A probabilistic model of a prokaryotic gene and its regulation[1]

Michael Andrew Gibson[2] & Jehoshua Bruck

Department of Computation and Neural Systems

Caltech 136-93

Pasadena, CA 91125

{gibson, bruck}@paradise.caltech.edu

1 Introduction/motivation

The previous chapter described several different kinds of models that are used to model gene regulation. This chapter focuses exclusively on stochastic models (Van Kampen, 1992),(McQuarrie, 1967). As mentioned in the previous chapter, the stochastic framework is important when the number of molecules involved is small, the time scale is short, or both. This chapter will not rehash the theoretical underpinnings of the previous chapter, but rather, will provide an extensive example, which illustrates many of the biological processes involved in (prokaryotic) gene regulation and also many of the stochastic processes used to model these biological processes. The system to be modeled is part of the regulatory circuitry of lambda phage, and is a small subset of the complete model in (Arkin et al., 1998).

1.1 Lambda phage biology

Lambda phage (Ptashne, 1992) is a virus that infects *E. coli* cells. It is called a *temperate* phage, because it has two possible developmental pathways (see Figure 1) – it can either:

- replicate and *lyse* (dissolve) the host cell, thus releasing about 100 progeny or
- integrate its DNA into the bacterial DNA and form a *lysogen*.

In the latter case, the virus will replicate passively whenever the bacterium replicates. A lysogen also has immunity from subsequent lambda infections; this protects the lysogen from being destroyed should another phage infect the host cell. Under the right conditions (e.g., exposure to UV light), a lysogen can be *induced*, i.e., the viral DNA excises itself from the bacterial DNA and undergoes normal replication and lysis. (See (Ptashne, 1992).)

[1] Supported in part by ONR grant N00014-97-1-0293, by a JPL-CISM grant, by NSF Young Investigator Award CCR-9457811 and by a Sloan Research Fellowship.

[2] To whom correspondence should be addressed.

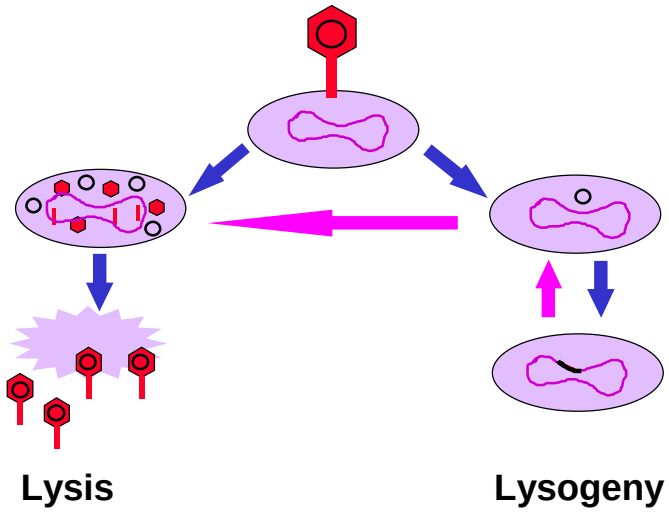

Figure 1– The life cycle of lambda phage. A phage injects its DNA into the *E. coli* host, then either (left) replicates and destroys the host cell – *lysis* – or (right) integrates its DNA into the host DNA – *lysogeny*.

Lambda has been studied extensively because it is one of the simplest developmental switches – systems with multiple possible end states. Its complete genome (~ 50,000 base pairs) was sequenced long before the era of large-scale genome sequencing. The details of which genes are expressed, when, and in what quantity have been studied extensively, as has lambda's host, *E. coli*. As a result, lambda is one of the best-understood organisms in existence. It is also an organism that exhibits probabilistic or stochastic behavior – some infected cells end up in lysis, others in lysogeny. Hence it is a perfect system in which to demonstrate the stochastic framework of gene regulation modeling.

For the present chapter, we shall focus on three key regulatory proteins in lamba – N, cro and the lambda repressor. There are a handful of other proteins that are equally important, but we have chosen only these three for purposes of illustrating the stochastic framework in this chapter.

- **Repressor** is present at high levels in lysogens and represses the expression of all other genes.
- **Cro** (Control of Repressor and Others) is important in the lytic pathway – it inhibits the production of repressor; it controls the production of the proteins that cause viral shell formation, cell lysis, replication of lambda DNA, etc.
- **Repressor** and **cro** are mutually inhibitory.
- **N** is a temporal regulator. Once lambda injects its DNA into the host, N is produced; thus N is produced very early in the life cycle. N facilitates the expression of early genes, genes necessary for both lysis and lysogeny. Late in the life cycle, once the phage has decided between one of the two fates (by expressing Repressor and not Cro, or vice versa), N production is halted, and other specialized proteins are produced – some for lysis, some for lysogeny.

Figure 2 illustrates the concentrations of each of these proteins as a function of time for typical cells – one that ends in lysis and one that ends in lysogeny.

To understand the entire lambda decision process would require a model much more detailed than the one in this chapter. For now, we shall focus almost entirely on N and how it is regulated. The *N* gene lies downstream of lambda's left promoter, P_L (see Figure 3). (A bit of notation: a gene is placed in italics, e.g., *N*; its protein product is placed in normal text, e.g., N.) In the absence of transcription factors, RNA Polymerase (RNAP) can bind to P_L and initiate transcription. (Note that this is one of the key differences between prokaryotes and eukaryotes – in prokaryotes, DNA and RNAP are sufficient for transcription; in eukaryotes, more factors are required.) Thus, when lambda first injects its DNA into the host, N can be produced. Later, as repressor and cro accumulate, these two transcription factors bind to P_L and inhibit further production of N. We shall be modeling production of N as a function of the concentrations of repressor and cro.

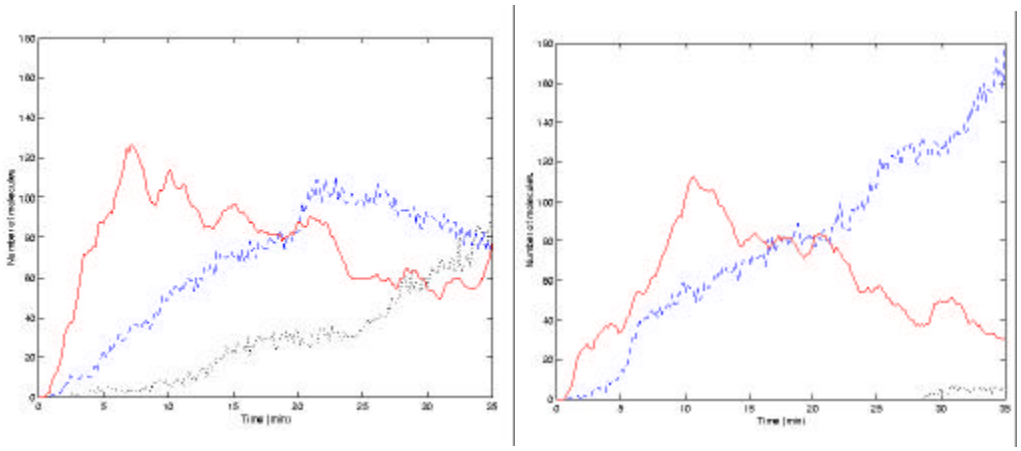

Figure 2– A typical plot of number of proteins as a function of time for the complete lambda model (not presented in this chapter). Solid line = N, dashed line = cro_2, dotted line = $repressor_2$. (a) Lysis, (b) Lysogeny.

1.2 Why a probabilistic model is useful

Lambda DNA is inserted into its *E. coli* host cell. At that time, there are no lambda proteins or mRNA present. Yet, despite this, sometimes the host will go to lysis, other times it will go to lysogeny. To model probabilistic outcomes requires a probabilistic model. A deterministic model would predict that the same initial conditions always lead to the same final conditions.

Figure 4 shows data from a more complete model of lambda, which includes the full interactions of five genes and their protein products. We have simulated protein production and have plotted the final concentration of repressor and of cro at the time when the host cell divides. Each dot represents one phage. Each phage is modeled with the same chemical reactions, the same parameters for those reactions, and the same initial conditions – the only difference between one dot and another is the set of random numbers chosen in our simulation. This is the key point: *in a deterministic model, there would be only one dot on this figure.*

Since the stochastic model and the deterministic model make different predictions, they are equally good for modeling this system. Because the final lysis/lysogeny decision is stochastic (some phages go to lysis, some go to lysogeny) a deterministic model would be inappropriate.

1.3 Origins of stochastic behavior

Where does this probabilistic or stochastic behavior come from? As mentioned in the last chapter, there are trade-offs to consider in modeling. In the limit of low numbers of molecules and short time steps, systems behave stochastically, while in the limit of high numbers of molecules and long time steps, certain averaging occurs and there is a deterministic outcome.

An *E. coli* cell, which acts as the host for lambda, is a rod shaped bacterium 2μm long with a diameter of 1μm (Watson et al., 1987). The volume of an *E. coli* cell is

$$V = \pi r^2 l = \pi/2 \times 10^{-15} \text{ liters.}$$

As will be explained later, significant differences in the amount of binding occur in the range 10^{-9} M to 10^{-7} M. (Remember that M = moles/liter.) The number of molecules that corresponds to, say, 10^{-8} M is

$$(\pi/2 \times 10^{-15} \text{ liters})(10^{-8} \text{ moles/liter})(6 \times 10^{23} \text{molecules/mole}) \approx 10 \text{ molecules}$$

The lambda model deals with 1 molecule of DNA, a few molecules of mRNA, and 10s to 100s of molecules of proteins.

Differential equations *with concentrations as variables* assume that the concentrations vary continuously and deterministically, or at least that the fluctuation around the average value of concentration is small relative to the concentration. Figure 4 showed that these assumptions lead to incorrect predictions.

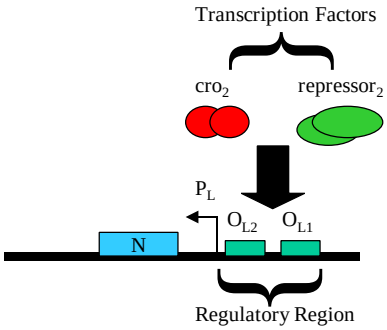

Figure 3– Regulatory logic for gene *N*. Gene *N* lies downstream of promoter P_L. The transcription factors cro_2 and $repressor_2$ bind upstream of P_L at operator sites O_{L1} and O_{L2}. Either transcription factor can bind to either site.

An aside (Molecular Dynamics):

At a lower level, one could (in principle) write out the molecular dynamics equations for each of the molecules involved, solve these deterministic equations, and average over all possible initial conditions. Note that the variables for this approach are *positions and momenta of atoms, not concentrations*. Such an approach would be computationally intractable – first in the shear number of atoms involved (DNA, multiple copies of each protein and mRNA, water molecules, other cellular machinery), second in the time scale (tens of minutes) and finally in the uncertainty of initial conditions – many of the cellular components are not even known.

The molecular dynamics approach to modeling is sometimes called *microscopic* chemistry; differential equations with concentrations as variables are called *macroscopic* chemistry; in between these two lies the stochastic approach, or *mesoscopic* chemistry (Van Kampen, 1992). As in the microscopic approach, one considers individual molecules – proteins and mRNA, for example – but ignores many molecules, such as water and non-regulated parts of the cellular machinery. Rather than considering the positions and momenta of the molecules modeled, one considers the statistics of which reactions occur and how often. The end result is that the statistics (e.g., the probability that there are *n* molecules at time *t*) follow simple laws, as described in the previous chapter.

2 Model

2.1 Outline

The previous chapter mentioned that gene regulation consists of many processes: transcription, splicing, translation, post-translational modifications, degradation, diffusion, cell growth, and many others. Fortunately, many of these processes are not present in general in prokaryotes or in particular in lambda. Our model of the production of N will consider only transcription, translation and degradation. The full lambda model (Arkin et al., 1998) contains cell growth as well, but we shall argue that it can be neglected in this chapter.

The model consists of two parts: TF/DNA binding and N production. The former is modeled as an equilibrium process, according to the principles of statistical thermodynamics explained in the previous chapter. The latter process consists of several individual sub-processes, each of which is modeled as a simple chemical equation. The complete set of chemical equations is found in Table 1, and the parameters for those equations are found in Table 2. Note that these are *mesoscopic* rate constants, i.e., they are the stochastic rate constants that apply to individual molecules. The relationship between these rate constants and the usual *macroscopic* rate constants, which apply to *concentrations* consisting of large numbers of molecules, will be discussed in a later section.

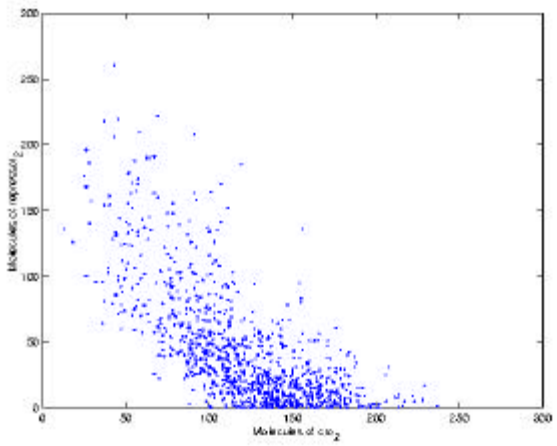

Figure 4– Number of molecules of cro2 and repressor2 at the end of the simulation. Each dot represents one phage.

2.1.1 Transcription

Transcription involves two processes: *initiation* and *elongation*. *Initiation* involves all of the biological details necessary for RNAP to create a transcription complex and start transcription. For this model, these processes will be lumped into Equation 1, with rate constant k_1. The value of k_1 is determined from an equilibrium model, which will be described in detail in a later section.

Elongation is the process, described by Equations 2 and 3 in Table 1, by which RNAP transcribes DNA into mRNA nucleotide by nucleotide. Equation 2 describes the majority of the elongation process, and Equation 3 describes the final step of elongation, namely that the RNAP reaches the final base of the DNA it is transcribing, releases mRNA, and separates from the DNA.

2.1.2 Translation

Translation is mathematically similar to transcription: it consists of an initiation step and an elongation step. The translation initiation rate is not regulated by any transcription factors, so it is simply a constant. The translational elongation process is entirely analogous to the transcriptional elongation process. The equations for translation are 4, 7 and 8 in Table 1.

2.1.3 Degradation

Transcription and translation end in the production of new mRNA and proteins. If these were the only processes present, the amount of mRNA and of protein would grow without limit. Instead, both mRNA and proteins are degraded by various cellular processes. Degradation of mRNA occurs when a ribonuclease (rather than a ribosome) binds to the mRNA and then degrades the mRNA (see Equations 5 and 6 in Table 1.) This competitive binding will be explored further in the Discussion section.

Degradation of a protein occurs when various cellular machinery, such as proteosomes, bind to the protein and degrade it. Equation 9 in Table 1 shows the pseudo-first order version of that process – i.e., we assume the rate depends only on the amount of protein N present, not on the concentration of the degradation machinery. This point is discussed further in the next subsection.

$$RNAP \bullet DNA_{closed} \xrightarrow{k_1} RNAP \bullet DNA_{open,0}$$

$$RNAP \bullet DNA_{open,n} \xrightarrow{k_2} RNAP \bullet DNA_{open,n+1}$$

$$RNAP \bullet DNA_{open,MAX} \xrightarrow{k_3} RNAP_{free} + DNA_{free} + mRNA_{free}$$

$$Ribosome + mRNA_{free} \xrightarrow{k_4} Ribosome \bullet mRNA_0$$

$$RNase + mRNA_{free} \xrightarrow{k_5} RNase \bullet mRNA$$

$$RNase \bullet mRNA \xrightarrow{k_6} RNase$$

$$Ribosome \bullet mRNA_n \xrightarrow{k_7} Ribosome \bullet mRNA_{n+1}$$

$$Ribosome \bullet mRNA_{MAX} \xrightarrow{k_8} Ribosome_{free} + mRNA_{free} + protein$$

$$protein \xrightarrow{k_9} \text{no } protein$$

Table 1 – Chemical reactions involved in gene expression in lambda phage (Arkin et al., 1998).

Constant	Value
k_1	Calculated via binding model (Shea and Ackers, 1985)
k_2	30 s^{-1}
k_3	30 s^{-1}
$k_4 \times$ [Ribosome]	0.3 s^{-1}
$k_5 \times$ [RNase]	0.03 s^{-1}
k_6	Fast compared to k_4 and k_5
k_7	100 s^{-1}
k_8	100 s^{-1}
k_9	0.00231 s^{-1} (half-life = 7min)
Transcription length	550
Translation length	320

Table 2- Values of mesoscopic rate constants in this model (Arkin et al., 1998).

2.1.4 Host cell growth is not important in this chapter

Over the course of about 35 minutes, an *E. coli* host cell will double in size and divide. In general, this is a very important non-linear effect, namely: concentrations are number of molecules per unit volume, so if the volume change is ignored, calculations that involve concentrations will be incorrect. Essentially all reaction rates will involve a rate constant and some power of the volume. *However*, first order reactions (and pseudo-first order reactions) involve the 0^{th} power of the volume, i.e., do not depend on the volume. Thus, Equations 2 through 9 in Table 1 will not be affected by the change in volume.

As for Equation 1 in Table 1, in this chapter, we will assume that the *concentrations* of the transcription factors, cro and repressor, in their active (dimer) forms are constant, rather than the *number* of cro and repressor molecules being constant. Thus Equation 1 is not affected by changes in volume, either. In summary, the host cell growth will not affect our model, so we shall ignore it.

Aside (Pseudo-first order degradation):

As mentioned above, Equation 9 in Table 1 shows the pseudo-first order version of the protein degradation process – i.e., the rate depends only on the amount of protein N present, not on the concentration of the degradation machinery. In other words, the protein degradation machinery is present in constant concentration, regardless of what happens to N, cro, and repressor. A more complete model would include some of the cellular machinery and treat degradation as a second-order effect. In lambda phage, for example, saturation of the machinery that degrades protein cII is thought to be one of the important ways in which lambda decides between lysis and lysogeny (Ptashne, 1992),(Arkin et al., 1998). To model this second order effect, we must explicitly include the degradation machinery in the model.

This chapter will use the pseudo-first order model only. We mention the full second order model only for completeness.

State s	O_L1	O_L2	? G_s (kcal/mol)	i	j	k	$k_1(s)$ (1/sec)
1	-	-	0	0	0	0	0
2	cro_2	-	-10.9	1	0	0	0
3	-	Cro_2	-12.1	1	0	0	0
4	$repressor_2$	-	-11.7	0	1	0	0
5	-	$Repressor_2$	-10.1	0	1	0	0
6	-	RNAP	-12.5	0	0	1	0.011
7	cro_2	Cro_2	-22.9	2	0	0	0
8	cro_2	$Repressor_2$	-20.9	1	1	0	0
9	$repressor_2$	Cro_2	-22.8	1	1	0	0
10	$repressor_2$	$Repressor_2$	-23.7	0	2	0	0

Table 3- Free energies used to calculate k_1 from a transcription factor/DNA binding model.

2.2 Details

This section spells out the details of the model outlined in the previous section.

2.2.1 TF/DNA binding

The promoter for *N* has two binding sites, O_L1 and O_L2 (see Figure 3). Each can bind either cro_2 or $repressor_2$. The possible states of the promoter are shown in Table 3 (Arkin et al., 1998). For each state, we have listed:

- a state number *s*, which is used to discuss the states only; it has no physical meaning
- the configuration of the two binding sites
- the difference in free energy from the ground state (State 1)
- three numbers i, j and k, which are the number of molecules of cro_2, $repressor_2$, and RNAP, respectively,

and finally,

- $k_1(s)$, the rate of transcription initiation, given that the DNA is in state *s*.

The previous chapter mentioned the possibility of measuring rate constants for transitions among the various states of such a model. We shall assume that the transitions have reached equilibrium, and so we ignore kinetics and use equilibrium thermodynamics to find the probability that the promoter is in each of the 10 possible states. Applying the theory from the previous chapter, those probabilities are of the form

$$P_s = \frac{[DNA]_s}{[DNA]_{total}} = \frac{e^{\frac{-\Delta G_s}{RT}}[cro_2]^i[repressor_2]^j[RNAP]^k}{Z} \qquad (1)$$

where the partition function Z is given by

$$Z = \sum_{s,i,j,k} e^{\frac{-\Delta G_s}{RT}}[cro_2]^i[repressor_2]^j[RNAP]^k$$

As mentioned above, we assume that $[cro_2]$ and $[repressor_2]$ are given. We shall use temperature=37 C and [RNAP]=30 nM for all subsequent calculations.

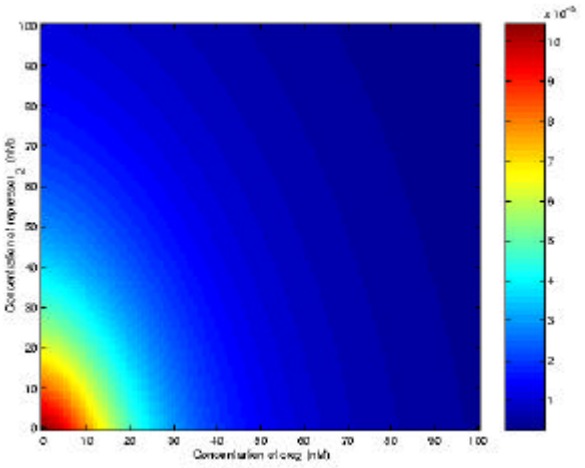

Figure 5– Rate constant k_1 as a function of [cro_2] and [$repressor_2$]. Color represents rate constant, in 1/sec.

Example:

Consider a simple example: [cro_2] = [$repressor_2$] = 0. Then nearly all of the states will have probability 0, according to Equation (1). Only States 1 and 6 have non-zero probability. Using T=37C =310K, [RNAP]=30nM, R=8.314 J/(K mole), and 1kcal = 4184 J,

$$P_1 = \frac{1}{1+19.5} = 0.049 \qquad P_6 = \frac{19.5}{1+19.5} = 0.951$$

Example:

Consider a more complex example: [cro_2] = 50nM, [$repressor_2$] = 10nM. The partition function, Z, evaluated at these conditions, is

$$Z = 1 + 2.41 + 17.0 + 1.77 + 0.13 + 19.5 + 34.9 + 0.27 + 5.9 + 5.1 = 88.0$$

The probabilities of the 10 states are 0.011, 0.028, 0.193, 0.020, 0.002, 0.222, 0.397, 0.003, 0.067 and 0.058, respectively.

Using the TF/DNA binding model, we can calculate k_1. For a given [cro_2] and [$repressor_2$], we calculate the probability P_s of each state *s*. Given the probabilities of finding the DNA in each of the 10 states, it is a simple matter to calculate k_1:

$$k_1 = \sum_s k_1(s) P_s$$

Example:

For the two examples above, $k_1(s)$ is 0 except for *s*=6. Thus, k_1 is simply $0.011 \times P_6$. Numerically, k_1 is 0.0105 s^{-1} for the first example and 0.0024 s^{-1} for the second.

Figure 5 shows k_1 as a function of [cro_2] and [$repressor_2$].

2.2.2 Doing the calculation

Given [cro_2] and [$repressor_2$], we can calculate k_1 as in the previous section. Tables 1 and 2, together with the value of k_1 constitute a system of chemical equations with constant coefficients. In this section, we shall look at how to deal with such a system computationally in the stochastic framework. We shall explain the basic ideas of the calculation only; the detailed algorithms can be found in the references.

2.2.2.1 Probability of reaction per unit time

To model a system in the stochastic framework, assume the system is in a given state, i.e., there are a specific number of molecules of each kind. Then, the *probability that a particular reaction will occur per unit time* is constant. The symbol μ will denote this probability per unit time. In particular, μ is equal to the rate constant, *k*, times the product of the number of molecules of each reactant.

Example:

Consider the protein degradation equation, Equation 9 in Table 1. For protein N, this equation can be rewritten as

$$N \xrightarrow{k_9} \text{no } N$$

From this,

$$m_9 = k_9 \times (\text{number of molecules of } N)$$

Table 2 gave first-order and pseudo-first order rate constants only. First order rate constants have the property that the mesoscopic (i.e., stochastic) rate constant is the same as the macroscopic (i.e., deterministic) rate constant. Second and higher order rate constants do not have this property.

Example:

A second-order *mesoscopic* rate constant has units of 1/(molecules x seconds). A second-order *macroscopic* rate constant has units of 1/(molar x seconds), where molar = moles/liter. Dimensional analysis shows that:

$$k_{macroscopic} = k_{mesoscopic} \times V \times A \times C \qquad \text{for second-order rate constants}$$

where V is the Volume, A is Avagodro's number and C is a dimensionless constant. In all the second order reactions in Table 1, C is 1; in other reactions it may be 1/(2!), 1/(3!), etc. For a more detailed discussion, see (Gillespie, 1977).

2.2.2.2 The Chemical Master Equation (see for example McQuarrie, 1967)

The previous chapter gave an example of using the *chemical master equation*. Essentially, the approach is this:

- Write out all possible states, i.e., all possible combinations of numbers of molecules.
- Use as variables the *probabilities* of being in each of the states. These variables are functions of time.
- Write a system of linear differential equations with constant coefficients (the probabilities of reaction per unit time, or μ's) that express how the variables (the probabilities) change as a function of time.
- Solve this system of linear equations explicitly.

This approach works only if the number of possible states is small. In the example in the previous chapter, the number of possible states was 4. For larger systems, it is sometimes possible to use mathematical techniques to lessen the number of state, but *this is not possible for arbitrary systems.* Thus, for arbitrary large systems, the chemical master equation approach is not feasible.

Example:

Just consider Equation 7 of Table 1. Suppose there is a single molecule of mRNA. Then, since the subscript *n* in Equation 7 can range from 0 to MAX (=320), there are MAX+1 possible states, namely

$$(n = 0), (n = 1), \ldots (n = MAX)$$

Suppose there are two molecules of mRNA. Since they behave independently, there are now about $\frac{1}{2}MAX^2$ states. For *m* molecules of mRNA, there are about $MAX^m/m!$ states **just for mRNA!** Because of this rapid growth in the number of states, the chemical master equation approach quickly becomes unfeasible.

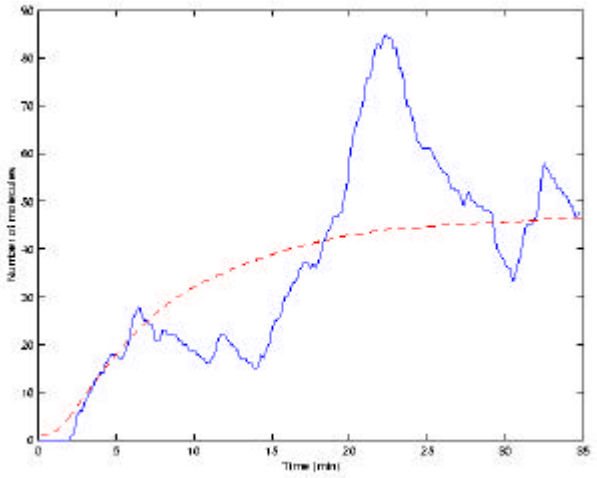

Figure 6– Number of molecules of protein N as a function of time, for the model presented in this chapter with [cro_2] = [$repressor_2$] = 0. Solid line = a typical trajectory. Dotted line = average trajectory.

2.2.2.3 Monte Carlo Algorithms

In the chemical master equation approach, one used as variables the probabilities of being in each state. These variables were continuous and followed simple differential equations. However, the approach became computationally intractable, because the number of variables grew quickly for complex systems. *Monte Carlo algorithms* provide a way around this growth of the number of variables:

- Use as variables the number of molecules of each type. This gets around the combinatorial explosion in number of variables – the number of variables is simply the number of types of molecules.
- Simulate one trajectory through state space:
  1. From each state, there are several possible next states. Pick *one* next state randomly, using the appropriate probability distribution.
  2. Using the new state, pick a new next state.
  3. Go to Step 2 and repeat, until some final time.
- Simulate many trajectories to get the statistics of the system.

So the general idea is: rather than calculating probability distributions explicitly, one uses a computer's random number generator to take samples of those distributions, and by sampling repeatedly one gets a good approximation of the actual distributions. The outline above does not specify how to deal with time, or how exactly to pick the next state. There are different algorithms that solve these problems in different ways.

2.2.2.3.1 Simple approach

A simple algorithm is to use a fixed time step Δt and step forward through time, using a random number generator to calculate whether any reactions occur during the time step. This algorithm has the advantage that the running time is a simple function of Δt. This algorithm has the disadvantage of approximating continuous time by a discrete step. If the time step is small enough, this approximation does not cause any problems, but if the time step is too large, multiple reactions that occur very close together may be missed. There is a trade-off between accuracy of computation and computation time, which comes down to choosing the time step. In particular, the optimal value of Δt is a function of the reaction constants, and may change over time. More complicated schemes, with adaptive time steps, are possible. However, there is another approach, which we now consider, that circumvents this problem and never trades off accuracy.

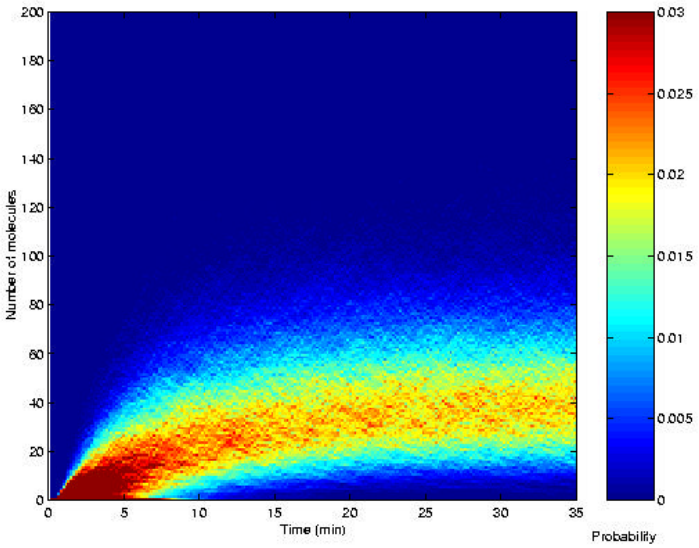

Figure 7– Probability of *n* molecules of protein N at time *t*, for the model presented in this chapter with [cro$_2$] = [repressor$_2$] = 0. Color represents probability.

2.2.2.3.2 Gillespie Algorithm

A different, and in many ways better, approach is that of Gillespie (Gillespie, 1977). His exact simulation algorithm never approximates continuous time by a discrete value; rather, it asks the question "What is the next reaction that occurs, *and when does it occur?*" The first part, which reaction occurs, is a discrete question. The appropriate probability distribution from which to draw the reaction can be calculated efficiently. The second part of the question deals with a time random variable, which is picked from the (exact) distribution, without any discreet approximation of time. In this way, no discrete approximations are made. By using properties of the chemical reactions, one can solve some of the mathematics offline and use those results in the algorithm to make these calculations efficient.

2.2.2.3.3 Gibson/Bruck Algorithm

The authors have developed several improvements to the Gillespie Algorithm, which make it run faster on gene regulation systems (Gibson and Bruck, 1999).

3 Results and Discussion

Using an appropriate algorithm, one may calculate the probability that there are *n* protein copies at time *t*. We start with the simplest case, [cro$_2$] = [repressor$_2$] = 0. We have previously shown that under these conditions, k_1 is 0.0105 s^{-1}. Using the Gibson/Bruck algorithm, generating 5000 trajectories took just over a minute on a pentium 2. More complicated simulations, involving feedback and high degrees of coupling, take longer.

Figure 6 shows a typical trajectory (solid line) and the average trajectory (dotted line). Note that the typical trajectory fluctuates wildly around the average, and that the fluctuations are on the same order as the average; they are emphatically *not* small compared to the average. Note also that the amount of N continues to increase up to and including the final time, because we are not including feedback regulation, in which increasing amounts of repressor and cro cause N production to shut down.

Figure 7 shows the probability that the system has *n* molecules of protein N at time *t*, for each *n* and *t*. The axes are *t* and *n*, and the color represents the probability.

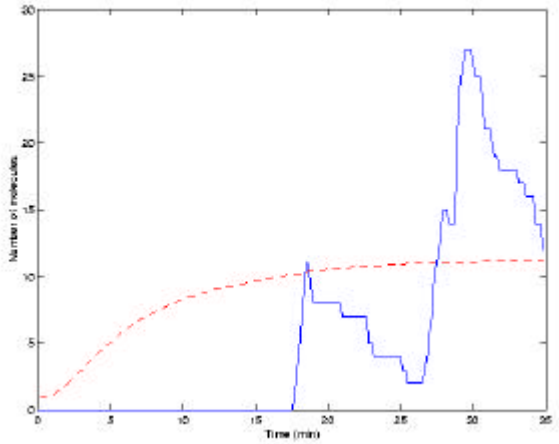

Figure 8– Number of molecules of protein N as a function of time, for the model presented in this chapter with [cro$_2$] = 50nM, [repressor$_2$] = 10nM. Solid line = a typical trajectory. Dotted line = average trajectory.

3.1 Discussion of plots – inhibition by cro, repressor

Cro and repressor inhibit expression of *N*. Consider [cro$_2$] = 50nM, [repressor$_2$] = 10nM. In a previous example, we showed that under these conditions, k_1 is 0.0024 s^{-1}. For this value, generating 5000 trajectories takes under a minute, because on average, there are fewer proteins and hence fewer reaction events to deal with. Figures 8 and 9 correspond to Figures 6 and 7 and use the new k_1 value. These show that cro and repressor do indeed inhibit production of N. This is perhaps easier to see in Figure 9 – increasing the concentration of cro and repressor shifts the probability distribution toward 0 molecules, i.e., the probability of *ever* having 10, 20 or more molecules of N drops. Also, the probability distribution shifts to the right – with inhibition it is likely to take longer to accumulate a fixed number of molecules.

3.2 Adding intuition

One way to deal with the stochastic framework is to take the chemical equations, plug them into numerical software that simulates within the stochastic framework, and look at the results. For some systems, that may be the only thing to do. This section will take another tactic and analyze key sub-systems, in order to gain experience and intuition with the stochastic framework.

There are several ways reactions can depend on each other. The previous chapter considered reversible reactions in equilibrium. Here we consider reactions that compete with each other and reactions that follow each other in succession.

3.2.1 Competition – concurrent reactions

Consider reactions 4 and 5 in Table 1: a single molecule of mRNA can do one of two things – it can bind a ribosome and begin translation, or it can bind ribonuclease (RNase) and be degraded (see Figure 10). We shall now ignore the rest of the model and focus solely on this sub-system. In this special case of a single molecule, we can explicitly write out all the states and use the chemical master equation approach directly.

We shall start by defining the states 0, 1 and 2 as in Figure 10. Let $\vec{P}(t)$ be the probability vector $[P(0\}, P(1), P(2)]^{\mathrm{T}}$. Then, using Markov chains as in the previous chapter,

$$\frac{d\vec{P}}{dt} = \frac{d}{dt}\begin{bmatrix} P(0) \\ P(1) \\ P(2) \end{bmatrix} = \begin{bmatrix} -k_4 - k_5 & 0 & 0 \\ k_4 & 0 & 0 \\ k_5 & 0 & 0 \end{bmatrix} \vec{P}(t) \qquad (1)$$

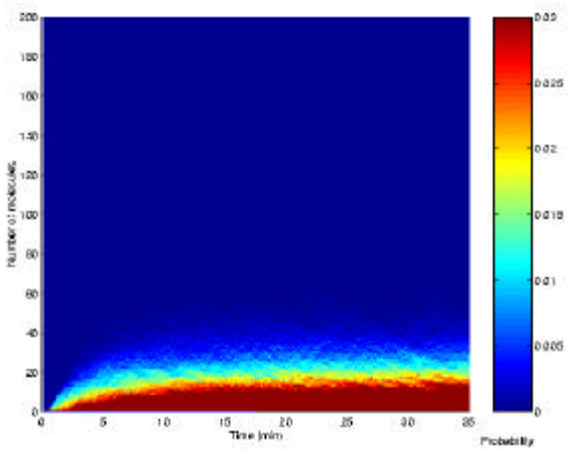

Figure 9– Probability of *n* molecules of protein N at time *t*, for the model presented in this chapter with [cro$_2$] = 50nM, [repressor$_2$] = 10nM. Color represents probability.

Using standard techniques, we can solve this as a function of time. Assuming we always start in State 0, we have:

$$\vec{P}(t) = \begin{bmatrix} P(0) \\ P(1) \\ P(2) \end{bmatrix} = \begin{bmatrix} \exp(-(k_4 + k_5)t) \\ k_4/(k_4 + k_5) \times [1 - \exp(-(k_4 + k_5)t)] \\ k_5/(k_4 + k_5) \times [1 - \exp(-(k_4 + k_5)t)] \end{bmatrix} \qquad (2)$$

Question:

What is the probability that the given mRNA will start translation rather than be degraded, and vice versa?

From (2), it is easy to calculate the probability of eventually winding up in State 1 or 2, respectively: we simply take the limit as $t \to \infty$. Thus, starting at State 0, an mRNA winds up in State 1 with probability $k_4/(k_4 + k_5)$ and in State 2 with probability $k_5/(k_4 + k_5)$. There are easier ways to calculate this than solving the Equation (1) for all possible times; for the sake of this case study, we have chosen to show the time solution as well.

Question:

What is the probability that n *proteins will be produced from one mRNA molecule before it degrades?*

The previous example showed that translation occurs with probability $p = k_4/(k_4 + k_5)$ and mRNA degradation occurs with probability $1 - p = k_5/(k_4 + k_5)$. Using this definition of *p*, the answer is simply:

$$P(n) = p^n(1 - p) \qquad (3)$$

In particular, this is the probability that translation occurs *n* times and then degradation occurs. This distribution (3) is called a *geometric distribution*. The fact that the values of *p* and 1-*p* came from our previous analysis of a Markov chain is irrelevant for answering this question.

Question:

What is the average number *of proteins produced per mRNA transcript?*

It can easily be shown (or looked up) that the expected value of a geometric distribution is *p*/(1-*p*). For the numbers given, *p*=0.3/(0.3 + 0.03)=10/11 and 1-*p*=1/11. Thus, the average number of proteins produced per mRNA is 10. (It might seem surprising that this worked out so cleanly, but in fact, the values of k_4 and k_5 were chosen to make the average number of proteins ten (Arkin et al., 1998).)

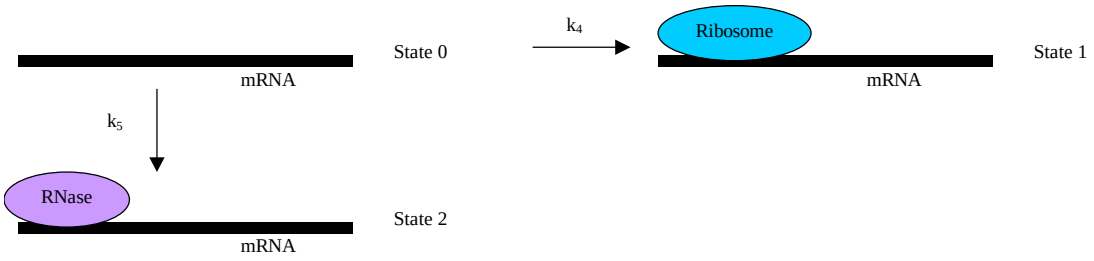

Figure 10– Competitive binding of ribosomes and RNase to mRNA, leading to transcription and degradation, respectively.

3.2.2 Simplification – sequential reactions

Consider once again Equation 7 of Table 1. There are MAX steps of the same type, each of which takes an exponential waiting time (this is a property of continuous time Markov processes). In the master equation approach, this process consists of MAX simple (exponential) reactions. In algorithms that only deal with simple reactions, such as Gillespie's (Gillespie, 1977) exact simulation algorithm, there is no good way to simplify the calculation based on the observation that the interesting parts of Equation 7 are the first step, translation initiation, and the last step, translation termination; *the intermediate states are not interesting.*

The *entire* process of translational elongation constitutes a Poisson process, i.e., a process consisting of sequential steps, with the time of each step exponentially distributed. The number of steps made *up to a fixed time* follows a Poisson distribution, while the time to make *a fixed number of steps* obeys a gamma distribution. The probability of finishing translational elongation at time t, $P_e(t)$ is given by:

$$P_e(t) = \frac{t^{MAX-1}}{(k_7)^{MAX}(MAX-1)!}\exp(-k_7 \times t)$$

Thus a single equation combines MAX separate steps. In this way, we can build up the entire simulation out of a series of more complicated steps, such as the gamma distribution above, instead of a much larger series of simple steps, as in the strict interpretation of Gillespie's algorithm. One advantage that the authors' algorithm has over Gillespie's is the ability to deal with arbitrary stochastic processes, not just exponentials. It should be mentioned, however, that this only works because of the loose coupling between reactions – if the reactions were highly coupled, it might be important to know which of the intermediate states the ribosome was in, and these simplifications would not be appropriate.

3.3 A more complete model

The simple model presented here lacks several subtleties that are necessary for a complete model. Among these subtleties are feedback and second order reactions, notably degradation, which are affected by changes in volume.

3.3.1 Feedback

The model presented is complicated: it contains lots of parts, each of which must be understood individually and in relation to the others. In another sense, though, we have simply shown (in a *very* detailed way) how to write the probability of having *n* molecules of protein N as a function of time, of [cro_2] and of [$repressor_2$], where these latter two quantities are constant. In other words, this is a feed-forward model: the concentrations of cro_2 and $repressor_2$ feed into the regulatory circuitry for N (Figure 3), but the concentration of N does not feed back and affect them.

To get a complete model, it is necessary to write an analogous model for the regulation of cro_2 and $repressor_2$. The tricky part of the complete model is that the regulatory inputs to each of the genes are not constant, but rather, depend on each other. Systems that involve feedback are in general more complex than the open-loop system we have modeled, for this reason. Subsequent chapters will discuss modeling

feedback systems in a deterministic framework – for work on modeling such systems in a probabilistic framework, see Arkin *et al.* (Arkin et al., 1998).

3.3.2 Second order reactions

This chapter has ignored changes in volume, by treating all reactions as first order or pseudo-first order. The probability per unit time of higher order reactions depends on volume, so the growth of the host cell may be an important effect. One can not always ignore growth, although the magnitude of this effect compared to others is hard to quantify. In the lambda genetic switch, there is strong positive feedback of repressor onto itself. As positive feedback can cause amplification, a small error (due to ignoring volume change or otherwise) could, in principle, be magnified to create a large calculation error.

Degradation provides another set of second order reactions. For a first approximation, we assumed that the processes modeled do not regulate the protein degradation machinery and used a pseudo-first order model of degradation. In the Markov chain approach discussed in the previous chapter, this amounts to saying the following: suppose there are *n* copies of protein N at time *t*. Then, at time *t+dt*,

$$P(n-1,t+dt \mid n,t) = n \times k_9 \times dt \qquad \text{and} \qquad P(n,t+dt \mid n,t) = 1 - n \times k_9 \times dt$$

We may explicitly include the degradation machinery. In the simplest case, consider *n* copies of protein N at time *t* and *m* copies of the degradation machinery. At time *t+dt*,

$$P(n-1,m-1,t+dt \mid n,m,t) = n \times m \times k'_9 \times dt$$

$$\text{and} \qquad P(n,m,t+dt \mid n,m,t) = 1 - n \times m \times k'_9 \times dt$$

Here k'_9 is the true second-order rate constant for Equation 9. It has units of 1/(molecules x seconds). Notice that the pseudo-first order approximation consists of treating *m* as constant and letting $k_9 = m\,k'_9$.

In general, and in lambda in particular, second order, non-linear reactions are important. For example, in lambda, saturation of a second-order degradation reaction, the degradation of protein cII, biases the genetic switch toward lysogeny.

4 Conclusion

We have presented an example of the stochastic framework for gene regulation modeling. Once elucidated, the basic ideas are simple, but there are lots of details. Regulatory proteins **cro** and **repressor** bind to DNA and achieve a fast equilibrium. Those proteins affect the rate of transcription initiation. Once transcription initiates, RNA polymerase moves step by step down the DNA and transcribes gene *N* into mRNA. The mRNA can then be degraded by ribonuclease, or it can bind to a ribosome and begin translation. Once translation is initiated, the ribosome proceeds step by step along the mRNA and produces a protein. Proteins in solution can be degraded by a pseudo-first order process or a second-order process, depending on the particular protein and the assumptions made.

All of these details can be written out as chemical equations, as in Table 1. Depending on the number, complexity and coupling of such equations, different techniques can be used to solve them. If there are only a very few equations, or the equations are uncoupled, it may be possible to get an analytical solution. If there are more equations, but they are still uncoupled, it may be possible to use the sort of simplifications mentioned in the chapter to simplify the calculation. In the general case of many, coupled equations, the best approach seems to be a Monte Carlo simulation of the type in Gillespie (Gillespie, 1977) or the type in Gibson and Bruck (Gibson and Bruck, 1999).

There are numerous open areas of research. For larger systems, computational simplifications will be needed. There has not been much systematic work on such simplifications in feed-forward systems, let alone in feedback systems. One important point to be noted is there is no need to reinvent the wheel to analyze each new regulatory system. There is already a significant amount of computer software devoted to solving chemical kinetics in the deterministic limit. Some programs have been developed to do stochastic simulations as well. Ideally certain conventions would be established for specifying the biological details of a gene regulation system in a calculation independent way, and then different programs could do the analysis in different ways.

It is rare to have a system for which so much experimental data is readily available. Even in this system, certain assumptions were made, for example, the promoter data was taken from a related promoter, for which experimental data *is* available. There are several strategies for dealing with systems with missing data:

- Use some sort of statistical technique to estimate (or "fit" or "learn") the missing parameters.
- Measure the missing parameters.
- Use theoretical methods to determine which parameters affect the performance of the model the most; measure those experimentally and estimate the rest.
- Only model systems for which all parameters are available.

Each of these approaches has its strengths and weaknesses. The experimental measurement procedures are still too slow and labor-intensive to provide all the data theorists would like. The parameter estimation procedure is well worked out for deterministic models, but much less so for stochastic models of this sort. We are not aware of any systematic work involving estimation of stochastic parameters for gene regulation systems.

Finally, there are many open theoretical questions. For example, what can one do with a model, other than just simulate trajectories? This question has been considered in detail for deterministic models, but not for stochastic models. The challenge in theoretical biology is to add value – to use mathematics and computation to gain insight into systems that would be impossible (or very difficult) with simpler, more intuitive methods. Stochastic systems defy much conventional intuition. The field is wide open for theoretical advances that help us to reason about systems in greater detail and with greater precision.

Reference List

Arkin, A., Ross, J. and McAdams, H.H. (1998) Stochastic kinetic analysis of developmental pathway bifurcation in phage lambda-infected Escherichia coli cells. Genetics 149, 1633.

Gibson, M. A. and Bruck, J. An Efficient Algorithm for Generating Trajectories of Stochastic Gene Regulation Reactions. Electronic technical report number 026. http://paradise.caltech.edu/ETR.html 1999.

Gillespie, D.T. (1977) Exact Stochastic Simulation of Coupled Chemical Reactions. J.Phys.Chem. 81, 2340.

McQuarrie, D.A. (1967) Stochastic Approach to Chemical Kinetics. J.Appl.Prob. 4, 413.

Ptashne, M. (1992) A Genetic Switch: Phage Lambda and Higher Organisms. Cell Press and Blackwell Scientific Publications,

Shea, M.A. and Ackers, G.K. (1985) The OR control system of bacteriophage lambda. A physical-chemical model for gene regulation. J.Mol.Biol. 181, 211.

Van Kampen, N.G. (1992) Stochastic Processes in Physics and Chemistry. Elsevier Science B. V., Amsterdam, The Netherlands,

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