd)* transcription factor and one for *hunchback (hb)*. It also has three repressive binding sites for each of *giant (gt)* and *Kruppel (Kr)*. The minimal stripe element acts as a "module" and suffices to produce the expression of *eve* in stripe 2 (out of seven *eve* stripes in the developing *Drosophila* embryo). Similar modules for stripes 3 and 7, if they can be properly defined (Small et al. 1996), would be less tightly clustered. These promoter "regions" or "modules" suggest a hierarchical or modular style of modeling the transcription complex and hence single gene expression, such as provided by (Yuh, Bolouri and Davidson) for Endo16 in sea urchin, or the Hierarchical Cooperative Activation model suggested in Section 5 and Chapter 5 (Mjolsness, this volume).

A different way to think about these binding site interactions is provided by (Gray et al .1995) who hypothesize three main forms of negative interaction between sites:

- *competitive binding*, in which steric constraints between neighboring binding sites prevent both from being occupied at once,
- *quenching*, in which binding sites within about 50 base pairs of each other compete, and
- *silencer regions*, promoter regions that shut down the whole promoter when cooperatively activated.

Given these observations, we can see that the biological understanding of eukaryotic cis-acting transcriptional regulation is perhaps ... "embryonic".

2.3.2 Translation

Eukaryotic genes are organized as introns and exons (Figure 2), and *splicing* is the process by which the introns are removed and the mRNA edited so as to produce the correct proteins. There can be alternative splicings, and control elements can be located in introns. There may be a nontrivial role for chromatin state at several length scales. These complications will not be considered further in any of the models discussed in this chapter.

2.4 Feedback and Gene Circuits

We have considered some of the mechanisms by which transcription of a single gene is regulated. A key point we have avoided is *feedback*. Simply stated, *the TFs are themselves subject to regulation*. This leads to interconnected systems that are more difficult to analyze than feed-forward systems. For single variable systems, there are two major kinds of feedback – positive and negative; for multivariable systems, feedback is more complicated.

Negative feedback is the way a thermostat works: when the room gets too hot, the cooling system kicks in and cools it down; when the room gets too cool, the heater kicks in and warms it up. This leads to stabilization about a fixed point. More complicated negative feedback is also possible, which leads to better control. The field of control, both of single input systems and of multiple input systems, is well beyond the scope of this chapter.

Positive feedback can create amplification, decisions, and memory. Suppose your thermostat were wired backwards, in the sense that if the room got too hot, the *heater* would turn on. This would make it even hotter, so the heater would turn on even more, etc., and soon your room would be an oven. On the other hand, if your room got too cold, the air conditioner would kick in, and cool it down even more. Thus, positive feedback would amplify the initial conditions – a small hot temperature would lead to maximum heat, a small cold temperature would lead to maximum cooling. This results in two stable fixed points as final states –very hot and very cold - and a decision between them. Roughly, this is how it is possible for cells to pick one of several alternate fates and to remember its decision amidst thermal noise and stochastic environmental input.

Several models of multiple-gene feedback circuits or networks will be presented in the remainder of this book.

3 Modeling basics

Having described the processes involved in gene regulation, we turn to the question of modeling. This section and the next will show how to create a predictive model from biochemical details, spelled out textually and in pictures as above. We split this process into two parts. First, this section develops the concept of a *calculation independent* model, i.e., a formal, precise, and quantitative representation of the processes of the previous section. The next section describes how to start with a calculation independent model, do calculations, and make predictions about the behavior of the system. There are numerous ways to do the calculation, depending on the assumptions one makes; each set of assumptions has its pros and cons.

There are two reasons why having a calculation independent model is useful. First, biologists can develop a model – a precise representation of the processes involved in gene regulation – without regard to the computational problems involved. For instance, the next section will show certain areas where the computational theory has not been worked out fully; precise description of biological processes should not be held hostage to computational problems. Rather, a precise model should be made, and when computations are to be done, additional assumptions and constraints can be added, which are understood to be *computational* assumptions and constraints, not *biological*. The second use of calculation independent models is that theorists can develop the tools – both computational and mathematical – to deal with all models that fit into this calculation independent framework, rather than ad-hoc methods that apply only to a particular biological system.

The rest of this section introduces the notation of chemical reactions and describes some fundamental ideas underlying physical chemistry: kinetics, equilibrium and the connection to thermodynamics.

3.1 Chemical reactions

Chemical reactions are the lingua franca of biological modeling. They provide a unifying notation in which to express arbitrarily complex chemical processes, either qualitatively or quantitatively. Specifying chemical reactions is so fundamental that the same set of chemical reactions can lead to different computational models, e.g., a detailed differential equations model *or* a detailed stochastic model. In this sense, representing processes by chemical equations is more basic than using either differential equations or stochastic processes to run calculations to make predictions.

A generic chemical reaction, such as:

$$n_a A + n_b B \xrightarrow{k} n_c C + n_d D$$

states that some molecules of type *A* react with some of type *B* to form molecules of types *C* and *D*. The terms on the left of the arrow are called the *reactants*; those on the right are called the *products*. There can be an arbitrary number of reactants and an arbitrary number of products, not just always two, and the number of reactants and products do not have to match.

In the reaction given, n_a molecules of *A* react with n_b molecules of B to give n_c molecules of *C* and n_d of *D*. The *n* terms are called *stoichiometric coefficients* and are small integers. For example, the reaction

$$A \xrightarrow{k} A'$$

has one reactant, *A*, and one product, *A'*; n_a is 1 and $n_{a'}$ is also 1. In other words, this reaction says that one molecule of type *A* reacts to give one molecule of type *A'*. Note that, as in this example, stoichiometric coefficients of 1 are frequently omitted.

In these two examples, the value *k* on the reaction arrow is a *rate constant*. Chemical reactions do not occur instantaneously, but rather, take some time to occur. The value *k* is a way of specifying the amount of time a reaction takes. The rate constant depends on temperature – at a fixed temperature, suppose n_a molecules of *A* collide with n_b molecules of B; the rate constant measures the probability that this collision will occur with sufficient energy for the molecules to react and give the products.

Example:

The process illustrated in Figure 6 is a simple example of TFs binding to DNA (as discussed in Section 2 and in Figure 1). In this particular example, two different proteins, P and Q, can bind to DNA. Using the notation *P•DNA* to mean "*P* bound to *DNA*", etc., the chemical equations for the process are:

$$1P + 1DNA_{free} \xrightarrow{k_{01}} 1P \bullet DNA$$

$$1Q + 1DNA_{free} \xrightarrow{k_{02}} 1Q \bullet DNA$$

$$1P + 1Q \bullet DNA \xrightarrow{k_{23}} 1P \bullet Q \bullet DNA$$

$$1Q + 1P \bullet DNA \xrightarrow{k_{13}} 1P \bullet Q \bullet DNA$$

$$1P \bullet DNA \xrightarrow{k_{10}} 1P + 1DNA_{free}$$

$$1Q \bullet DNA \xrightarrow{k_{20}} 1Q + 1DNA_{free}$$

$$1P \bullet Q \bullet DNA \xrightarrow{k_{32}} 1P + 1Q \bullet DNA$$

$$1P \bullet Q \bullet DNA \xrightarrow{k_{31}} 1Q + 1P \bullet DNA$$

3.2 Physical chemistry

3.2.1 Concept of state

The *state* of a system is a snapshot of the system at a given time that contains enough information to predict the behavior of the system for all future times. Intuitively, the state of the system is the set of variables that must be kept track of in a model. Different models of gene regulation have different representations of the state:

- In Boolean models, the state is a list, for each gene involved, of whether the gene is expressed ("1") or not expressed ("0").
- In differential equations models, the state is a list of the concentrations of each chemical type.
- In stochastic models, a configuration is a list of the actual number of molecules of each type. The state is either the configuration or the current probability distribution on configurations, depending on the particular variant of stochastic model in question.
- In a molecular dynamics model of gene regulation, the state is a list of the positions and momenta of each molecule.

Although the state in each of these models is very different, there are some very fundamental similarities.

- Each model defines what it means by the state of the system.
- For each model, given the current state *and no other information*, the model predicts which state or states can occur next.
- Some states are *equilibrium states* in the sense that once in that state, the system stays in that state.

Physical chemistry deals with two problems: kinetics, i.e., changes of state, and equilibria, i.e., which states are equilibrium states. Amazingly, the latter question can be answered by thermodynamics, just by knowing the energy differences of the different possible states of the system, without knowing the initial state or anything about the kinetics. We shall consider in turn kinetics, equilibria, and thermodynamics, as they apply to gene regulation.

3.2.2 Kinetics – changes of state

The chemical equation

$$A \xrightarrow{k} A'$$

deals with two chemical *species*, *A* and *A'*. According to the equation, *A* is transformed to *A'* at a rate of *k*. In other words, the chemical equation specifies *how* the state of the system changes, and *how fast* that change occurs.

In the differential equations approach, this equation specifies that state changes in the following way: the concentration of *A* decreases and the concentration of *A'* increases correspondingly. For small times *dt*, the amount of the change is given by $k \cdot [A] \cdot dt$ where *[A]* is the concentration of *A*.

In the stochastic approach, this equation specifies that the state changes in a different way: one single molecule of *A* is converted into a molecule of *A'*; the total number of molecules of *A* decreases by one and the total number of molecules of *A'* increases by one. The probability that this event occurs in a small time *dt* is given by $k \cdot \{\#A\} \cdot dt$, where *{#A}* is the number of molecules of *A* present.

3.2.3 Equilibrium

A system is said to be at *equilibrium* when its state ceases to change. Consider the previous example:

$$A \xrightarrow{k} A'$$

In the previous section, we saw that, as long as there is available *A*, the state will change at a rate determined by *k*. Thus, when this system reaches equilibrium, all of the *A* will be gone, and only *A'* will remain. At the equilibrium point, once all the *A* has been used up, there can be no more changes of state.

It is not in general true that equilibrium only occurs when one of the components has been used up. Consider, for example, a certain DNA binding protein X, whose binding to and unbinding from DNA follow simple and complementary chemical kinetics:

$$X + DNA \xrightarrow{k_1} X \bullet DNA \qquad (1)$$

$$X \bullet DNA \xrightarrow{k_{-1}} X + DNA \quad (2)$$

This set of reactions involves chemicals of three types: *X*, *DNA* and *X •DNA*. Equilibrium will occur when the rate at which *X* and *DNA* are converted to *X •DNA* (according to Equation 1) exactly equals the rate at which *X •DNA* is converted to *X* and *DNA* (according to Equation 2). Note that in this equilibrium, there are still changes: both reactions still occur constantly. However, there are no *net* changes. And since the state of the system the list of is (for example) concentrations of all chemical types involved, the state does not change.

The concepts of kinetics and equilibrium are general – they appear in one form or another in all the different models in this chapter. At this point, however, we shall present a detailed example in one particular framework, that of differential equations for reaction kinetics, to illustrate key points. This example will illustrate concepts and ideas that are true not just for differential equations, but also in general.

Example (Equilibrium in the Differential Equations framework):

The two chemical equations above lead to three differential equations, one for the concentration of *X* (denoted [*X*]), one for [*DNA*] and one for [*X •DNA*]. A later section will explain how to derive the differential equations from chemical equations, and in particular, will show that the differential equation for [*X •DNA*] is:

$$\frac{d[X \bullet DNA]}{dt} = k_1[X][DNA] - k_{-1}[X \bullet DNA]$$

Note that the right hand side has two terms: the first one is production of new *X •DNA* due to equation (1), the second is degradation of existing *X •DNA* due to equation (2). With the corresponding equations for [*X*] and [*DNA*], plus the initial concentrations of each of the three, we can solve for [*X •DNA*] as a function of time. Doing so would be the *kinetics* approach to the problem.

Instead, assume the two competing processes have reached *equilibrium*, i.e. there are no further *net* changes. Thus,

$$0 = \frac{d[X \bullet DNA]}{dt} = k_1[X][DNA] - k_{-1}[X \bullet DNA] \quad (3)$$

This leads to the following equation, valid for equilibrium concentrations:

$$\frac{[X \bullet DNA]}{[X][DNA]} = \frac{k_1}{k_{-1}} \equiv K_{eq} \quad (4)$$

Here K_{eq} is called the *equilibrium constant* of this reaction. Notice that applying the equilibrium condition in (3) reduced a differential equation to an algebraic equation. Applying the equilibrium condition in an *arbitrary* framework removes the time-dependence of the kinetic equations, and the resulting equations are algebraic. Now, in addition to the rate constants of the previous subsection, chemical reactions may have an equilibrium constant, which is a property only of the system, not the computational framework of the system. For example, in the stochastic framework, it is possible to define an equilibrium constant just as in (4). Since the stochastic framework typically uses number of molecules, rather than concentration of molecules, the stochastic equilibrium constant is typically reported in different units.

Real physical systems tend toward equilibrium unless energy is continually added. So another interpretation of equilibrium is *the state of the system as time® ¥, in the absence of energy inputs*. To simplify models (and experiments), reactions that are fast (compared to the main reactions of interest) are often assumed to be at equilibrium. Given this assumption, for example, the pair of equations (1) and (2) can be abbreviated as

$$X + DNA \xleftrightarrow{K_{eq}} X \bullet DNA \quad (5)$$

It is possible to go directly from the chemical equation (5) to the algebraic equation (4) by multiplying all the concentrations of the products (in this case only [X*DNA]) and dividing by the product of the concentrations of the reactants (in this case [X][DNA]).

Consider Equation (5) and the additional equilibrium equation:

$$X \bullet DNA \xleftrightarrow{K_{eq2}} X' \bullet DNA \qquad (6)$$

For example, this could mean that X undergoes a conformational change to X' while bound to DNA. Returning to the differential equations framework, the equilibrium equation is:

$$\frac{[X' \bullet DNA]}{[X \bullet DNA]} = K_{eq2} \qquad (7)$$

Notice that

$$\frac{[X' \bullet DNA]}{[X][DNA]} = \frac{[X \bullet DNA]}{[X][DNA]} \frac{[X' \bullet DNA]}{[X \bullet DNA]} = K_{eq} K_{eq2}.$$

This is an important property of equilibria: if A and B are in equilibrium, and B and C are in equilibrium, then A and C are in equilibrium, and the resulting A-C equilibrium constant is simply the product of the A-B and B-C equilibrium constants.

3.2.4 Thermodynamics

Two ways of determining equilibrium constants have been presented thus far: calculating equilibrium constants from the forward and reverse rate constants, and calculating them from the equilibrium concentrations of the chemicals. There is a third way – equilibrium constants can be calculated from thermodynamics, using just the energy difference between the products and reactants. From this energy difference alone, one can predict the *final* state of the system, but not the time-course of the state from initial to final.

For chemical reactions, the *Gibbs free energy*, *G*, is defined by

G = (Total internal energy)– (absolute temperature)×(entropy) +(pressure) × (volume)
= Σ(chemical potential)×(particle number)

Typically, values of ΔG are reported instead of the values of K_{eq}. From thermodynamics in dilute solutions [Hill 1985, Atkins 1998] it follows that, for reactants and products with free energy difference ΔG,

$$K_{eq} = e^{\frac{-\Delta G}{RT}},$$

where R is the ideal gas constant, namely Boltzmann's constant time Avagadro's number, and T is the absolute temperature.

To deal with multiple states in equilibrium with each other, one uses *partition functions*. In general, the fraction of the system in a certain configuration *c* (e.g., the fraction of the DNA bound to protein P) is given by:

$$frac_c = \frac{\exp(-\Delta G_c / RT)[Species_1]^{power_1} \cdots [Species_n]^{power_n}}{\sum_i \exp(-\Delta G_i / RT)[Species_1]^{power_1} \cdots [Species_n]^{power_n}}$$

The power of chemical species *x* in configuration *c* is simply the number of molecules of type *x* present in configuration *c*. The denominator of this equation is the partition function.

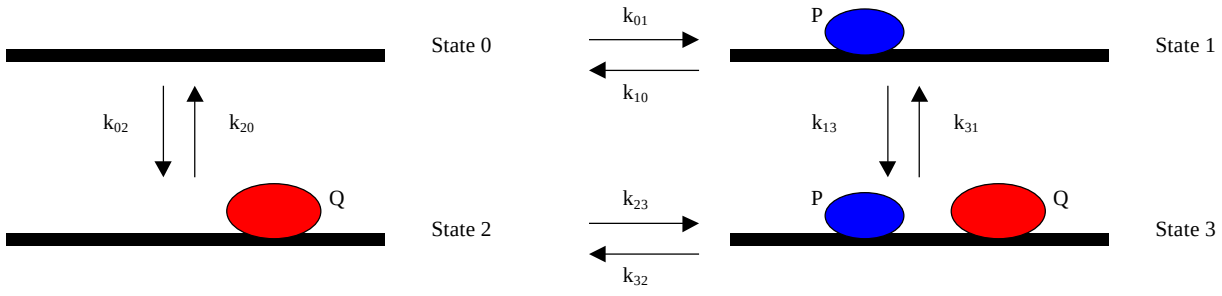

Figure 6– Kinetic model of two proteins, P and Q, binding to DNA.

Example (Partition Functions):

Consider the example in Figure 6, but this time, assume that equilibrium has been reached. Then

$$\frac{[DNA]_0}{[DNA]_{total}} = \frac{1}{Z}, \qquad \frac{[DNA]_1}{[DNA]_{total}} = \frac{K_1[P]}{Z},$$

$$\frac{[DNA]_2}{[DNA]_{total}} = \frac{K_2[Q]}{Z} \quad \text{and} \quad \frac{[DNA]_3}{[DNA]_{total}} = \frac{K_3[P][Q]}{Z},$$

where each of the K's is defined to be the equilibrium constant between a given state and State 0, and

$$Z \equiv 1 + K_1[P] + K_2[Q] + K_3[P][Q].$$

Here Z is the partition function.

Derivation:

Let $[DNA]_0$ be the concentration of DNA in State 0, and let [P] and [Q] be the concentration of free (i.e., unbound) protein P and Q, respectively. Then, there are equilibrium constants K_1, K_2 and K_3, such that at equilibrium

$$\frac{[DNA]_1}{[P][DNA]_0} = K_1, \qquad \frac{[DNA]_2}{[Q][DNA]_0} = K_2 \qquad \text{and} \qquad \frac{[DNA]_3}{[P][Q][DNA]_0} = K_3.$$

The total concentration of DNA is given by

$$[DNA]_{total} = [DNA]_0 + [DNA]_1 + [DNA]_2 + [DNA]_3,$$

which leads to

$$\begin{aligned}[DNA]_{total} &= [DNA]_0 + K_1[P][DNA]_0 + K_2[Q][DNA]_0 + K_3[P][Q][DNA]_0 \\ &= [DNA]_0 Z\end{aligned}$$

The result of the previous example follows directly.

4 Generating predictions from biochemical models

This section will assume that a computationally independent model of a biological process has been created. Such a model, as described in the previous section, may consist of chemical equations, parameter values, and physical constraints such as diffusion and growth. The model is a formal, precise way to describe the biological process, and writing the model is a biological problem. Once the model has been created, as assumed in this section, the remaining computational problem is how to generate predictions from the model.

This section will discuss several different modeling methods. Although this list is by no means complete, it should serve as a good comparison of different types of models and as a jumping-off point for further investigation. There are three main questions to keep in mind when understanding the different types of models:

- What is the *state* in each model, i.e., which variables does one consider?
- How does the state change over time, i.e., what are the kinetic or dynamic properties of the model?
- What are the *equilibrium states* of the model, i.e., which states are stable?

There are also several practical questions to keep in mind, which relate to the *applicability* of the model:

- What assumptions does the model make? Which are biological and which are strictly computational?
  Note that all models make some computational assumptions, so the mere existence of assumptions or approximations should not be grounds for rejecting a model. The specific assumptions made, however, may be.
- For what time scale is the model valid?
  At very short time scales (seconds or less), the low-level details of binding/unbinding and protein conformational changes have to be modeled. At longer time scales, these can be considered to be in equilibrium, and certain average values can be used (this point will be discussed in more detail later). At

very long time scales (e.g. days), processes such as cell division, which can be ignored at shorter time scales, may be very important,.

- How many molecules are present in the biological system?
  If the number of molecules is very small (i.e. in the 10s or low 100s), stochastic models may need to be used. However, once the number of molecules becomes very large, differential equations become the method of choice.
- How complex is the computation resulting from this model?
  The more detail and the longer one wants to model a process for, the higher the complexity, and hence the longer it takes computationally.
- How much data is available?
  For many systems, there is a lot of high-level qualitative data, but less quantitative, detailed data. This is particularly true of complex eukaryotes. The nature of the data required by a model in order to make predictions, an important practical property of the model, may depend on the power of the data-fitting algorithms available.
- What can the model hope to predict? What can it not?
  To begin modeling, one must focus on what types of predictions are sought. For simple predictions, simple models are sufficient. For complex predictions, complex models may be needed.

4.1 Overview of models

Rather than advocating a single, definitive model of gene regulation, this section will describe a variety of modeling approaches that have different strengths, weaknesses, and domains of applicability in the context of the foregoing questions. Note that improvements in modeling precision typically carry a cost; for example they may require more data or more precise data. First we outline the basic modeling approaches, then give a detailed explanation of each, with examples where appropriate.

At one end of the modeling spectrum are *Boolean network* models, in which each gene is either fully expressed or not expressed at all. This coarse representation of the state has certain advantages, in that the next state, given a certain state, is a simple Boolean function; also, the state-space is finite, so it is possible to do brute-force calculations that would be intractable in an infinite state-space. These models are typically used to give a first representation of a complex system with many components, until such time as more detailed data become available.

One step along the spectrum come *kinetic logic* models. These models represent the state of each gene as a discrete value: “not expressed,” “expressed at a low level,” “expressed at a medium level,” or “fully expressed,” for example, thus providing more granularity than in Boolean networks. Additionally, these models attempt to deal with the rates at which the system changes from one state to another. Rather than assuming that all genes change state at the same time (as in Boolean networks), these models allow genes to change state at independent rates. The functions describing the changes of state are more complicated in these models than in Boolean networks, so more data is required to find the functions. In the end, however, the predictions of this type of model are more precise and tie more closely to the biology.

Next in complexity come the continuous logical models, which ties between kinetic logic and differential equations. The model still contains discrete states, but the transition from one state to another is governed by linear differential equations with constant coefficients. These differential equations allow more modeling precision, but again, require more detailed data.

Differential equation models provide a general framework in which to consider gene regulation processes . By making certain assumptions, one can transform essentially any system of chemical reactions and physical constraints into a system of non-linear ordinary differential equations (ODE’s), whose variables are concentrations of proteins, mRNA, etc. One way to do this is by reaction kinetics (or enzyme kinetics), considering the transition rates between all microscopic states. When the number of states becomes too large or poorly understood, as for example in protein complexes, coarser and more phenomenological ODE systems may be postulated. Either way the equations can be solved numerically for their trajectories, and the trajectories analyzed for their qualitative

dependence on input parameters. There are also many different ways to approximate the non-linear differential equations, some of which are advantageous for parameter estimation, for example, as discussed in the next section.

Not all systems can be modeled with differential equations. Specifically, differential equations assume that changes of state are continuous and deterministic. Discontinuous transitions in deterministic systems can be modeled with various hybrids between discrete and continuous dynamics, including continuous-time logic, special kinds of grammars, and Discrete Event Systems. Non-deterministic systems, systems where the same state can lead to different possible outcomes, generally must be modeled in a different framework. One such framework, borrowed from physics, is the *Langevin* approach. Here, one writes a system of differential equations as before, then adds a noise term to each. The noise term is a random function. For a *particular* noise function, one may just solve the differential equation as before. Given the *statistics* of the noise function, one may make statements about the *statistics* of a large number of systems, i.e., which outcomes occur with which probabilities.

Another approach to overcoming the limitations of differential equations is the *Fokker-Planck* method. Here, one starts from a probabilistic framework and writes a full set of equations that describe the change in *probability distribution* as a function of time. One still assumes continuity, i.e., that the probability distribution is a continuous function of concentration. This leads to a certain partial differential equation of the probability distribution (partial in time and in concentration). One can solve this numerically – although partial differential equations are notoriously harder to solve than ordinary differential equations – to get full knowledge of the probability distribution as a function of time.

Both the Langevin and Fokker-Planck equations are approximations of the fully stochastic model to be considered later. Under certain conditions, the approximations are very good, i.e., the difference between approximation and exact solution is much smaller than the variance of the approximation. Under other conditions, the difference may be on the order of the variance, in which case these models do not make precise predictions.

A more general formalism, van Kampen's $1/\Omega$ expansion, phrases the fully stochastic problem as a Taylor expansion in powers of a parameter Ω, e.g., the volume. Collecting terms of the same Ω order, one first gets the deterministic differential equation, then the Fokker-Planck equation (or equivalently, the Langevin equation), then higher order terms.

Fully stochastic models consider the individual molecules involved in gene regulation, rather than using concentrations and making the continuity assumption of differential equations. The *probability* that the next state consists of a certain number of molecules, given the current state, can be expressed in a straight-forward way. Typically, it is computationally intensive to deal with this framework, so various computational methods have been developed.

The next subsection considers three main views of gene regulation – fully Boolean, fully differential equations, or fully stochastic – in greater detail. Each of these can be stated relatively simply. The subsection after that discusses all the other methods, which will be viewed as combinations or hybrids between these three main methods.

4.2 "Pure" methods

4.2.1 Boolean

There are numerous "Boolean network" models based on Boolean logic [Kauffman 1993, 1969]. Each gene is assumed to be in one of two states: "expressed" or "not expressed." For simplicity, the states will be denoted "1" and "0," respectively. The state of the entire model is simply a list of which genes are expressed and which are not.

Example (Boolean State):

Consider in three genes, *X*, *Y*, and *Z*. If *X* and *Y* are expressed, and *Z* is not, the system state is 110.

From a given state, the system moves deterministically to a next state. There are two ways to express this: first, one may write out a *truth table*, which specifies what the next state is for each current state. If there are *n* genes,

there are 2^n states, each of which has exactly one next state. If one considers a reasonably small number of genes, say 5, one can write out then entire truth table exhaustively.

Example (Truth Table):

A possible truth table for the three genes in the previous example is:

Current state	Next State
000	000
001	001
010	010
011	010
100	011
101	011
110	111
111	111

Boolean functions provide an alternate representation of truth tables. Each variable is written as a function of the others, using the operators *AND*, which is 1 if all of its inputs are 1, *OR*, which is 1 if any of its inputs are 1, and *NOT*, which is 1 if its single input is 0. Truth tables and Boolean functions are equivalent in the sense that one can convert from one to the other using standard techniques. However, Boolean functions for the next state are typically relatively simple; although arbitrarily complex truth tables can exist, most of those are not possible gene regulation logic.

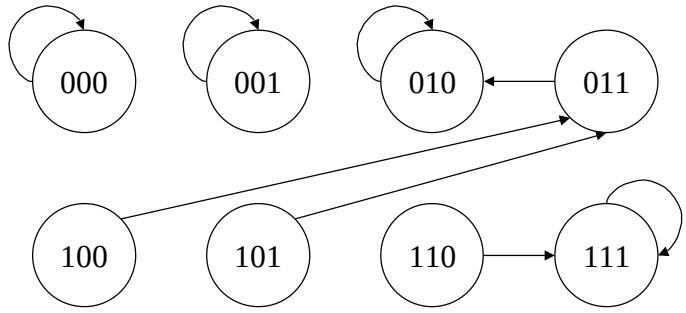

Figure 7– Finite state machine for the Boolean network example in the text.

Example (Boolean functions):

The functions that lead to the truth table in the previous example are:

X(next) = X(current) AND Y(current)
Y*(next) = X(current) OR Y(current)*
Z(next) = X(current) OR ((NOT Y(current)) AND Z(current))

A simple description of a biological system might be expressed as Boolean functions, for example, "C will be expressed if A AND B are currently expressed." Translating from this statement to a formal Boolean network model is straightforward, if it is known how the system behaves for all possible states. Typically, however, one does not know all states, and a Boolean network is one way to explicitly list states and find previously unknown interactions.

A third equivalent way to represent changes of state is as a *finite state machine*. Here, one draws and labels as many circles as there are states, then draws arrows from a state to the next state, as in Figure 7.

Equilibrium states can be found exhaustively. In the above example, the truth table shows that 000, 001, 010 and 111 are equilibrium states. In the finite state machine representation, the equilibrium states are easily recognizable as the states whose transition is to themselves.

4.2.2 Differential Equations (Reaction Kinetics)

In the differential equations approach to modeling gene regulation, the state is a list of the concentrations of each chemical species. These concentrations are assumed to be continuous; they change over time according to differential equations. It is possible to write differential equations for arbitrary chemical equations.

Consider a chemical equation, such as:

$$A \xrightarrow{k} A'$$

This denotes a reaction that occurs at a rate of $k[A]$, i.e., as $[A]$ increases, there is a linear increase in rate at which *A'* is created. As the reaction occurs, $[A]$ decreases and $[A']$ increases. In particular,

$$\frac{d[A]}{dt} = -k[A] \qquad \text{and} \qquad \frac{d[A']}{dt} = k[A]$$

Note that the rate of reaction only depends on the *reactants*, not on the *products*. For reactions with more than one reactant, the rate of reaction is the rate constant *k* times the product of the concentrations of each reactant.

Example:

The chemical equation

$$A + B \xrightarrow{k} AB$$

leads to the differential equations

$$\frac{d[A]}{dt} = -k[A][B], \qquad \frac{d[B]}{dt} = -k[A][B], \qquad \text{and} \qquad \frac{d[AB]}{dt} = k[A][B].$$

For stoichiometric coefficients other than one, one must raise the corresponding concentration to the power of the coefficient. Also, the rate of change of such a concentration will be equal to the stoichiometric coefficient times the rate of the reaction.

Example:

The chemical equation

$$2A \xrightarrow{k} A_2$$

leads to

$$rate = k[A]^2, \qquad \frac{d[A_2]}{dt} = k[A]^2 \qquad \text{and} \qquad \frac{d[A]}{dt} = -2k[A]^2.$$

When considering more than one chemical equation, the differential equation for a certain chemical is simply the sum of the contributions of each component reaction. For example, a previous section asserted that the chemical equations

$$X + DNA \xrightarrow{k_1} X \bullet DNA$$
$$X \bullet DNA \xrightarrow{k_{-1}} X + DNA$$

lead to the differential equation

$$\frac{d[X \bullet DNA]}{dt} = k_1[X][DNA] - k_{-1}[X \bullet DNA]$$

This transformation is simply the sum of the component rates of the two reactions.

Notation: One typically writes a vector, *v*, consisting of all the concentrations, i.e., the entire state. The change of state can then be expressed as $\frac{d}{dt}\vec{v} = f(\vec{v})$, where *f* is a known, but often non-linear, function.

To solve a system of differential equations, one needs not only the equations themselves, but also a set of initial conditions – the values of the concentrations at time 0, for example. Typically, writing the equations and finding the correct values of the rate constants and of the initial values are the conceptually interesting parts – in most actual gene regulatory systems, the equations become too complicated to solve analytically and so must be solved numerically. Many standard tools exist for solving and analysing systems of differential equations.

As stated in the previous section, the equilibrium states in the differential equations framework are found by setting all derivatives to 0, i.e., imposing the condition that there is no further change in state. Finding the equilibrium states is then just the algebraic problem: find $\vec{v}$ such that $0 = f(\vec{v})$.

Example (Ackers *et al.*):

Ackers *et al*. ('82, later extended in Shea and Ackers '85) modeled a developmental switch in lambda phage. The model considers the regulation of two phage proteins, repressor and cro. Regulation occurs in two ways: production and degradation. Degradation of the proteins follows the chemical equation

$$P \xrightarrow{k} nothing$$

Production is more complicated – a detailed model is provided of the binding of these two proteins and RNAP at three DNA binding sites – OR_1, OR_2 and OR_3. The binding is fast compared to protein production, and hence is assumed to have reached equilibrium. Recall that the fraction of the DNA in each possible binding configuration, *c*, is

$$frac_c = \frac{\exp(-\Delta G_c / RT)[Species_1]^{power_1} \cdots [Species_n]^{power_n}}{\sum_i \exp(-\Delta G_i / RT)[Species_1]^{power_1} \cdots [Species_n]^{power_n}}$$

In this model, each configuration also corresponds to a rate of production of cro and of repressor. The total *average* rate of production is just the sum of the rates of each configuration, weighted by the fraction of DNA in each configuration. Thus the differential equation for the concentration of repressor is:

$$\frac{d[rep]}{dt} = \frac{\sum_i \{rate_i(rep)\}\{\exp(-\Delta G_i / RT)[rep]^p [cro]^q [RNAP]^r\}}{\sum_i \exp(-\Delta G_i / RT)[rep]^p [cro]^q [RNAP]^r} - k_{rep}[rep]$$

There is a corresponding equation for [*cro*]. The key biochemical work is in determining the configurations (*c*) of the binding site, the free energies (ΔG_i) of these configurations, the rate constants of protein production – $rate_i$ (*rep*) and $rate_i$ (*cro*) – in each configuration, and the degradation constants (k_{rep} and k_{cro}). The original 1982 model considered only 8 configurations, and did not consider RNAP. The more complete 1985 model considered 40 configurations and also RNAP.

A generic problem with reaction kinetics models (and some others as well) is exponential complexity: the number of equations, hence unknown constants, can increase exponentially with the number of interdependent state variables. For example, *k* phosphorylation sites on a protein or *k* DNA binding sites in a promoter region would in principle yield 2^k states eligible to participate in reactions. . This problem is one reason for moving to coarser, more phenomenological models as in Section 5. Much of the practical work in gene regulation revolves around finding approximations to *f* that have certain properties. For example, *linear models* assume that *f* is linear in each of the concentrations, and *neural network* models assume that *f* consists of a weighted linear sum of the concentrations, followed by a saturating non-linearity. These sorts of approximations are useful in analyzing systems and in parameter estimation, i.e., taking some incomplete knowledge of the biochemistry plus experimental knowledge of the change of system state, and using those to develop better estimates of the rate and equilibrium constants. These topics are introduced in Section 5.

4.2.3 Stochastic Models

The stochastic framework considers the exact number of molecules present, a discrete quantity. The state is a list of how many molecules of each type are present in the system. The state changes discretely, but which change occurs and when that change occurs is probabilistic. In particular, the rate constants specify the probability per unit time of a discrete event happening.

Example:

A simple chemical equation, such as:

$$A \xrightarrow{k} A'$$

is interpreted as "one molecule of *A* is transformed into *A'*, the probability of this event happening to a given molecule of *A* in a given time *dt* is given by *k dt*." So, the probability of *some* molecule of *A* being transformed in a small time is *k {#A} dt*, where *{#A}* means "the number of molecules of A present."

For reactions with more than one reactant, the probability per unit time is given by the rate constant times the number of molecules of each – this is completely analogous to the differential equations case.

Aside: There is a subtlety in reactions with multiple copies of the same reactant, which will be discussed in the next chapter.

If there are multiple possible reactions, the state will change when any one of the reactions occurs. For example, consider the chemical reactions

$$X + DNA \xrightarrow{k_1} X \bullet DNA$$
$$X \bullet DNA \xrightarrow{k_{-1}} X + DNA$$

A short time later, the state will change from state (*{#X}, {#DNA}, {#X•DNA}*) to state (*{#X}-1, {#DNA}-1, {#X•DNA}+1*) with probability k_1 *{#X} {#DNA} dt*, to state (*{#X}+1, {#DNA}+1, {#X•DNA}-1*) with probability k_{-1} *{#X•DNA} dt*, or will stay in the same state with probability 1 - k_1 *{#X} {#DNA} dt* - k_{-1} *{#X•DNA} dt*

This framework is very different from the other two considered thus far, in that changes of the state of the system are probabilistic, not deterministic. From the same initial state, it is possible to get to multiple possible successor states. One ways to deal with this is to use a Monte Carlo simulation, i.e., to use a random number generator to simulate the changes of state. For any given set of random numbers picked, the change of state is deterministic. By making multiple trajectories through state-space, i.e., by picking different sets of random numbers, one generates statistics of the process. There are efficient algorithms available to do such Monte Carlo simulations (see Gillespie ('77), McAdams and Arkin ('98), and the next chapter). For general systems, this numerical Monte Carlo approach is the best known way to deal with the problem.

For small systems, i.e., systems where there are a small number of possible states, there is another technique: one deals with probabilities rather than numbers of molecules. In other words, the list of the numbers of molecules is a *configuration*, and the state is the probability distribution over all configurations. If the number of configurations is small, the number of probabilities one must deal with is also small. The probabilities obey a *master equation*, a linear differential equation with constant coefficients and certain other properties, whose solution is particularly easy if the number of states is finite and (typically) small.

Example (Stochastic Binding and Unbinding):

Consider the detailed diagram of transitions for protein/DNA binding in Figure 6. Let P(0,*t*) be the probability that the single molecule of DNA is in configuration 0 at time *t*. Assume there are *p* molecules of protein P present and *q* of protein Q. The probability of being in each state at time *t+Dt* is given by

$$\vec{P}(t+\Delta t) \equiv \begin{bmatrix} P(0,t+\Delta t) \\ P(1,t+\Delta t) \\ P(2,t+\Delta t) \\ P(3,t+\Delta t) \end{bmatrix} = A\vec{P}(t)$$

where

$$A = \begin{bmatrix} 1-pk_{01}\Delta t - qk_{02}\Delta t & k_{10}\Delta t & k_{20}\Delta t & 0 \\ pk_{01}\Delta t & 1-k_{10}\Delta t - qk_{13}\Delta t & 0 & k_{31}\Delta t \\ qk_{02}\Delta t & 0 & 1-k_{20}\Delta t - pk_{23}\Delta t & k_{32}\Delta t \\ 0 & qk_{13}\Delta t & pk_{23}\Delta t & 1-k_{31}\Delta t - k_{32}\Delta t \end{bmatrix}$$

Taking the limit as $Dt \to 0$ leads to a differential equation in the probabilities,

$$\frac{d\vec{P}}{dt} = \lim_{\Delta t \to 0} \frac{\vec{P}(t+\Delta t) - \vec{P}(t)}{\Delta t} = B\vec{P}(t)$$

where

$$B = \begin{bmatrix} -pk_{01} - qk_{02} & k_{10} & k_{20} & 0 \\ pk_{01} & -k_{10} - qk_{13} & 0 & k_{31} \\ qk_{02} & 0 & -k_{20} - pk_{23} & k_{32} \\ 0 & qk_{13} & pk_{23} & -k_{31} - k_{32} \end{bmatrix}$$

This type of system is called a continuous time *Markov chain*, because the probability of the next transition depends on the current state only, not on the history of states (Feller, van Kampen). Given initial conditions, one can solve this system of differential equations using standard methods. This example is a relatively simple case, in that there is precisely one molecule of DNA. In a different process, with two substances that could each be present in multiple copies, the approach would be similar, but the number of states would grow quickly.

In the stochastic framework, it is not possible to define an equilibrium configuration in terms of number of molecules, since any reaction that occurs changes the number of molecules. Recall that chemical equilibria in the differential equations framework are *dynamic* – chemicals are constantly changing from one form to another – but changes balance each other, and so there is no *net* change. This concept of equilibrium suffices for differential equations models, but not for stochastic models. For stochastic models, there is an *equilibrium probability distribution*, a set of probabilities that the system has certain numbers of molecules, and even though the system changes number of molecules, the changes and their probabilities balance out exactly.

Example (Equilibrium of Stochastic Binding and Unbinding):

The equilibrium distribution for the previous example, can be found by applying the equilibrium condition to the differential equation for the probabilities, namely:

$$0 = \frac{d\vec{P}}{dt} = B\vec{P}(t)$$

Thus, the equilibrium distribution is simply the normalized eigenvector of B that corresponds to the eigenvalue 1.

A different formulation of stochastic models for molecular simulations is provided by Stochastic Petri Nets as described by (Goss and Peccoud 1998).

4.2.4 Aside: Formal basis of the relationship between low-level models and higher

In principle, one could write out the full molecular dynamics – the positions and momenta of each particle. At an even lower level, one could create a quantum mechanical description of a system. Since either of these approaches would be computationally intractable for gene regulation and would not give insight into relevant system behavior, one makes certain assumptions and reduces the model to the stochastic framework, the differential equations framework, or the Boolean framework. The assumptions made should be noted. Defining precisely and justifying rigorously the conditions – number of molecules, time scale, etc. – under which each of the modeling frameworks is appropriate, remains an open problem.

To use a stochastic model rather than molecular dynamics, one assumes the solution is well-mixed, at least locally, i.e., that a given molecule is equally likely to be anywhere in the solution, or equivalently that the rate of molecular collisions is much greater than the rate of reactions.

Example:

The diffusion rate of green florescent protein in *E. coli* has been measured to be on the order of 1-10$\mu m^2/s$ (Elowitz *et al.*). *E. coli* has dimensions of order 1 μm, so for gene regulation on the order of seconds or tens of seconds, this assumption is fine.

Two additional assumptions are required to use the differential equations framework: one, that the number of molecules is sufficiently high that discrete changes of a single molecule can be approximated as continuous changes in concentration, and two, that the fluctuations about the mean are small compared to the mean itself.

It is straight forward to check whether the first condition is met. The second often appeals for its justification to the Central Limit Theorem of probability theory (Feller), which states that the sum of independent, identically distributed random variables with finite moments tends to a gaussian distribution, whose variance grows as the square root of the mean. Typically, one assumes this $1/\sqrt{N}$ noise is sufficiently small for $N \geq 100 - 1000$.

Example:

Consider a large number of molecules, each of which can degrade or not degrade independently. At a fixed time, the number remaining is simply the sum of the random variable θ, defined for a single molecule to be 1 if the molecule is present and 0 if it has degraded. This sum meets the criteria of the mean value theorem, so the differential equations approach is probably valid.

Example:

Consider the process of transcriptional elongation, i.e., when DNA is transcribed into mRNA nucleotide by nucleotide. One may model each nucleotide step as taking an exponentially-distributed amount of time, so the total time to move several steps down the DNA is a random variable that meets the criteria of the central limit theorem.

For more complex processes, it is not so simple to apply the central limit theorem; doing so may not even be legitimate. Under what precise conditions one can and cannot remains an open problem.

Fortunately we can also appeal to experimental data, rather than further modeling, to settle the question of model applicability for a particular system. From quantitative immunoflourescence measurements of *hunchback* and *Kruppel* protein expression levels in *Drosophila* syncytial blastoderms [Kosman et al. 1998], variations in measured flourescence between nuclei occupying similar positions on the anterior-posterior axis of a single blastoderm would seem to be about 10% of the average value for an "on" signal and 50-100% of the signal average when it is very low, or "off". These values would seem to be consistent with the requirements of differential equation modeling, and indeed differential equation models have high predictive value in this system.

To go from differential equations to Boolean models, one assumes that the function *f* in the differential equation $\frac{d}{dt}\vec{v} = f(\vec{v})$ has saturating non-linearities, i.e., that when *v* is very small or very large, *f(v)* tends a limit, rather than growing without bounds. Further, one assumes that the non-saturated region between the two extremes is transient and can be ignored. The lower and upper limits become "0" and "1", respectively.

4.3 Intermediate and "Hybrid" methods

The first part of this section described the three main views of gene regulation. The remaining models can be considered intermediates or hybrids, models that combine aspects of the three approaches previously presented.

4.3.1 Between Boolean Networks and Differential Equations

There have been numerous attempts to use the basic ideas of Boolean networks – that the state gene expression can be represented by discrete values, and the change in state follows simple rules – yet make models that are more related to the detailed biology than are Boolean networks themselves. Two examples are kinetic logic and the continuous logical networks of (Glass and Kaufmann 73). Others models combine grammars rather than boolean networks with differential equations.

4.3.1.1 Kinetic Logic

The *kinetic logic* formalism of Thomas *et al.* ('90 and '95) is more complex than Boolean networks, but has greater predictive value. The state is still discrete, but instead of each gene being "not expressed" or "expressed" only, as in the Boolean model, this formalism considers levels 0, 1, 2, 3, etc., which might correspond to "no

expression," "low level expression," "medium expression" and "high expression." Different genes may have a different granularity – one may only have states 0 and 1, while another may have multiple intermediate levels.

The rules for change of state are somewhat complex. One can write a *desired next state* for each possible current state. However, the *actual* next state is not necessarily the *desired* next state – one must first apply two constraints: *continuity* and *asynchronicity*. *Continuity* says that if a gene's current state is 0 and its desired next state is 3, for example, then its actual next state will be 1, i.e., it makes one step at a time toward its final goal. Biologically, this means a gene that is "not expressed" will be "expressed at a low level" before becoming "fully expressed." *Asynchronicity* means that the next state is found by letting a single gene change its state. This is in contrast to Boolean network models, where all genes change their states at the same time (synchronously). For a given state, it is possible that there can be more than one next state, each one consisting of a single gene changing its state.

Example:

Consider two genes, each of which have 3 possible states, "not expressed," "low level expression," and "fully expressed." The following truth table, which states the desired next state in terms of the current state, has an extra column with possible next states, found by applying the conditions of continuity and asynchronicity.

Current state(XY)	Desired next state	Possible next states
00	20	10
01	02	02
02	01	01
10	20	20
11	00	10, 01
12	00	02, 11
20	00	10
21	00	11, 20
22	00	12, 21

As was true for Boolean networks, one can represent this truth table as a finite state machine, as in Figure 8. Note that some states have multiple possible next states, indicated by multiple arrows out of those states.

It is useful to write logical equations for the desired next state rather than listing the truth table explicitly. As with Boolean networks, the functions that are possible are typically a subset of all possible functions. For multi-valued logic, however, it is somewhat more complicated to express the next state. Toward that end, we define a Boolean threshold function, defined by

$$X_T = \begin{cases} 0 & \text{if } X < T \\ 1 & \text{if } X \geq T \end{cases}$$

The next state logic can be defined in terms of such threshold functions. The biological meaning of these functions is simply that below a certain threshold level, gene *X* does not affect gene *Y*, but above that level, it does.

Example:

The desired next state function in the truth table above was generated by the functions:

$$X_{next} = \begin{cases} 0 & \text{if} \quad -2X_2 - 4Y_1 < -1 \\ 2 & \text{if} \quad 1 < -2X_2 - 4Y_1 \end{cases} \qquad Y_{next} = \begin{cases} 0 & \text{if} \quad -4X_1 + 4Y_1 - 2Y_2 < 1 \\ 1 & \text{if} \quad 1 < -4X_1 + 4Y_1 - 2Y_2 < 3 \\ 2 & \text{if} \quad 3 < -4X_1 + 4Y_1 - 2Y_2 \end{cases}$$

This corresponds to the textual statements: "At low concentrations, *X* represses expression of *Y*," "at low concentrations, *Y* represses expression of *X* and enhances its own expression," and "at high concentrations, each gene represses further expression of itself."

An interesting question arises when searching for equilibrium states. In the example, it appears there are no equilibrium states: the system will eventually go to the pair of states {01, 02} or to {10, 20}. What this really means is that the equilibrium points are on the thresholds between states. By defining a threshold state T_{12}, the model now reaches one of two equilibria, $0T_{12}$ or $T_{12}0$. In addition to the formalism, Thomas *et al*. have worked out the analysis to find such equilibria, which relies on the notion of *characteristic state* of a feedback loop.

4.3.1.2 Continuous Logical Networks

Another formalism between Boolean and differential equations is the work of (Mestl *et al*. 1995, 1996). Here the state consists of continuous concentrations, as in differential equations, but the change of state is simpler. Specifically, the formalism uses a simple linear differential equation for protein production, namely

$$\frac{d[X]}{dt} = k_1 - k_2[X]$$

Here k_1 and k_2 are functions of the concentrations of [X], [Y], [Z] and of any other proteins present in the system, but those functions are piecewise constant in the discrete ranges of "low," "medium," etc. (Boolean valued regions are a special case.) So within a certain region of state space, the coefficients k_1 and k_2 are constant, and one can solve the simple linear differential equations. By piecing together these simple solutions, one gets the complete behavior of the system.

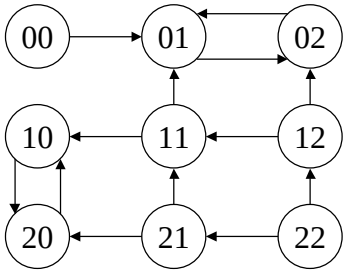

Figure 8– Finite state machine for the kinetic logic example in the text.

This formalism allows one to calculate steady states and more complete trajectories than are possible in the strictly algebraic approach (Boolean networks or kinetic logic), but it may require more detailed data to determine the dynamics from rate parameters.

4.3.1.3 Other Hybrid Systems

A more expressive modeling formalism than logical models or ODE's alone is provided by hybrid models such as developmental "grammars" that incorporate subsets of differential equations whose applicability is modulated by grammatical "rules" in a discrete, discontinuous manner (Mjolsness et al. 91, Prusinkiewicz et al. 93, Fleisher 95). A related formalism is provided by Discrete Event Systems.

4.3.2 Between Differential Equations and Stochastic

The theoretical biology community has done very little work between fully averaged formalisms, such as deterministic differential equations, and fully stochastic ones, such as master equations and transition probability matrices. There is, however, a useful level of modeling that has been used successfully by the physics and chemistry communities. This level is described equivalently by either a differential equation in concentrations with a stochastic noise term added (the Langevin equation) or a deterministic differential equation for the dynamics of a probability distribution (the Fokker-Planck equation). More generally, one may expand the master equation and drop higher ordered terms as in (van Kampen). For example, a stochastic process may be approximately Gaussian at any fixed time point, with mean and variance that follow differential equations. These formalisms and some conditions for their validity are introduced e.g. in (Risken 1989) and (van Kampen). They hold some promise for modeling the dynamics of chemical species (including many gene products) where the number of molecules involved is perhaps 10-1000, in between "a few" and "large numbers". We outline these approaches here, with the

warning that the literature on these topics is more mathematical than many of the others for which a good biological introduction is available.

4.3.2.1 Langevin

Recall that in the differential equations formalism, one writes the state as a vector *v* of concentrations of different chemicals, which changes in time according to a differential equation

$$\frac{d}{dt}\vec{v} = f(\vec{v})$$

Here, *f* is derived from the chemical equations of the system. In the simplest Langevin approach, one extends this function to be

$$\frac{d}{dt}\vec{v} = f(\vec{v}) + n(t) \qquad (1)$$

The function *n* is a (multivariable) function of time only; it does not depend on the current state. The function *n* is a noise term – it induces stochastic fluctuations of the state *v* about its deterministic value.

For a particular function *n*, one can solve the equation (1) using the standard techniques of differential equations. The key, however, is that *n* is a random function, so one deals with the expected value of the state *v*, denoted <*v*>, and with fluctuations about that state. Specifically, one assumes that the noise term has zero mean and is uncorrelated with itself over time:

$$\langle n(t)\rangle = 0 \quad \text{and} \quad \langle n(t)n(t')\rangle = \Gamma \times \delta(t-t') \qquad (2)$$

The latter equation means that the correlation is 0 for two *different* times *t* and τ, and is Γ for the same time (i.e., the variance is Γ).

The general approach to using the Langevin equation is:

- Write the deterministic set of differential equations.
- Add a random term *n(t)* with the properties (2).
- Adjust Γ to give the correct equilibrium distribution, as determined by thermodynamics.
- Take expectations of (1) to get the state and the variance as functions of time.

Aside (Applicability of Langevin approach):

Consider the last step of this approach, taking expectations of (1)

$$\left\langle \frac{d}{dt}\vec{v} \right\rangle = \left\langle f(\vec{v}) \right\rangle + \langle n(t)\rangle$$

Because expectation is a linear operator, the right hand side may legitimately be split into two parts. This equation would be a deterministic differential equation with variable <*v*> if it were:

$$\frac{d}{dt}\left\langle \vec{v} \right\rangle = f\left(\left\langle \vec{v} \right\rangle\right) + 0$$

The first part, changing the order of expectation and the derivative, is always legitimate. The last part – the zero – follows from (2). The middle part, switching the order of *f* and of the expectation, is strictly valid only if *f* is linear. Alternately, if *f* is non-linear, but can be approximated as linear, this is also valid. In physics, the Langevin approach has been applied successfully to linear cases. If, however, *f* has large non-linearities on the order of the fluctuation, this approach breaks down. Specifically, two different linear approximations may produce very different final results.A more complicated version of the Langevin approach consists of adding a noise term that *can* depend on the state. In particular, one replaces (1) with

$$\frac{d}{dt}\vec{v} = f(\vec{v}) + g(\vec{v})n(t) \quad (3)$$

Here *g(v)* is a deterministic function of *v* only, which determines the scaling of the noise term. Unfortunately, this simple addition complicates the mathematics significantly, because *n(t)* is a singular function consisting of δ functions, as in (2). We shall not go into the details of this problem, but only say that there are two different interpretations of (3), the Itô interpretation and the Stratonovich interpretation. See van Kampen or Risken for a more detailed discussion.

4.3.2.2 Fokker-Planck

One way of characterizing the Langevin approach would be "start with the differential equations approach and make it more like the stochastic approach." In a complementary way, the Fokker-Planck approach starts with the fully stochastic model and makes it more like the differential equations model.

Recall that the state of the fully stochastic model consists of the number of molecules present, a discrete quantity. Equivalently, the probability distribution *P(number of molecules, t)* is a multivariable function that depends both on the continuous variable *t* and the discrete variables, *number of molecules*. This function is hard to deal with explicitly, unless the number of discrete states is small, in which case one can handle it exhaustively above. The Fokker-Planck approach deals with *P*, but assumes that it depends *continuously* on the number of molecules. By letting the vector *v*, the number of each kind of molecule, consist of real numbers and not just integers, the function *P(v, t)* will be continuous in all its variables. Given this approximation, the Fokker-Planck approach writes a *partial* differential equation for *P*, namely

$$\frac{\partial}{\partial t}P(\vec{v},t) = -\frac{\partial}{\partial v}A(\vec{v})P(\vec{v},t) + \frac{1}{2}\frac{\partial^2}{\partial v^2}B(\vec{v})P(\vec{v},t) \quad (1)$$

From the particular form of this equation, it follows that

$$\frac{\partial}{\partial t}\left\langle \vec{v} \right\rangle = \left\langle A(\vec{v}) \right\rangle \quad \text{and} \quad P_{eq}(\vec{v}) = \frac{const}{B(\vec{v})}\exp\left[2\int_0^v \frac{A(\vec{v}')}{B(\vec{v}')}d\vec{v}'\right] \quad (2)$$

Note that P_{eq} is an equilibrium probability distribution, as in the fully stochastic case, not an equilibrium state, as in the differential equations case.

As with the Langevin approach, the Fokker-Planck equation (1) is most useful when the dynamics are linear. (Here "linear" means *A* is a linear function of *v* and *B* is constant.) For nonlinear equations, the same caveats apply as in the Langevin method. For linear functions,

$$\frac{\partial}{\partial t}\left\langle \vec{v} \right\rangle = A(\left\langle \vec{v} \right\rangle) \quad (3)$$

and one may use the following algorithm to apply the Fokker-Planck formalism:

- Let *A* be the (linear) function that describes the differential equations version of the dynamics, as in (3).
- From thermodynamics, find the equilibrium probability distribution, P_{eq}.
- Use this equilibrium probability distribution, *A*, and Equation (2) to find *B*.
- From *A* and *B*, solve the partial differential equation (1) to find the complete dynamics as a function of time.

A generalization of the Fokker-Planck approach, the *Kramers-Moyal* expansion, views (1) as the first term in a Taylor expansion for the actual function *P*. The Kramers-Moyal expansion consists of terms of all orders. Cutting off after the first-order term leads to (1). More generally, one can cut off the Kramers-Moyal expansion of the master equation after any number of terms and use that approximation.

Note: The Fokker-Planck formalism is provably equivalent to the Langevin approach.

4.3.2.3 1/W expansion (van Kampen)

There are other, more general, approaches to dealing with a stochastic system where one can assume the fluctuations are small compared to the average. One such approach, van Kampen's 1/Ω expansion, assumes the fluctuations are on the order of the square root of the average and writes the actual number of molecules, *n*, in terms of the average concentration, the fluctuations, and a parameter, Ω, say the volume:

$$n(t) = \Omega c(t) + \sqrt{\Omega} f(t)$$

One now rewrites the master equation for the dynamics of the system in terms of *c(t)* and *f(t)*, not *n(t)*, and expands in powers of Ω. Collecting terms with the same power of Ω gives, in order:

1. The macroscopic, deterministic equation.
2. The Fokker-Planck equation (equivalent to a Langevin approach).
3. Higher order terms.

The details of this expansion are beyond this chapter. We refer the reader to van Kampen for a detailed explanation.

5 Parameter Estimation, Data fitting, Phenomenology

5.1 Introduction, overview

The previous section described how to use complete models to generate predictions. This section considers the inverse problem – how to use observed system dynamics to generate or improve models. This can be considered as two sub-problems:

- Write equations with unknown parameters.
- Find the values of those unknown parameters that lead to predictions that best match the dynamics.

If all the chemical equations are known, and only the parameters are unknown, the first problem is solved, and one need only consider the second. However, coarser and more phenomenological alternative models may be required as a practical matter when the relevant states and transition rates are unknown or are known to a very limited extent, as is likely for large protein complexes such as subcomplexes of the eukaryotic transcription complex.

The general problem of parameter estimation is hugely important across different scientific disciplines. As such, there are different communities that have different names for parts of the process, different problems with which they are concerned, and different techniques. The linear systems community, for example, considers parameter estimation in connection with linear models; the parameter estimation is called *system identification*. The speech recognition community has considered many problems related to estimating parameters of Markov chains, in particular, *Hidden Markov Models*. The computational learning theory community refers to this type of estimation as *learning*, and considers: which structures (such as neural networks) provide a good functional basis for estimating arbitrary non-linear functions, how much data is necessary to make an estimate with a particular degree of certainty, which algorithms (such as simulated annealing, backpropagation, etc.) can be used to do estimation, and many other problems.

A major impetus for the parameter-fitting point of view is that systems with many interacting state variables, such as medium-sized or large protein complexes, have exponentially many states; therefore detailed state-by-state modeling becomes prohibitive to constrain with real data. More approximate, phenomenological models become attractive as targets of parameter-fitting. We introduce several such models below.

5.2 Boolean Models

There are several ways to learn Boolean models from data. If one can make a complete truth table, there are standard techniques that are taught in every introductory computer engineering class. Typically, this exhaustive enumeration of states is not feasible in biological systems, and one uses a more refined technique. The computational learning theory community (see Kearns and Vazirini) has developed techniques for learning

Boolean functions from observations of the inputs and outputs. See for example, the work of Jackson. Somogyi *et al*. have developed a technique based on information theory, which has been applied to biological systems and will be covered in detail in Chapter 6. Finally, some computational learning theory work has focussed on uncovering the structure of a finite state machine without being able to see the individual states, only the final states and the inputs required to get to that state (the algorithm is covered in Kearns and Vazirini). Although this has not been applied to biology, it could potentially provide an interesting way to deduce regulatory details that one cannot observe directly.

5.3 Methods for ODE models

The training methods for models consisting of systems of ordinary differential equations (ODE's) are so far the most developed. Loosely, one can consider two problems: techniques for approximating the full dynamics to allow easier training and techniques for training, given the approximated dynamics.

5.3.1 Approximating dynamics

Here the problem is to find a family of functions that can approximate the function *f* in the dynamics,

$$\frac{d}{dt}\vec{v} = f(\vec{v})$$

given only time course or related behavioral data. Below we list series of formulations for f which have the "universal approximation" property that, given enough unobserved state variables, they can approximate the update dynamics of "most" other dynamical systems. The linear approximation is an exception but is sometimes convenient. Each of these ODE formulations generalizes immediately from one gene to many interacting genes in a feedback circuit.

5.3.1.1 Linear

As a first approach, one may approximate *f* as a linear function, using a matrix *L*:

$$\frac{d}{dt}\vec{v} = L\vec{v}$$

or in component notation

$$\frac{dv_i}{dt} = \sum_j L_{ij} v_j \,.$$

The techniques for learning the matrix *L* are well studied under the name *system identification* (see Ljung).

5.3.1.2 Recurrent Artificial Neural Networks

Another approach, which has been applied to real gene expression data in several cases as described in Chapter 5 of this volume, is to use analog-valued recurrent Artificial Neural Network (ANN) dynamics with learnable parameters. One version of such network dynamics, in use for gene regulation modeling, is given by (Mjolsness et al. 91)

$$t_i \frac{dv_i}{dt} = g(\sum_j T_{ij} v_j + h_i) - l_i v_i$$

Here the state variables *v* are indexed by *i*, and therefore connection strengths *T* between them take two such indices *i* and *j*. The function *g* is a saturating non-linearity, or sigmoid. The parameters *T*, τ, λ, and *h* have been successfully "trained" from spatio-temporal patterns of gene expression data (Reinitz et al. 92).

Diffusion and cell-cell signaling interactions have been incorporated in this model (Mjolsness et al. 91, Marnellos 97). Controlled degradation could perhaps be handled as well using a degradation network $\hat{T}_{ij}$:

$$t_i \frac{dv_i}{dt} = g(\sum_j T_{ij} v_j + h_i) - l_i v_i g(\sum_j \hat{T}_{ij} v_j + \hat{h}_i),$$

though no computer parameter-fitting experiments have been performed on such a model yet.

5.3.1.3 Sigma-Pi Neural Networks

Two further models can be mentioned as attempts to incorporate promoter-level substructure into gene regulation networks that are otherwise similar to the ANN approach. In Chapter 1 of this volume, "Sigma-Pi units" (Rumelhart, Hinton and McClelland 86) or "higher-order neurons" are introduced to describe promoter-level substructure: regulatory "modules" in sea urchin Endo16 promoter. A typical feed-forward Sigma-Pi neural network is described by the relationships between variables *v* and weights *T* at layer network *l* and variables *v* at network layer *l+1*:

$$v_i^{l+1} = g(\sum_{jk} T_{ijk}^l v_j^l v_k^l + \sum_j T_{jk}^l v_j^l + h_i^l) \ .$$

A novelty here is the third-order (three-index) connections T_{ijk} in which two input variables indexed by *j* and *k* jointly influence the activity of the *i*-th output variable. This equation can hold at each of a number of layers in a feed-forward architecture. If the three-index connection parameters T_{ijk} are all zero, the system specializes to a conventional feed-forward layered neural network with second-order (two-way) connections T_{ij} at each layer. Yet higher-order connections may also be included, but can be transformed into multiple third-order interactions with the addition of extra variables *v* to compute intermediate products. (To be exact, this transformation requires that some g's be linear.)The generalization to continuous-time recurrent ODE's as in Section 5.3.1.2 is straightforward.

5.3.1.4 Hierarchical model

In Chapter 5 Section 3, partition functions for promoter regulatory regions, dimerization, and competitive binding are proposed as a way to expand a single-node description of selected genes in a regulatory circuit into a subcircuit of partial activation states within a eukaryotic transcription complex. A simplified version of the HCA model, omitting heterodimers and competitive binding and restricting to two levels of hierarchy, is:

$$t_i \frac{dv_i}{dt} = \frac{Ju_i}{1 + Ju_i} - l_i v_i$$

$$u_i = \prod_{a \in i} \left(\frac{1 + J_a \tilde{v}_a}{1 + \hat{J}_a \tilde{v}_a} \right)$$

$$\tilde{v}_a = \frac{\tilde{K}_a \tilde{u}_a}{1 + \tilde{K}_a \tilde{u}_a}$$

$$\tilde{u}_a = \prod_{b \in a} \left(\frac{1 + K_b v_{j(b)}^{n(b)}}{1 + \hat{K}_b v_{j(b)}^{n(b)}} \right)$$

where *i* and *j* index transcription factors, α indexes promoter modules, *b* indexes binding sites, and the function *j(b)* determines which transcription factor *j* binds at site *b*.

5.3.1.5 Generalized mass action

Another example of a phenomenological model for gene regulation is the proposal by (Savageau 1998) to use "Generalized Mass Action" with nonintegral exponents *C* for modeling transcriptional regulation. In the above notation,

$$t_i \frac{dv_i}{dt} = \sum_a k_{ia}^+ \prod_j v_j^{C_{iaj}^+} - \sum_a k_{ia}^- \prod_j v_j^{C_{iaj}^-}$$

which generalizes enzyme kinetics equations to nonintegral stoichiometric coefficients, and is also related to Sigma-Pi neural networks. State variables v_i remain positive if $C_{iai}^- > 0$, as they should if they are to be interpreted as concentrations. A more computationally and analytically tractable specialization of GMA is to "S-systems" (Savageau and Voit 87):

$$t_i \frac{dv_i}{dt} = k_i^+ \prod_j v_j^{C_{ij}^+} - k_i^- \prod_j v_j^{C_{ij}^-}$$

which have only one positive and one negative summand (interaction) per gene v_i . This form retains universal approximation capability.

The simplified enzyme kinetics model of (Furusawa and Kaneko 98), a specialization of GMA with diffusion added, is also worth mentioning as a possible target for parameter-fitting.

5.3.2 Training parameters

Several techniques exist for training parameters from trajectory (behavioral) data. System identification has already been mentioned in the context of linear models. For the non-linear approximation schemes above, several different techniques are available.

5.3.2.1 Back Propagation

For feed-forward neural networks, with two-way connections or higher-order ones, one can develop efficient gradient descent algorithms for training the network to match a "training set" of input/output patterns. The best known of these algorithms is the "batch mode" version of back-propagation (see Rumelhart , Hinton and Williams 86) in which an error signal propagates backwards through the network layers and alters each connection. In practice, a simpler "incremental" version of the algorithm, which only follows the gradient when averaged over many time steps, is often more effective. The incremental algorithm is in effect a form of stochastic gradient descent, in which minor local minima can sometimes be escaped.

An easy generalization of the back-propagation algorithm applies to discrete-time recurrent neural networks such as fixed-timestep approximate discretizations of continuous-time recurrent neural networks. This "back-propagation through time" (BPTT) algorithm involves unrolling the network into a deep, many-layer feedforward neural net in which each layer shares connection strength parameters with the others. But it is not necessary to make this approximation; a directly applicable gradient-descent learning algorithm for continuous-time recurrent neural networks was investigated by (Pearlmutter 89).

These and other variations of back-propagation training are surveyed in (Hertz et al. 91).

5.3.2.2 Simulated Annealing

Another form of stochastic gradient descent is simulated annealing. This technique is typically much less efficient, but may still work when other data-fitting optimization procedures become ineffective due to large numbers of local minima. A particular variant of simulated annealing (Lam and Delosme 88a, 88b) has proven effective at inferring small analog neural networks from trajectory data (Reinitz et al. 92, Reinitz and Sharp 95). This version adaptively chooses discrete step sizes for adjusting the analog parameters such as connection strengths, and also adjusts the temperature schedule, so as to maintain a constant ratio of accepted to rejected annealing steps. A relatively gently-saturating sigmoid function *g* is used to help smooth out the objective function for simulated annealing optimization.

An attractive possibility is to combine recurrent back-propagation with simulated annealing, perhaps by using the former as an inner loop in a simulated annealing optimization. (Erb and Michaels 99) have provided a sensitivity analysis of the fitting problem for ANN models, illustrating some of the computational difficulties encountered.

5.3.2.3 Genetic Algorithms

Genetic algorithms are another class of stochastic optimization methods that sometimes work relatively well on hard optimization problems. Here, single-parameter update steps such as those of simulated annealing are augmented by "crossover" operations in which two complete configurations of the unknown parameters, chosen out of a population of N partially optimized ones, are combined to produce "progeny" subject to selection as part of the optimization procedure. This kind of algorithm has been investigated for inferring analog neural network models of gene regulation in (Marnellos 97), where it outperformed simulated annealing on artificial test problems but not on the more realistic biological problems tried. It is possible that further work could substantially improve the situation.

5.3.2.4 Boolean Network Inference

The "REVEAL" algorithm of (Liang et al. 98) provides a starting point for inferring Boolean networks of low input order and 50 network elements. Building on this work, much larger networks were identified by (Akutsu et al 99). It is possible that inference of Boolean networks will prove to be essentially more tractable than inference of continuous-time recurrent neural networks, in cases where a parsimonious Boolean network approximation of the target trajectory data actually exists.

5.4 Stochastic Models

A future challenge will be to develop efficient algorithms to train parameters in the stochastic framework. The speech recognition community and the bioinformatics community have done a significant amount of work using *Hidden Markov Models* (Rabiner) – this work includes estimating parameters when the state (and maybe even the model) is not uniquely determined. The ion-channel community has done work on estimating parameters when better knowledge of the state and of the biology are available (Colquhoun and Hawkes). To our knowledge, these techniques have not been applied to gene regulation systems.

6 Summary & Roadmap

6.1 Summary

One way to approach this chapter is to ask how to use the techniques covered here to aid in understanding a biological system. The method outlined in this chapter is:

- Create a calculation independent model, a system of chemical and physical equations summarizing what is known or suspected about the system.
- Make some additional, computational assumptions, and use one of the model frameworks in the chapter. Chose a modeling framework whose assumptions are reasonable for the biological system.
- If biochemical constants are available, use them. If some or all of the biochemical constants are not available, use the parameter estimation techniques covered to find values of the missing constants.
- Use the model to make predictions.
- Compare the predictions of the model with experimental data to improve the model and to increase understanding of the system.

Another view of the chapter is as a starting point for further research. There are several major areas:

- Develop faster or better algorithms to deal with the frameworks available. In particular, develop tools that are easy to use for the non-specialist.

- Fill in gaps in this process. There are numerous spots in the chapter where a point is missing because no research has been done in that area. Examples include, but are not limited to: applying the Fokker-Planck or Langevin formalisms to gene regulation systems, developing better ways to parameterize complex eukaryotic systems, and developing estimation techniques for the fully stochastic formalism.
- Use modeling and theory to add value to the biology – there is more do with a model than just simulate time courses. Some useful analyis techniques have been worked out for the differential equations formalism; techniques for the other formalisms are incomplete or non-existent.

6.2 Where to go from here

The remainder of this book contains a number of chapters related to gene regulation. Chapter 3 presents a simple model of prokaryotic gene regulation in the fully stochastic formalism. Chapter 4 presents a model of a eukaryotic system, and includes elements of the differential equations formalism and some discrete interactions as well. Chapter 5 presents a neural network model of *Drosophila melanogaster* and illustrates not only the differential equations framework, but also the usefulness of phenominological models and the techniques of parameter estimation. Chapter 6 presents an extensive example of the Boolean network framework and parameter estimation in that framework.

In addition to the chapters of this book, several other references are available. The list that follows is by no means complete, but should provide a starting point for further study of gene regulation and of mathematical modeling.

7 References

7.1 General

7.1.1 Physical Chemistry

Cooperativity Theory in Biochemistry: Steady-State and Equilibrium Systems.
T. L. Hill
Springer Series in Molecular Biology, Springer-Verlag, 1985.
See especially pp. 6-8, 35-36, and 79-81.

The Fokker-Planck Equation: Methods of Solution and Applications (Second Edition)
H. Risken
Springer, 1989

7.1.2 General Biology/Development

Developmental Biology
S. F. Gilbert
Sinauer Associates, (1997)

Molecular Biology of the Gene
J. D. Watson, N. H. Hopkins, J. W. Roberts, J. Argetsinger Steitz and A. M. Weiner
The Benjamin/Cummings Publishing Company, Inc. (1987)

Molecular Biology of the Cell, 3rd Edition
B. Alberts, D. Bray, J. Lewis, M. Raff, K. Roberts, J. D. Watson
Garland Publishing, Inc. (1994)

Gene Activity in Early Development
E. H. Davidson
Academic Press, Inc. (1986)

Molecular Mechanisms of Cell-Type Determination in Budding Yeast
A. Johnson
Current Opinion in Genetics & Development 5:552-558, 1995.

Regulation of *even-skipped* Stripe 2 in the *Drosophila* Embryo,
S. Small, A. Blair, and M. Levine,
The EMBO Jounral 11:11, pp. 4047-4057, 1992.

Regulation of Two Pair-Rule Stripes by a Single Enhancer in the *Drosophila* Embryo
S. Small, A. Blair, and M. Levine,
Developmental Biology 175:314-324, 1996.

Transcriptional Repression in the *Drosophila* Embryo
S. Gray, H. Cai, S. Barolo and M. Levine
Phil. Trans. R. Soc. Lond. B 349, pp. 257-262, 1995.

The Riddle of MAP Kinase Signaling Specificity
H. D. Madhani and G. R. Fink
Trends in Genetics 14:4, 1998.

Automated Assay of Gene Expression at Cellular Resolution
D. Kosman, J. Reinitz, and D. H. Sharp
Pacific Symposium on Biocomputing ‘98
Eds. R. Altman, A. K. Dunker, L. Hunter, and T. E. Klein
World Scientific 1998

The N-end rule
A. Varshavsky
Cold Spring Harbor Symposium on Quantitative Biology
60: 461-478 1995

7.1.3 Probability and Stochastic Processes

An Introduction to Probability Theory and Its Applications
W. Feller
John Wiley & Sons, Inc. (1966)

Stochastic Processes in Physics and Chemistry
N. G. Van Kampen
North-Holland

7.1.4 Reviews

Simulation of Prokaryotic Genetic Circuits
H. H. McAdams and A. Arkin
Annu. Rev. Biophys. Biomol. Struct. 27 (1998) p199-224

An Integrated Model of the Transcription Complex in Elongation, Termination, and Editing
P. H. von Hippel
Science 281 (31 July 1998) p660-665

7.2 Specific modeling techniques

7.2.1 Based on Physical Chemistry

Quantitative Model for Gene Regulation by Lamba Repressor.
G. K. Ackers, A. D. Johnson and M. A. Shea
PNAS USA 79 (1982) p1129-1133

The OR Control System of Bacteriophage Lambda, A Physical-Chemical Model for Gene Regulation.
M. A. Shea and G. K. Ackers
J Mol Biol 181 (1985) p211-230

Exact Stochastic Simulation of Coupled Chemical Reactions
D. T. Gillespie
J Phys Chem 81 #25 (1977) p2340-2361

Stochastic Approach to Chemical Kinetics
D. A. McQuarrie
J Appl Prob 4 (1967) p413-478

Stochastic Mechanisms in Gene Expression
H. H. McAdams and A. Arkin
PNAS USA 94 (February 1997) p814-819

Stochastic kinetic analysis of developmental pathway bifurcation in phage λ-infected Escherichia coli cells.
A.P. Arkin, J. Ross, H. H. McAdams

Genetics (1998) 149:1633-1648

Protein mobility in the cytoplasm of Escherichia coli
M. B. Elowitz, M. G. Surette, P. E. Wolf, J. B. Stock, and S. Leibler
J Bacteriol 1999 Jan;181(1):197-203

Quantitative Modeling of Stochastic Systems in Molecular Biology by Using Stochastic Petri Nets
P. J. E. Goss and J. Peccoud
Proc Natl. Acal. Sci. USA 95:6750-6755, 1998.

7.2.2 Boolean Network Models

The Origins of Order
S. A. Kauffman
Oxford University Press, 1993.
See also: J. Theor. Biol. 22, p. 437, 1969.

Biological Feedback
R. Thomas and R. D'Ari
CRC Press (1990)

Dynamical Behaviour of Biological Regulatory Networks - I. Biological Role of Feedback Loops and Practical Use of the Concept of the Loop-Characteristic State
R. Thomas, D. Thieffry and M. Kaufman
Bull Math Biol 57 #2 (1995) p257-276

7.2.3 Differential Equation and Hybrid Models

A Connectionist Model of Development
E. Mjolsness, D. H. Sharp, and J. Reinitz
Journal of Theoretical Biology 152:429-453, 1991.

The Logical Analysis of Continuous, Non-Linear Biochemical Control Networks
L. Glass and S. A. Kauffman
J. Theor. Biol. 39:103-129, 1973.

A Mathematical Framework for Describing and Analysing Gene Regulatory Networks
T. Mestl, E. Plohte and S. W. Omholt
J Theor Biol 176 (1995) p291-300

Chaos in High-Dimensional Neural and Gene Networks
T. Mestl, C. Lemay, L. Glass
Physica D 98, p. 33, 1996.

Genomic Cis-Regulatory Logic: Experimental and Computational Analysis of a Sea Urchin Gene,
C.-H. Yuh, H. Bolouri, and E. H. Davidson,
Science 279:1896-1902, 1998.

Animation of Plant Development
P. Prusinkiewicz, M. S. Hammel, and E. Mjolsness
SIGGRAPH '93 Conference Proceedings, ACM 1993.

A Multiple-Mechanism Developmental Model for Defining Self-Organizing Geometric Structures

K. Fleischer
PhD Thesis, Caltech Computer Science Department, 1995.

7.2.4 Phenomenogical Models, Learning and Parameter Estimation

An Introduction to Computational Learning Theory
M. J. Kearns, U. V. Vazirani
MIT Press, Cambridge, Mass. 1994

An Efficient Membership-Query Algorithm for Learning DNF with Respect to the Uniform Distribution
J. Jackson
IEEE Symp. Found. Comp. Sci., 35, pp. 42-53, 1994.

System identification: theory for the user
L. Ljung
Prentice-Hall, Englewood, N.J. 1987
A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition
L. R. Rabiner
Proceedings of the IEEE, vol 77. No. 2, Feb 1989

Model for Cooperative Control of Positional Information in *Drosophila* by Bicoid and Maternal Hunchback
J. Reinitz, E. Mjolsness, and D. H. Sharp
Technical Report LAUR-92-2942 Los Alamos National Laboratory, 1992.
Also Journal of Experimental Zoology 271:47-56, 1995.

Mechanism of *eve* Stripe Formation
J. Reinitz and D. H. Sharp
Mechanisms of Development 49:133-158, 1995.

Gene Network Models Applied to Questions in Development and Evolution
G. Marnellos
Ph.D. Disseration, Yale University, 1997.

A General Framework for Parallel Distributed Processing
D. E. Rumelhart, G. E. Hinton, and J. L. McClelland
Parallel Distributed Processing, Vol. 1, Chapter 2
Rumelhart and McClelland, eds.
MIT Press, 1986.

Recasting Nonlinear Differential Equations as S-Systems: A Canonical Nonlinear Form
M. A. Savageau and E. O. Voit
Mathematical Biosciences 87:83-115, 1987.

Rules for the Evolution of Gene Circuitry
M. A. Savageau
Pacific Symposium on Biocomputing ‘98
Eds. R. Altman, A. K. Dunker, L. Hunter, and T. E. Klein
World Scientific 1998

Emergence of Multicellular Organisms with Dynamic Differentiation and Spatial Pattern
Chikara Furusawa and Kunihiko Kaneko
Artificial Like VI
Eds. C. Adami, R, K. Belew, H. Kitano, and C. E. Taylor
MIT Press 1998.

Introduction to the Theory of Neural Computation
J. Hertz, A. Krogh and R. G. Palmer
Addison-Wesley, Redwood City, CA 1991

Learning Internal Representations by Error Propagation
D. E. Rumelhart, G. E. Hinton, and R. J. Williams
Parallel Distributed Processing, Vol. 1, Chapter 8
Rumelhart and McClelland, eds.
MIT Press, 1986.

Learning State Space Trajectories in Recurrent Neural Networks
B. A. Pearlmutter
Neural Computation 1, p. 263-269, 1989.

An Efficient Simulated Annealing Schedule: Derivation.
J. Lam and J. M. Delosme
Technical Report 8816, Yale University Electrical Engineering Department, New Haven CT, 1988.

An Efficient Simulated Annealing Schedule: Implementation and Evaluation
J. Lam and J. M. Delosme
Technical Report 8817, Yale University Electrical Engineering Department, New Haven CT, 1988.

Sensitivity of Biological Models to Errors in Parameter Estimates
R. S. Erb and G. S. Michaels
Pacific Symposium on Biocomputing '99
Eds. R. B. Altman, A. K. Dunker, L. Hunter, T. E. Klein and K. Lauderdale
World Scientific 1999.

REVEAL, A General Purpose Reverse Engineering Algorithm for Inference of Genetic Network Architectures.
S. Liang, S. Fuhrman and R. Somogyi
Pacific Symposium on Biocomputing '98
Eds. R. B. Altman, A. K. Dunker, L. Hunter and T. E. Klein
World Scientific 1998.

Identification of Genetic Networks from a Small Number of Gene Expression Patterns under the Boolean Network Model
T. Akutsu, S. Miyano, S. Kuhara
Pacific Symposium on Biocomputing '99
Eds. R. B. Altman, A. K. Dunker, L. Hunter, T. E. Klein and K. Lauderdale
World Scientific 1999.

On the Stochastic Properties of Bursts of Single Ion Channel Openings and of Clusters of Bursts
D. Colquhoun and A. G. Hawkes
Phil. Trans. R. Soc. Lond. B. 8000, 1-59 (1982)