# 1 Exploring the expression of an unregulated gene

Consider a single gene that is transcribed at a constant rate and degrades at a rate proportional to the number of copies of the RNA present (i.e. each individual RNA molecular has a fixed probability of decaying in any given second). We can express this verbal model as a differential equation that gives the rate of change of the RNA concentration. (We will simplify our thought process by assuming that cell volume is constant, so our units of "concentration" will be molecules per cell.)

$$\frac{dm_i}{dt} = S_i - D_i m_i \tag{1}$$

Where $m_i$ is the concentration of mRNA $i$ (in molecules per cell), $S_i$ is the transcription rate of gene $i$ (in molecules per second per cell), and $D_i$ is the degradation rate constant for mRNA $i$ (in units of "per second"). This equation can be integrated in closed form to give the concentration $m_i$ as a function of time. Dropping the subscript $i$ for convenience, the result is:

$$m(t) = \frac{S}{D} - (\frac{S}{D} - M_0)e^{-tD} \tag{2}$$

where $M_0$ is the initial concentration of the mRNA (you have to know the starting value to compute later values). Notice that the equilibrium concentration – the limit as $t$ goes to infinity – is $\frac{S}{D}$. It makes sense that if the transcription rate increases and the degradation rate stays the same, the equilibrium concentration increases. Similarly, if degradation rate increases while the transcription rate stays the same, the equilibrium concentration decreases. Mathematica will do this integration for you.

# 2 Modeling transcription and translation of an unregulated gene

Let us assume that the rate of production of the protein encoded by gene $i$ is proportional to the amount of mRNA available. This implies that the availability of ribosomes is not limiting. We also assume that the translation rate is unregulated, and that there are no global changes in translation rate (any of these assumptions may need to be reconsidered for a particular modeling problem). We also make the same assumptions for protein degradation that we did for RNA degradation – that each protein molecule has a fixed probability of being degraded in each second. These assumptions lead to the following differential equation:

$$\frac{dp_i}{dt} = T_i m_i - D_i^p p_i \tag{3}$$

where $T_i$ is the translation rate constant for mRNA $i$ and $D_i^p$ is the degradation rate constant for protein $i$ (the superscript $p$ is meant to distinguish it from the mRNA degradation rateconstant).

**Things to notice**

- This model has a separate translation rate constant for each mRNA and a separate degradation rate constant for each mRNA and each protein. In reality these constants do vary from gene to gene. However, the degradation rate constants vary more than the translation rate constants, so if it were necessary to simplify the model one could assume that all mRNAs are translated at the same rate.
- The rate of change of the concentration of protein $i$ depends on the concentration of mRNA $i$, which means that the differential equation for the protein is *coupled* to the differential equation for the mRNA.

To compute the concentration of protein $i$ as a function of time we could substitute (2) for $m$ in (3) and then integrate analytically, producing a formula for $p_i(t)$. (The Mathematica notebook shows this.) However, most non-trivial problems involving gene regulation cannot be integrated analytically, so we will focus on numerical integration. The basic idea behind numerical integration starting from initial values of the concentration variables is to discretize time into tiny steps. The derivatives at time zero are calculated from the initial values according to the given differential equations. They are then used to calculate the concentrations at the next time step. This process is repeated through the entire integration interval. The idea is simple but there are a number of complexities that we will gloss over for now, trusting Mathematica to know what it's doing.

# 3 Dynamics of a repressor and its target

Transcription factors (TFs) are DNA-binding proteins that bind specific sequences and affect the rate at which genes are transcribed. In bacteria and yeast the binding of a TF is generally thought to influence transcription of a gene (or a pair of divergently transcribed genes) whose transcription start site is within a few hundred to a thousand bases of the binding site. When a TF binds to a site in the promoter of a gene it either represses or activates transcription of the gene (occasionally a TF may activate a target in some conditions and repress it in others).

We would like models that allow us to predict the transcription rate of gene $i$ as a function of concentrations of the TFs that regulate it. In the simplest case, gene $i$ is regulated by a single repressor protein, $j$. A simple model is

$$S_i(p_j) = \frac{V_i}{1 + \theta_{i,j} p_j} \tag{4}$$

where as $S_i(p_j)$ is the transcription rate of gene $i$, as in (1). $V_i$ is the transcription rate in the absence of repressor (that is, when $p_j = 0$) and $\theta_{i,j}$ represents the efficiency with which protein $j$ represses transcription of gene $i$. Specifically, protein $j$ represses transcription of gene $i$ to half of $V_i$ at a concentration of $1/\theta_{i,j}$. This model has some reasonable properties, in that there is no repression when $p_i = 0$ and complete repression as $p_i$ goes to infinity.

There is an argument as to why this repression model makes biochemical sense for a single TF binding site in "Goutsias, J. and S. Kim. (2004). A nonlinear discrete dynamical model

for transcriptional regulation: construction and properties. Biophys J: 86(4): p. 1922-45." If TF $j$ can bind independently to any combination of $n$ sites in the promoter of gene $i$ (that is, binding at one site neither facilitates nor inhibits binding at another site), Goutsias and Kim argue, the transcription rate function should be:

$$S_i(p_j) = \frac{V_i}{(1 + \theta_{i,j} p_j)^n} \tag{5}$$

Models for activation are necessarily a little more complicated – we'll talk more about these later, too.

# 4 The bistable switch

One of the best known examples of a rationally designed and successfully implemented transcriptional regulatory circuit is the bistable switch, described in "Construction of a genetic toggle switch in Escherichia coli," by Gardner et al. Please read that paper.

## 4.1 The model in dimensionless units

The box in the paper presents a very reduced model of protein production and its regulation. The model does not separate transcription and translation, which is fine as long as translation and degradation of mRNAs is relatively quick. In that case the need to translate mRNAs before you get protein imposes relatively little delay. Similarly, the time between a decrease in mRNA production and the corresponding decrease in protein production is not greatly delayed by the time it takes existing mRNAs to decay.

To understand the equations in the box, we begin with a simple model of protein production and its regulation, similar to those we have used for mRNA.

$$\frac{dp_1}{dT} = \frac{a_1}{1 + (\theta_2 p_2)^\beta} - D p_1 \tag{6}$$

Here $p_1$ is the concentration of protein 1, $T$ is time in seconds, and $a_1$ is the maximal rate of protein production (in the absence of any repressor protein). $\theta_2$ is one over the concentration of protein 2 (a repressor) required to reduce the production rate of protein 1 to half its maximal value, when $\beta = 1$. $p_2$ is the concentration of repressor protein $p_2$. $\beta$ is the "cooperativity" in the repressive effect of different concentrations of $p_2$. Such cooperativity may arise because the promoter of gene 1 has multiple binding sites for $p_2$ or because multiple copies of the protein $p_2$ must bind to each other before the complex binds to the promoter. This model is fairly neutral about the physical causes of the cooperativity. $D$ is the effective degradation rate constant (which lumps together the mRNA degradation rate constant, the protein degradation rate constant, and the dillution rate due to growth). $D$ is assumed to be the same for both proteins.

The paper uses a "dimensionless" version of this equation, which can be derived as follows. First, define $t = TD$.

$$\begin{aligned}\frac{dp_1}{dt} =& \frac{dp_1}{dT}\frac{dT}{dt} \\ =& \frac{dp_1}{dT}\frac{1}{D} \\ =& \frac{a_1/D}{1+(\theta_2 p_2)^\beta} - p_1\end{aligned}$$

This is a rescaling of time so that, if degradation happens quickly the clock ticks more often, and vice versa, making degradation appear to happen with a rate constant of 1 per unit time.

The next step is to rescale the protein concentration by defining $u = \theta_1 p_1$ and $v = \theta_2 p_2$. Then

$$\begin{aligned}\frac{du}{dt} =& \frac{du}{dp_1}\frac{dp_1}{dt} \\ =& \theta_1 \frac{dp_1}{dt} \\ =& \frac{a_1\theta_1/D}{1+(\theta_2 p_2)^\beta} - \theta_1 p_1 \\ =& \frac{a_1\theta_1/D}{1+v^\beta} - u\end{aligned}$$

This is a rescaling of protein concentration so that if the protein is a powerful repressor (large $\theta$) then we represent that by saying there is more protein. The power of the protein to repress is thus absorbed into the variable describing the protein concentration.

The final step is to define $\alpha_1 = a_1\theta_1/D$, yielding

$$\frac{du}{dt} = \frac{\alpha_1}{1+v^\beta} - u \tag{7}$$

This is a rescaling of the maximum production rate to the new measures of protein concentration and time.

To derive the equation for $dv/dt$ you can go through the same process, swapping the roles of $u$ and $v$ and substituting $\gamma$ for $\beta$.

This process of rescaling to reduce the number of parameters preserves qualitative properties. For example, if there exists a certain number of stable equilibria in the rescaled system then there exist corresponding equilibria in the original system. If you want to compare modeled quantities to measured quantities you must rescale back to standard units.

# 5 The repressilator

There is quite a lot of interest in designing and implementing regulatory circuits with interesting dynamic properties. One such property is sustained oscillation, in which the concentration of an

mRNA or protein repeatedly rises and falls. There are several ways to implement such a system, including two-gene systems that include both activation and repression. The best known, however, is the cleverly named "repressilator," first described in "Elowitz, M.B. and Leibler, S. 2000. A synthetic oscillatory network of transcriptional regulators. Nature 403(6767): 335-338." The repressilator consists of three genes, each of which represses another and is repressed by another in a cycle: $A \dashv B \dashv C \dashv A$. Elowitz et al. built such a system in *E. Coli* and, at least in some cells, it appeared to oscillate in a sustained and regular fashion. They also developed and analyzed a mathematical model for a repressilator consisting of three genes that are identical in every way, except for which other gene they repress and which one they are repressed by. They showed that, depending on the parameters, the system might have damped oscillation, in which case the pulses decrease in size and eventually die out, or undamped oscillation. If the translation rate is held fixed, the key parameters that determine this are the ratio of RNA degradation rate to protein degradation rate and the "cooperativity" of the repression, corresponding to the $n$ in (5).

**Exercise 5.1.** The box in the repressilator paper presents a model that is somewhat more complicated than the one in the bistable switch paper, though in a similar spirit. To obtain this dimensionless system, one must carry out a derivation that is similar to the one shown above for Equation (7). However, another variable change or two is required. To see how that works, please start with the model in standard units:

$$\begin{aligned}\frac{d\mu_i}{dT} =& \frac{a}{1+(\theta\pi_j)^n} + a_0 - \delta^m \mu_i \\ \frac{d\pi_j}{dT} =& \tau\mu_j - \delta^p \pi_j\end{aligned}$$

where $\mu_i$ is the mRNA concentration of the $i^{th}$ gene (in mRNA molecules per unit volume, but we'll omit the volume units since volume is fixed). $T$ is time (in seconds), $a$ is the unrepressed transcription rate (in mRNA molecules per second), $\theta$ is the repression efficiency (in "per protein molecule"), $\delta^m$ is the mRNA degradation rate constant (in "per second"), $\pi_j$ is the protein concentration of the $j^{th}$ gene (in protein molecules), $n$ is the cooperativity factor, $\tau$ is the translation rate constant (in protein molecules per mRNA molecule per second), and $\delta^p$ is the protein degradation rate constant (in "per second"). All the constants, $a$, $a_0$, $\theta$, $\delta^m$, $\tau$, and $\delta^t$, are taken to be the same for all three genes.

To derive the dimensionless form shown in the paper, you must carry out several variable changes, defining $t$ in terms of $T$ and one or more of the constants, $m_i$ in terms of $\mu_i$ and one or more of the constants, $p_j$ in terms of $\pi_j$ and one or more of the constants, $\alpha$ in terms of $a$ and constants, and $\alpha_0$ in terms of $a_0$ and constants. All of these new variables should be dimensionless, meaning that the units of all the constants and variables in their definitions cancel out. If you read the text in the gray box of the paper, right below the equations, it basically tells you what the definitions of the dimensionless variables are. When you have the right dimensionless definitions, you will use the "chain rule" of calculus several times to calculate $dm_i/dt$ as a function of $d\mu_i/dT$ and $dp_j/dt$ as a function of $d\pi_j/dt$. All of this is very similar to the derivation of (7).

Hints: You can leave the definitions of $\alpha$ and $\alpha_0$ until after you've taken care of $t$, $\mu$, and

$\pi$. At that point it will be obvious what to do. At the very end, you can put in $\beta$ to replace $\delta^p/\delta^m$.

# 6 Determining parameters in a natural regulatory network

When one wants to model a specific biological system involving transcriptional regulation of some set of genes, it is not difficult to write down a set of generic equations to describe the system. We saw in the exercises that, if we know the parameter values (e.g. rate constants, etc.) and we know the concentrations of the mRNAs and proteins at some time $t_0$, we can use a numerical integrator to predict the mRNA and protein concentrations at any time after $t_0$. The most difficult part of a real modeling problem is not writing down or integrating the equations but determining the parameter values. Sometimes some or all of the parameters can be measured directly, especially if one is willing to settle for *in vitro* measurements. However, conditions *in vitro* may be quite different from those *in vivo*, so *in vivo* measurements are often preferred. Furthermore, *in vitro* biochemical measurements are generally painstaking, involving the purification of components, and so they may be difficult to scale up for efficient, systematic elucidation of transcriptional regulation systems. As a result, there is a great deal of interest in determining model parameters by fitting the generic model to measured states of a system.

Technologies for measuring the states of cells are changing all the time, but for the last ten years or so it has been much easier to measure mRNA concentrations than protein concentrations, especially if you want to measure a lot of molecules at once. In fact, the concentrations of all mRNAs in a cell can be measured with reasonably good accuracy at a relatively low cost by using micrarrays or, more recently, by high throughput, short-read sequencing (RNA-seq). So a natural question is, "How far can you get in estimating the parameters of a transcriptional regulation system by measuring intracellular mRNA concentrations?" We will consider some simple cases of parameter estimation from mRNA concentrations in this section.

When thinking about parameter estimation for transcriptional networks, the first consideration is the degree to which the structure of the network – i.e. the set of proteins that regulate each gene – is known in advance. Sometimes the main goal is to learn the structure, while learning the parameters is secondary. In other cases, the structure is assumed to be completely known and the main focus is learning parameters. In intermediate cases, one assumes that most of the structure is known but allows the possibility that a few novel regulatory relationships may be discovered. A second consideration is whether the data to be used for parameter estimation is taken at steady state, where the derivatives of the concentrations are zero, or in a dynamic situation.

In the Kuttykrishnan et al. paper we find an example of how to approach this problem.

## 6.1 Parameter estimation for a repressor system at steady state

Consider a population of cells that is not growing or changing in volume. If the cells are given an external stimulus, such as the addition of glucose to the medium, they may respond to that stimulus by changing the rates at which various genes are being transcribed. Once the new transcription rates are in effect, the concentrations of mRNAs and proteins will change for a while. Eventually, a new equilibrium will be established in which, for each mRNA and protein, the long run average of its synthesis rate equals the long run average of its degradation rate. This equilibration happens because the degradation rates are proportional to concentrations. If the degradation rate is less than the synthesis rate then the concentration goes up, increasing the degradation rate. If the degradation rate exceeds the synthesis rate then the concentration goes down, reducing the degradation rate. The concentrations may oscillate, as we saw for the repressilator, but the long run averages have to match. If a population of oscillating cells is large and the cycles have not been synchronized each cell will be in a random part of its cycle. As a result, the average concentration over all these cells is equivalent to the average concentration in one cell over a long period of time. Thus, the degradation rate at a particular moment, averaged over a large population of unsynchronized cells, must be equal to the synthesis rate averaged over the same population.

If the number of cells is increasing through division then the total volume is increasing. This decreases all concentrations by a constant factor, so instead of (1) we have

$$\frac{dm_j}{dt} = S_j - D_j m_j - \delta m_j$$

where $D_j$ is the true mRNA degradation rate and $\delta$ is the dilution rate. However, for notational simplicity we will use $D_j$ to refer to the combined degradation and dilution rates. (This doesn't affect anything unless the true degradation rates are measured directly in cells with a different dilution rate. In that case, the dilution rate must be added in directly.) Thus, the synthesis and degradation (i.e. degradation plus dilution) rates must still match:

$$S_j(p_1 \dots p_n) = D_j m_j \tag{8}$$

$$T_j m_j = D_j^p p_j \tag{9}$$

where $S_j(p_1 \dots p_n)$ indicates that the transcription rate of gene $j$ may depend on the concentrations of any combination of other proteins. For now, we will write the equations as if we had no prior knowledge of which proteins are TFs and which of those TFs regulate each gene.

These steady state equations can also be derived directly from Equations (1) and (3) because, by definition, the time derivatives are zero at steady state.

To see how these steady state equations can be used in parameter inference, let's adapt our simplest regulation model, Equation (4), for the effect of multiple, independent repressors on $S_i$.

$$S_i(p_1 \dots p_n) = \frac{V_i}{\prod_{j=1}^{n}(1 + \theta_{i,j} p_j)} \tag{10}$$

In a particular application, the prior knowledge that a particular protein $p_j$ does not regulate $S_i$ can be encoded by fixing $\theta_{i,j} = 0$ rather than allowing it to be estimated from data. If $p_j$ is known not to be a TF then $\theta_{i,j}$ is fixed at $0$ for all $i$.

If we assume protein concentrations are not easily observable, it is desirable to eliminate $p_j$ from the system of equations. We can do this by solving (9) for $p_j$ in terms of $m_j$, $T_j$, and $D_j^p$, and then substituting for $p_j$ in (10). The same can be done for each $j$ and each $i$, leaving us with just one equation for each gene:

$$\frac{V_i}{\prod_{j=1}^{n}(1 + \theta_{i,j}\frac{T_j}{D_j^p}m_j)} = D_i m_i \tag{11}$$

This system of equations predicts mRNA concentrations at steady-state without referring to protein concentrations. Note that this only works at steady state, where the concentration of each protein is proportional to the concentration of its mRNA (under our model). These equations can be applied to determining how many distinct steady states a system has and to determining how the steady states change when certain RNA concentrations are held fixed by external manipulation – e.g. deleting the gene from which an RNA is transcribed. These are now algebraic rather than differential equations and they cannot be used to predict the state of a system at any particular time.

Notice that Equation (11) depends on the ratio $\theta_{i,j}\frac{T_j}{D_j^p}$ but it does not depend on the specific values of the three parameters. Therefore, we can define a single parameter $c_{i,j}$ by

$$c_{i,j} = \theta_{i,j}\frac{T_j}{D_j^p}$$

and substitute into Equation (11), leaving the simpler set of equations

$$\frac{V_i}{\prod_{j=1}^{n}(1 + c_{i,j}m_j)} = D_i m_i$$

To simplify further, we can define $b_i = \frac{D_i}{V_i}$, so

$$\frac{1}{\prod_{j=1}^{n}(1 + c_{i,j}m_j)} - b_i m_i = 0 \tag{12}$$

To use this system of equations, we need only measure $m_1 \ldots m_n$ at steady state, substitute into the $n$ equations of the form (12), and estimate $c_{i,j}$ and $b_i$ for each $i$ and $j$. However, there are two complications. The first is that the system is not linear. This means that getting an exact, closed form solution for the parameters might be difficult and ugly. There are symbolic algebra programs that can solve these equations, but in most cases it is probably more practical to use a numerical solver to estimate the values of the parameters that minimize the total error – that is, the sum of squares of the left hand sides. For an exact solution the error will be zero, but a numerical solver may not get an exact solution.

The second complication is that there are more unknowns than equations. Specifically, if $k$ of the $n$ genes are TFs then there are $n + nk$ unknown parameters, but there are only $n$

equations. The system is massively under determined. More data are required to obtain the parameter values. One way to get more data is to measure steady-state mRNA levels in a cell where one or more of the TFs in the network has been deleted by genetic manipulation. Deleting one TF creates a new system of $n-1$ equations (the equation for the gene that was knocked out doesn't apply in the modified system). Deleting gene $j$ sets $m_j$ to zero, but it leaves the parameters to be estimated unchanged. Therefore, these $n-1$ new equations can be added to the $n$ equations for the original network, providing further constraint on the values of the parameters. If the deleted TF had a non-zero concentration in the measurements of the original system then the new measurements will generally provide new information. If it is zero in the original measurements then it is probably better to explicitly eliminate the associated variables from the problem until there is a non-zero measurement for that TF.

If each of the $k$ TFs is individually knocked out and mRNA measurements are made, this adds $k(n-1)$ additional equations for a total of $n+k(n-1)$, which is still $k$ less than the $n+nk$ unknowns. In practice, the best way to approach this is to assume there is no auto-regulation by fixing the $k$ parameters describing the effect of each TF on itself to be zero. Although autoregulation does exist, it's effects cannot be estimated simply by gene deletion.

The number of parameters can also be reduced by assuming that the maximum observed concentration of $m_i$ is the maximum possible concentration, which under the model should be $1/b_i$. This assumption is sometimes justified on the grounds that we must have observed a pretty broad sample of the possible states of the system, or else the parameter estimation problem would be impossible. I find this to be a pretty risky assumption, but to simplify the exercise below we will go with max-observed-is-max-possible assumption.

It would also be useful to get more constraints by creating some strains in which multiple expressed TFs are knocked out. This is attractive mathematically in that there are plenty of combinations of TFs available to provide the necessary constraints. Each double mutant provides another $n-2$ constraints.

## 6.2 Application to specific pathways

So far, we have assumed that there is no prior knowledge about which TFs regulate which targets. This is the genomic ideal, where all data is obtained by global, high-throughput measurements and all knowledge is extracted automatically from the high-throughput data. Humans are needed only to applaud and to find applications of this trove of automatically generated knowledge. In practice, however, most research is focused on elucidating specific pathways about which a good deal is already known. In a typical application of network inference, some TFs are known to regulate some targets relevant to a biological phenomenon of interest. Scientists usually proceed by positing new regulatory relationships only when compelled to by the data. In the absence of compelling data to the contrary, the known TF-target relationships are assumed to be the only significant relationships. The goal, then, is to obtain quantitative parameters for the relationships that are already believed to exist and to identify any new relationships that are required to explain the data.

Given that we have algorithms for inferring regulatory networks *de novo*, what advantage, if any, is conferred by the assumption that the structure of the network is already known? The advantage is simplification of the parameter optimization problem. As we have seen, identifying parameters of complex regulatory models poses many challenges. The objective functions are generally non-convex, so finding the global minimum may require a great deal of random search or, depending on the size of the problem, it may be completely infeasible. Even finding local minimima can be challenging. What's more, the solutions that are returned by numerical optimization algorithms may not be stable, in the sense that, if you iterate the process of identifying a model, simulating data from that model, and identifying a new model from the simulated data, the solution may change drastically on each iteration. Assuming that the structure of the network is known can greatly simplify the optimization problem, sometimes to the point where no optimization is needed at all.

Consider a network structure where gene $i$ is known to be regulated by TF $j$ and no others. If we have measurements of the RNA concentrations in the wild-type strain ($m_i^{WT}$ and $m_j^{WT}$) and in a strain with gene $j$ deleted ($m_i^{\Delta j}$) then the parameters for gene $i$ can be determined by solving the following simple system of two equations:

$$b_i m_i^{WT}(1 + c_{i,j} m_j^{WT}) = 1 \tag{13}$$

$$b_i m_i^{\Delta j} = 1 \tag{14}$$

These equations can be described as second order and first order, respectively, where the order of an equation is the maximum number of parameters that are multiplied together in a single term. This system can easily be solved exactly by substitution. No optimization is required, so the solution is exact and unique. Furthermore, the estimation of $b_i$ and $c_{i,j}$ only depends on three mRNA measurements, so errors in any of the other measurements will not propagate to these parameters.

If gene $i$ is known to be regulated by exactly two TFs, $j$ and $l$, and we have measurements in the wild-type and both single deletion strains ($\Delta j$ and $\Delta l$), then the parameters for transcription of gene $i$ can be obtained by solving three equations in three variables.

$$b_i m_i^{WT}(1 + c_{i,j} m_j^{WT})(1 + c_{i,l} m_l^{WT}) = 1 \tag{15}$$

$$b_i m_i^{\Delta l}(1 + c_{i,j} m_j^{\Delta l}) = 1 \tag{16}$$

$$b_i m_i^{\Delta j}(1 + c_{i,l} m_l^{\Delta j}) = 1 \tag{17}$$

There are still exactly enough constraints to determine the parameters, but the equations are now third order, second order, and second order, respectively. This makes solving them considerably harder. This system can be simplified by adding measurements in a double deletion strain ($\Delta jl$), yielding an additional equation: $b_i m_i^{\Delta jl} = 1$. This additional equation provides an extra constraint, which may help compensate for measurement error, but it also adds expense to the experimental plan.

In summary, for every TF that is known not to regulate a target $i$, the number of simultaneous equations that need to be solved to determine the parameters for transcription of gene $i$ is reduced by one, and the orders of all the remaining equations are also reduced by one. This

both facilitates optimization and contains the spread of measurement errors. The structure of the network can also guide the selection of mutant strains that further simplify or even eliminate the optimization.

A possible down side of assuming known network structure is the inability to discover novel, previously unknown network edges. To overcome this, there must be some process of reasoning about the results outside the formal procedure that assumes a known network. One way to approach this is to compare the residual error under the given network structure to the error under another structure with a novel edge. If the novel edge reduces the residual error enough then that edge is added to the model. This may seem very similar to fully automated network inference with no known structure, but there are differences. Starting with an acceptable model, one would tend to accept a new edge only if it reduced the error by a relatively large amount. In other words, there is a strong bias against modifying the model. In addition, if the only modifications considered are additions of an edge (no deletions, swaps, or simultaneous changes to multiple edges) then the number of model comparisons is relatively small. When starting with a "blank slate" and searching the entire space of possible models many more comparisons must be made, many more modifications must be accepted to get to a decent model, and a lower threshold must be set for accepting a modification. The difference is essentially one of a local search in the neighborhood of an acceptable model versus a global search with no specific criteria for what makes a model good enough.

Relying on error reduction to determine whether a model should be changed poses a problem: How much reduction is enough to justify a change? Theoretically, there are ways of trying to figure out whether an error reduction is statistically significant – i.e. unlikely to be due to sampling error (not having enough data). But a statistically significant reduction in error is not enough to justify adding edges, and hence parameters, to a model. Ordinary scientific practice requires that changes to a model be supported by a sufficient weight of evidence. This is why regularization terms that penalize additional edges are often added to the residual error calculation. However, the choice of regularization terms is somewhat arbitrary and does not tend to reflect the choices scientists make as well as one might like. An alternative to deciding whether a structural modification is warranted entirely on the basis of error reduction is to consider the effect on parameter estimates. For example, suppose that adding edge $e_1$ to an acceptable model structure and reestimating parameters reduces the residual error by an amount $x$. It does so by assigning a weak effect to edge $e_1$ and also making small modifications to a large number of other parameters. Suppose that adding edge $e_2$ instead of $e_1$ and reestimating parameters also reduces the residual error by $x$. However, it does so by assigning a strong effect to edge $e_2$ and leaving the other parameters unchanged. The addition of edge $e_2$, which has a localized effect on parameter values, is much more appealing to most people.

There is a similar preference for locality in experimental evidence. Suppose a model is consistent with a body of data collected from a range of deletion strains, but a new strain consistently yields results that substantially increases the residual error or yields substantially different estimates for a parameter. That feels like strong evidence that there is something wrong with the model. If, on the other hand, a model is slightly inconsistent with a large body of experiments, the total error it produces may be the same but there is no single experiment that can be

pointed to as the source of inconsistency. This feels like much weaker evidence that there is something wrong with model. Another way to think about this is that people prefer to hang on to their current model unless there is a clear culprit – a parameter or two that must be changed, a clear best choice among alternative models, and a distinct body of data that can be pointed to as justifying the change.

Having articulated these preferences for changing a model when the discrepancies between the model and the data are localized to a few variables and experiments, we can try to quantify it in various ways. For example, if we have more data than are needed to determine the parameters, we can estimate the variance in parameter estimates by randomly sampling subsets of the data and reestimating parameters for each subset. If an alternative structure reduces the variance in parameter estimates, even without reducing the residual error, that model is to be preferred. Perhaps better than looking at the total variance in parameter estimates would be to look at the maximum variance across all parameters. Whatever measure we choose, though, we are more likely to observe localized discrepancies between specific data and specific parameters using the lower order equations we get by having fewer regulators of each gene.

Another important difference between the blank-slate, genomic inference problem and the inference problem starting from an acceptable structure is that the acceptable structure can guide the design of an informative set of experiments. Even if those experiments produce data that lead to modification of the structure, the guidance it provides for experimental design is useful. Starting with an acceptable model moves one from the realm of "discovery based" science, in which the data are all generated prior to analysis, into the realm of hypothesis-driven science, where the choice of experiments is guided by a (formal or informal) model.

# 7 General model of transcriptional regulation

In 1985, Shea and Ackers introduced a general model of transcriptional regulation that is often called "the thermodynamic model". Within this framework, one can model a promoter containing any number of binding sites for any number of transcription factors, including both repressors and and activators, as well as interactions among the proteins bound to the different sites. Given the arrangement of binding sites in a promoter and all the necessary parameter values, this model predicts transcription rate at the promoter as a function of the concentrations of all the transcription factors. Thus, it models the same type of thing as (5) but covers a much more general set of situations.

## 7.1 The RNA Polymerase

One key idea behind the model is that the transcription rate is determined by the rate of transcription initiation at the transcription start site (TSS), where the RNA polymerase (RNAP) initially binds to the DNA. There is assumed to be some fixed probability of initiating transcription in any given instant during which the RNAP is bound to the promoter. Thus, the

expected transcription rate is proportional to the expected fraction of time during which the RNAP is bound – i.e. the probability of TSS being bound by RNAP at any given instant. The proportionality constant is the probability that bound RNAP will initiate transcription (assumed to be an irreversible process). This constant, though it is not known in advance of parameter fitting, is assumed to be the same for all promoters in the genome of a given organism. Thus, the focus is on the probability of the TSS being bound by RNAP, as a function of the promoter architecture and the concentrations of TFs.

The other key idea (and the reason the model is called "thermodynamic") is the assumption that all reversible binding reactions are approximately in equilibrium. For example, consider the reversible binding of the RNAP to the TSS portion of a promoter's DNA, assuming no other proteins are present. We can write this reaction as:

$$\text{RNAP} + \text{DNA} \underset{k_2}{\overset{k_1}{\rightleftharpoons}} \text{RNAP·DNA}$$

where $k_1$ and $k_2$ are the rate constants for the binding and unbinding reactions, respectively. At equilibrium, the rates of the forward and reverse reactions are equal and the following relation, called "the law of mass action," holds:

$$\frac{[\text{RNAP·DNA}]}{[\text{RNAP}][\text{DNA}]} = K_a \tag{18}$$

where [RNAP] and [DNA] represent the concentrations of unbound RNAP and promoter DNA, [RNAP·DNA] represents the concentration of the complex, and $K_a$ is a constant. $K_a$ is called the *assiciation constant*. One intuitive way to think about this (although it may not be the most modern, physical way, see Wikipedia article on mass action) is that the rate of the forward reaction and the rate of the reverse reaction are each proportional to the concentrations of the reactants. At equilibrium, where the forward and reverse reaction rates are the same,

$$[\text{RNAP}][\text{DNA}]k_1 = [\text{RNAP·DNA}]k_2$$

so that

$$\frac{[\text{RNAP·DNA}]}{[\text{RNAP}][\text{DNA}]} = \frac{k_1}{k_2}$$

and $K_a = k_1/k_2$. Regardless of the derivation, equation (18) implies

$$\frac{[\text{RNAP·DNA}]}{[\text{DNA}]} = K_a[\text{RNAP}] \tag{19}$$

Then $K_a[\text{RNAP}]$ is the ratio of bound TSSs to unbound TSSs. In a large population of cells, where empirical counts give very good estimates of probabilities, we can call this the *odds ratio* for the binding of RNAP to the DNA – the probability that the DNA is bound divided by the probability that it is unbound. In this context we will use the probabilities and their empirical esimates interchangeably.

The equilibrium binding odds $K_a[\text{RNAP}]$ depends on the sequence of the TSS DNA and the temperature: $K_a = e^{-\Delta G^\circ/RT}$, where $\Delta G^\circ$ is the Gibbs free energy of binding of RNAP to the

specific sequence of the binding site, $R$ is the ideal gas constant, and $T$ is the temperature. Here, we will assume constant temperature, so that under the prevailing conditions $K_a$ depends only on the DNA sequence of the RNAP binding site.

Since we will be working with association constants for a lot of different pairs of molecules, we'll drop the subscript $a$ and instead use a subscript that identifies the pair of molecules. For the association constant between between the RNAP and the TSS, we'll use $K_r$.

If no TFs are bound on the promoter, then the probability of the TSS being bound by RNAP is:

$$P_r = \frac{K_r[\text{RNAP}]}{1 + K_r[\text{RNAP}]} \tag{20}$$

To think about conditions of constant concentrations, it is often convenient to absorb the concentrations into the equilibrium constants. For example, if we define $q_r = K_r[\text{RNAP}]$ the probability of the TSS being bound becomes

$$P_r = \frac{q_r}{1 + q_r} \tag{21}$$

## 7.2 A general single-site model

In this section we derive the general form of (4) from the thermodynamic model and show that (4) is a special case.

To model the effects of a TF binding at a single site on the probability of RNAP binding, we consider all the possible states of a promoter consisting of the TSS and the TF binding site (TFBS). There are four states: neither site bound, only TFBS bound, only TSS bound, and both sites bound. The probability of TSS being bound is the relative frequency of promoters with bound TSS (regardless of whether TFBS is bound) divided by the frequency of all promoters (regardless of state).

$$\Pr(\text{TSS-bound}) = \frac{[\text{RNAP·DNA}] + [\text{RNAP·DNA·TF}]}{[\text{RNAP·DNA}] + [\text{RNAP·DNA·TF}] + [\text{DNA}] + [\text{DNA·TF}]} \tag{22}$$

We can use frequency and concentration interchangeably here because everything is in a fixed volume. If we now use R, D, and T as abbreviations for RNAP, DNA, and TF, and we divide both the numerator and denominator by [D], we have

$$\begin{aligned}\Pr(\text{TSS-bound}) &= \frac{[\text{R·D}]/[\text{D}] + [\text{R·D·T}]/[\text{D}]}{[\text{R·D}]/[\text{D}] + [\text{R·D·T}]/[\text{D}] + 1 + [\text{D·T}]/[\text{D}]} \\ &= \frac{[\text{R·D}]/[\text{D}] + ([\text{R·D·T}]/[\text{D·T}])([\text{D·T}]/[\text{D}])}{[\text{R·D}]/[\text{D}] + ([\text{R·D·T}]/[\text{D·T}])([\text{D·T}]/[\text{D}]) + 1 + [\text{D·T}]/[\text{D}]}\end{aligned}$$

Keep in mind that concentrations refer to a single state of the promoter, so for example [R·D]/[D] refers to the concentration of promoter DNA bound only by polymerase, divided by the centration of promoter DNA bound by nothing.

Now if we assume that there is an equilibrium between binding and unbinding of R to D, we can rewrite [R·D]/[D] as [R]$K_r$, where $K_r$ is the association constant between RNAP and the naked DNA (see Equation 19). Similarly, if there is an equilibrium between binding and unbinding of T and D then [D·T]/[D]=[T]$K_t$. Finally, if there is an equilibrium between binding and unbinding of T·D and R then

$$[\text{R·D·T}]/[\text{D·T}] = [R]K_{tr} \tag{23}$$

Making all these substitutions, we have

$$\Pr(\text{TSS-bound}) = \frac{[\text{R}]K_r + [\text{R}]K_{tr}[\text{T}]K_t}{[\text{R}]K_r + [\text{R}]K_{tr}[\text{T}]K_t + 1 + [\text{T}]K_t} \tag{24}$$

Instead of using D·T as the intermediate state between D and D·T·R, we could use D·R. In that case we would use

$$[\text{R·D·T}]/[\text{D·R}] = [T]K_{rt}$$

instead of (23), where $K_{rt}$ is the appropriate association constant (not necessarily equal to $K_{tr}$). That would lead to the equation

$$\Pr(\text{TSS-bound}) = \frac{[\text{R}]K_r + [\text{T}]K_{rt}[\text{R}]K_r}{[\text{R}]K_r + [\text{T}]K_{rt}[\text{R}]K_r + 1 + [\text{T}]K_t} \tag{25}$$

Since the left-hand-sides (24) and (25) are the same, the right-hand sides must also be equal. This implies that $K_{tr}K_t = K_{rt}K_r$. We can make (24) and (25) look the same by defining

$$\omega = \frac{K_{rt}}{K_t} = \frac{K_{tr}}{K_r}$$

Substituting for either $K_{tr}$ in (24) or $K_{rt}$ in (25) yields

$$\Pr(\text{TSS-bound}) = \frac{[\text{R}]K_r + \omega[\text{T}]K_t[\text{R}]K_r}{[\text{R}]K_r + \omega[\text{T}]K_t[\text{R}]K_r + 1 + [\text{T}]K_t} \tag{26}$$

which is symmetrical with respect to R and T. In this formulation, [R]$K_r$ is equilibrium binding odds of the TSS, in isolation from TFBS. Similarly, [T]$K_t$ is the equilibrium binding odds of the TF binding site, in isolation from TSS. When bound, RNAP and TSS can either attract or repel one another. $\omega$ represents the effect of this interaction – the degree to which the probability of both sites being bound differs from the product of probabilities of the independent binding events.

For manipulating this equation, it is useful to define $q_r = [\text{R}]K_r$ and $q_t = [\text{T}]K_t$. Substituting in to (26) yields

$$\Pr(\text{TSS-bound}) = \frac{q_r + \omega q_r q_t}{q_r + \omega q_r q_t + 1 + q_t} \tag{27}$$

To focus on transcription rate as a function of TF concentration, we assume the concentration of RNAP is constant (it probably isn't, but we can worry about that later if need be). This means that $q_r$ is constant, but $q_t$ depends on [T]. Although [T] is the concentration of TF

that is not bound to the promoter DNA, this is very close to the total amount of TF under physiological conditions. The reason is that there is only one or two copies of this particular promoter in each cell and the affinity of the TF for the DNA is low enough that those copies would hardly ever be bound unless there were many molecules of the TF present. (In reality, a lot of the TF molecules would likely be bound to other parts of the genomic DNA, but we'll ignore that for now.)

Despite all the caveats, we can use this as a model of transcription rate as a function of the concentration of one TF.

Each of the terms that is being added together in the numerator and in the denominator represents the concentration of a particular state of the promoter DNA relative to the concentration of the unbound promoter DNA. These terms are called the Boltzmann weights associated with the corresponding states. The Boltzmann weights for each of the four states of this simple promoter are:

| State | | Boltzmann weight |
|---|---|---|
| TF bound | RNAP bound | |
| No | No | 1 |
| Yes | No | $q_t$ |
| No | Yes | $q_r$ |
| Yes | Yes | $q_t q_r \omega$ |

We can calculate the probability of any given set of states by summing the Boltzmann weights of those states and dividing by the sum of the weights of all possible states, as in (27). The sum of the weights of all possible states is sometimes called the "partition function".

Our next task is to show that Equation (4) is a special case of (27). To do this, we make $\alpha$ (the conversion from the probability of bound TSS to the rate of transcription) explicit and set $\omega = 0$, which is equivalent to saying that the TF and the RNAP cannot both be bound at the same time. This results in a transcription rate of:

$$\alpha \frac{q_r}{q_r + 1 + q_t} \tag{28}$$

Substituting $q_t = [T]K_t$ leaves

$$\alpha \frac{q_r}{q_r + 1 + [T]K_t} \tag{29}$$

where [T] is the concentration of the TF, which is notated $p_j$ in (4). If (27) is the same as (4) then the two functions must yield the same transcription rates when $[T] = 0$, therefore

$$V_i = \alpha \frac{q_r}{q_r + 1} \tag{30}$$

**Exercise 7.1.** After substituting the Equation (30) into (4), compute the value $\theta_{i,j}$ needed to make (4) and (29) the same. $\theta_{i,j}$ will be a function of $K_t$ and $q_r$. The fact that there is such a value of theta means that, in the case of a single site for a perfect repressor (one which cannot occupy its site at the same times as the RNAP occupies its site), the two models are the same.

**Exercise 7.2.** Going back to the general 1-site model in Equation (26), what is the probability of RNAP being bound when the concentration of the TF goes to infinity (i.e. the limit as $[T] \to \infty$)? What about when $[T] = 0$. Comparing these, what parameter values determine whether $[T]$ is a repressor or an activator?

**Exercise 7.3.** Using the general one-site model (27) and holding $q_r$ and $K_t$ fixed at 1, plot $\Pr(\text{TSS-bound})$ versus T for a variety of values of $\omega$ on a single graph. Turn in your graph.

**Exercise 7.4.** We used a ratio of sums of the Boltzmann weights in the table to calculate the probability that the TSS will be bound by the RNAP at any given moment. That is, the sum of the probabilities of all the states in which RNAP binds the TSS. The result is shown in Equation (26).

Using the same approach with the same Boltzmann weights, write down an equation for the probability that the TF will be bound to its site. Don't forget that there is more than one state in which the TF is bound to its site.

## 7.3 A general multi-site promoter model

Thermodynamic models are based on an analysis of the possible states of a promoter. A state is a complete specification of the occupancy of the transcription start site and of every TF binding site.

The following treatment is based on (and in some places paraphrased from) the online supplement to "Buchler, N.E., Gerland, U., and Hwa, T. 2003. On schemes of combinatorial transcription logic. Proceedings of the National Academy of Sciences of the United States of America 100(9): 5136-5141." A promoter has L binding sites for particular TFs plus one binding site for RNAP. There is assumed to be just one TF that can bind each site. We introduce a binary variable $\sigma_i \in \{0, 1\}$ to denote the occupancy of each site $i$. A state is specified by a vector $(\sigma_1 \ldots \sigma_L)$ giving the occupancy of each site in the promoter. The Boltzmann weight of a state has a factor for the independent binding odds $q_i$ of each bound site. The product of these factors can be written as $\prod_{i=1}^{L} q_i^{\sigma_i}$. The exponents provide a convenient notation for including only the factors associated with bound sites. If a site $j$ is not bound then $\sigma_j = 0$, so $q_j^{\sigma_j} = 1$. If site $j$ is bound then $\sigma_j = 1$ so $q_j^{\sigma_j} = q_j$. In addition to these independent binding factors, the Boltzmann weight has a factor $\omega_{i,j}$ for the interaction of the TF bound at each site with the TFs bound at each of the other sites. The product of these factors can be written $\prod_{j<i} \omega_{i,j}^{\sigma_i \sigma_j}$, where the product is over all values of $i$ and $j$ such that $j < i$. The inequality ensures that each pair of sites is only included once. The exponent, $\sigma_i \sigma_j$, is one if both sites $i$ and $j$ are bound; otherwise it is zero. Thus, the total Boltzmann weight for state $(\sigma_1 \ldots \sigma_L)$ is

$$W(\sigma_1 \ldots \sigma_L) = \prod_{i=1}^{L} q_i^{\sigma_i} \prod_{j<i} \omega_{i,j}^{\sigma_i \sigma_j} \tag{31}$$

when RNAP is not bound.

The sum of the weights of all states in which the RNAP is not bound is then

$$Z_{\mathsf{OFF}} = \sum_{\sigma_1 \in \{0,1\}} \cdots \sum_{\sigma_L \in \{0,1\}} W(\sigma_1 \ldots \sigma_L) \tag{32}$$

To get the weights for the states in which RNAP is bound, we multiply the weights of the corresponding unbound states by $q_r$, the independent binding odds of the TSS. We also multiply by $\prod_{i=1}^{L} \omega_{i,r}^{\sigma_i}$, which represents the interaction of the TF bound to each site with the RNAP bound to the TSS. The total Boltzmann weight for all the states with TSS bound is therefore:

$$Z_{\mathsf{ON}} = \sum_{\sigma_1 \in \{0,1\}} \cdots \sum_{\sigma_L \in \{0,1\}} W(\sigma_1 \ldots \sigma_N) q_r \prod_{i=1}^{L} \omega_{i,r}^{\sigma_i}$$

The probability of RNAP being bound is then

$$P = \frac{Z_{\mathsf{ON}}}{Z_{\mathsf{ON}} + Z_{\mathsf{OFF}}} \tag{33}$$

If we want to make the TF concentrations explicit then we can replace $q_i$ in Equation (31) by $p_i K_i$ where $p_i$ is the concentration of the TF protein that binds site $i$ and $K_i$ is the association constant of $p_i$ with the sequence of site $i$.

# 8 Parameter estimation from a library of random promoters

So far we have seen how to approach a parameter estimation problem for a gene regulatory network whose structure is known. We have made specific assumptions about the regulation of genes and about the post-translational and post-transcriptional processes. In other cases we might want to know how a specific transcription factor, or set of transcription factors, regulates a target gene, or if and how specific transcription factors interact. We call this combinatorial regulation. In other words we want to know what effect a change in the promoter would have in the way a specific transcription factor regulate its target gene. Or how the synergistic effect of 2 or more factors would change when we move their sites around the promoters. Learning these "rules" of promoter architecture would shed light on transcriptional regulation in general.

This problem might need to be approached in a different way. We can start by looking at the genome. *Saccaromices cerevisiae* has about 6000 genes; of these, about 200 are thought to encode sequence-specific transcription factors. If we want to ask how each of these factors work when coupled with another factor, we would need 40,000 possible promoters, one for each pair of TFs. That's way more than the number of native yeast promoters. In Gertz et al. this problem is tackled with the use of synthetic promoters. They use hundreds of novel promoters containing binding sites for a limited number of factors, sampling a large portion of all possible regulatory element combinations and arrangements. All these promoters drive the expression of the same gene, so that transcription and translation rates, as well as mRNA degradation and protein degradation rate constants, are the same for all promoters tested. Thus, the only effect of these promoters is on the regulation of mRNA production.

## 8.1 Inferring parameters with activation and repression

The general model above includes cooperative and competitive interactions among TFs at the promoter, but to extend our parameter inference formula (12) to both activation and repression, we return to the assumption that TFs act independently. To model the effect of multiple independently acting (i.e. non-cooperative, non-competitive) TFs, we have to multiply regulation terms like (27) for each TF. To convert $P()$, the probability that the polymerase will be bound at the promoter of gene $i$, to $S_i()$, the transcription rate, we must also multiply by the global conversion factor $\alpha$:

$$S_i(p_1 \dots p_n) = \alpha \prod_{j=1}^{n} \frac{q_i(1 + p_j K_{i,j} \omega_j)}{q_i(1 + p_j K_{i,j} \omega_j) + 1 + p_j K_{i,j}} \tag{34}$$

where the subscripts now reflect the fact that we are talking about multiple TFs, each with at most one binding site at promoter $i$. This model has only one more parameter for each TF than (10), for a total of $k$ additional parameters, where $k$ is the number of TFs. $K_{i,j}$ is the association constant for the binding TF $j$ to its site in the promoter of gene $i$, $p_j$ is the concentration of TF $j$, and $\omega_j$ is the affinity of TF $j$ for RNAP. To increase the number of measurements by the number of new parameters, we need one more set of measurements for each mRNA, which could be obtained by one additional knockout combination or by expressing one TF at a third level, in addition to the wild-type level and zero in the strain in which TF $j$ is deleted.

In addition, we must estimate one $q_r$ for each promoter, representing the strength of the "basal promoter" (in the absence of any TF) and there is only one $\omega$ for each TF, representing the strength of attraction or repulsion it exerts on the RNAP. If $\omega_j < 1$ then $p_j$ is a repressor at all promoters; if $\omega_j > 1$ it is an activator at all promoters.

To apply (34) in an inference system, we can follow the same procedure as we did for (10). The only difference is that $V_i$ could easily be combined with $D_i$, the mRNA degradation rate, whereas $q_i$ cannot (or at least there's no way that's obvious to me). Thus, instead of using $b_i = \frac{D_i}{V_i}$, we will use $b_i = \frac{D_i}{\alpha}$. That costs an additional $n$ parameters, for a total of $k + n$. However, we can eliminate the $D_i$ by measuring the mRNA degradation rates directly (that wouldn't have helped before, since we would still have had to infer $V_i$ from steady state data). It is still necessary to infer $\alpha$.

It is possible to eliminate $k$ parameters by using the following simpler variant of the model:

$$S_i(p_1 \dots p_n) = \alpha \prod_{j=1}^{n} \frac{q_i(1 + p_j r_{i,j})}{q_i(1 + p_j r_{i,j}) + 1 + p_j} \tag{35}$$

However, this version allows each TF to be an activator at some promoters and a repressor at others. In fact, it seems likely that the parameters returned by the inference algorithm will have many TFs serving as both repressors and activators. If that situation is considered to be unlikely in the biological system being studied then some useful constraint on the solutions

is lost. In other words, the solution returned by the inference algorithm may be considered biologically implausible.

Another effect of using (35) instead of (34) is that the parameters are completely determined by the minimum and maximum transcription rate. For any given min and max, there is only one possible slope, or rate of transition between the min and the max.

**Exercise 8.1.** Suppose that gene $i$ is regulated only by a single TF, TF $k$, so that $S_i$ in (35), is a function only of $p_k$. Using a pencil and paper or a Mathematica notebook, calculate the value of $S_i$ in (35) when $p_k = 0$, the limit as $p_k \to \infty$, and the derivative $\frac{dS_i(p_k)}{dp_k}$. What determines whether TF $k$ represses or activates gene $i$?

# 9 A probabilistic approach to parameter estimation

## 9.1 A probabilistic take on parameter estimation

In the previous section, we used `lsqnonlin` to estimate parameters of a repression system at steady state. `lsqnonlin`, which stands for "least squares, nonlinear", attempts to find parameter values that minimize the sum of squares of the residuals. Minimizing the sum of squared residuals, or "least squares regression", may seem like an arbitrary function of the residuals. To some extent this is true – other functions of the residuals, such the as the sum of their absolute values, can also be used. However, least squares regression can be understood and justified in terms of a reasonable probabilistic model. Suppose that, when the parameter values are correct, the model actually predicts mRNA levels correctly. The difference between the predicted mRNA level and the true one is zero. However, we don't have direct access to the true mRNA level. All we have are error-prone measurements of mRNA levels. So when the residuals are calculated using these measurements, they differ from zero due to measurement error. If the measurement error arises from a variety of independent error-producing processes then they should be approximately normally distributed, with mean zero. In other words, if $\varepsilon_i$ is the $i^{th}$ residual, the probability density function of $\varepsilon_i$ is given by

$$\varepsilon_i \sim \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\varepsilon_i^2/2\sigma^2\right) \tag{36}$$

where $\sigma^2$ is called the *variance*. When the measurements are very noisy then $\sigma^2$ is large; when they are not very noisy, $\sigma^2$ is small.

One more leap of faith is required – assuming that the measurement errors of different mRNAs are independent of one another. That is, we don't have noisy data days and good data days. This is almost certainly false, but let's see where it leads anyway. The joint p.d.f. for the

errors is then

$$f(\varepsilon_1, \ldots, \varepsilon_n) \sim \prod_{i=1}^{n} \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\varepsilon_i^2/2\sigma^2\right) \tag{37}$$

$$= \frac{1}{(2\pi\sigma^2)^{n/2}} \exp\left(-\frac{1}{2\sigma^2} \sum_{i=1}^{n} \varepsilon_i^2\right) \tag{38}$$

Now we can see the link to least squares regression. Under this p.d.f, the parameter set that minimizes the sum of squared residuals is the one that maximizes the probability of the residuals. The residuals are determined by the mRNA measurements and the model parameters. For example, using the model in (12),

$$\varepsilon_i(\mathbf{m}, \mathbf{c}_i, b_i) = m_i - \frac{1}{b_i \prod_{j=1}^{n}(1 + c_{i,j} m_j)} \tag{39}$$

where the bold faced $\mathbf{m}$ indicates all the $m_i$ for all valid indices, $i = 1 \ldots n$. Similarly, $\mathbf{c}_i$ indicates all $c_{i,j}, j = 1 \ldots n$ (holding $i$ fixed). Thus, the p.d.f. on the residuals can be viewed as a function of the measurments and the parameters:

$$\begin{aligned} g(\mathbf{m}, \mathbf{c}, \mathbf{b}) &= f(\varepsilon_1(\mathbf{m}, \mathbf{c}_1, b_1), \ldots, \varepsilon_n(\mathbf{m}, \mathbf{c}_n, b_n)) \\ &= \frac{1}{(2\pi\sigma^2)^{n/2}} \exp\left(-\frac{1}{2\sigma^2} \sum_{i=1}^{n} \varepsilon_i(\mathbf{m}, \mathbf{c}_i, b_i)^2\right) \end{aligned}$$

where $\mathbf{c}$ indicates all $\mathbf{c}_i, i = 1 \ldots n$ and $\mathbf{b}$ indicates all $b_i, i = 1 \ldots n$. Now $g$ is not a p.d.f. on $\mathbf{m}$, $\mathbf{c}$, and $\mathbf{b}$ because it's integral over those variables is not necessarily one. However, if we divide $g$ by it's integral we get a new function $g'$.

$$g'(\mathbf{m}, \mathbf{c}, \mathbf{b}) = \frac{g(\mathbf{m}, \mathbf{c}, \mathbf{b})}{\int_{\mathbf{m},\mathbf{c},\mathbf{b}} g(\mathbf{m}, \mathbf{c}, \mathbf{b})} \tag{40}$$

The integral of $g'$ is one, so it can be p.d.f. The integral in the denominator is just a number, but it is one that is very difficult to calculate, since it involves integrating over a large number of variables. So about the best we can say is that $g'$ is proportional to $g$:

$$\begin{aligned} g'(\mathbf{m}, \mathbf{c}, \mathbf{b}) &\propto \frac{q}{(2\pi\sigma^2)^{n/2}} \exp\left(-\frac{1}{2\sigma^2} \sum_{i=1}^{n} \varepsilon_i(\mathbf{m}, \mathbf{c}_i, b_i)^2\right) \\ &\propto \exp\left(-\frac{1}{2\sigma^2} \sum_{i=1}^{n} \varepsilon_i(\mathbf{m}, \mathbf{c}_i, b_i)^2\right) \end{aligned}$$

As long as we have an unknown proportionality constant, we might as well incorporate all the $2\pi\sigma$ stuff into it! The overall proportionality constant will always be the integral of the unnormalized function, so you don't have to keep track of any constants:

$$g'(\mathbf{m}, \mathbf{c}, \mathbf{b}) = \frac{\exp\left(-\frac{1}{2\sigma^2} \sum_{i=1}^{n} \varepsilon_i(\mathbf{m}, \mathbf{c}_i, b_i)^2\right)}{\int_{\mathbf{m},\mathbf{c},\mathbf{b}} \exp\left(-\frac{1}{2\sigma^2} \sum_{i=1}^{n} \varepsilon_i(\mathbf{m}, \mathbf{c}_i, b_i)^2\right)}$$

If we want to take some of the variables as given and compute a conditional probability distribution over the others, we just change the integral in the denominator to integrate out only the remaining unknowns. Recall that the definition of conditional probability is just a renormalization process.

$$g'(\mathbf{c}|\mathbf{m},\mathbf{b}) = \frac{\exp\left(-\frac{1}{2\sigma^2}\sum_{i=1}^{n}\varepsilon_i(\mathbf{m},\mathbf{c}_i,b_i)^2\right)}{\int_{\mathbf{c}}\exp\left(-\frac{1}{2\sigma^2}\sum_{i=1}^{n}\varepsilon_i(\mathbf{m},\mathbf{c}_i,b_i)^2\right)}$$
$$g'(\mathbf{m}|\mathbf{c},\mathbf{b}) = \frac{\exp\left(-\frac{1}{2\sigma^2}\sum_{i=1}^{n}\varepsilon_i(\mathbf{m},\mathbf{c}_i,b_i)^2\right)}{\int_{\mathbf{m}}\exp\left(-\frac{1}{2\sigma^2}\sum_{i=1}^{n}\varepsilon_i(\mathbf{m},\mathbf{c}_i,b_i)^2\right)}$$

We don't know the proportionality constants (i.e. the denominators) for any of these, but we do know they're just fixed numbers and the important thing is the sum of squared errors. Specifically, the conditional probability of the measurements given the parameters (the likelihood function) is proportional to the sum of squared errors. Thus, if we choose the parameters that minimize the sum of squared errors, we are carrying out maximum likelihood estimation under this model.

Given this model, we can measure each mRNA level several times, on several different occasions to estimate $\sigma$. If there is reason to believe that measurements of different genes have different noise levels, we can use a gene-specific variance $\sigma_i^2$, which can also be estimated from repeated measures. In that case, maximum likelihood estimation would involve minimizing a sum of squared errors each weighted by the inverse of its variance.

Maximum likelihood estimation is OK, but there are additional things one can do using the joint conditional p.d.f. of the parameters given the measurements, $f(\mathbf{c},\mathbf{b}|\mathbf{m})$. In principle, we could compute the expected value of each parameter, integrating over all the others, as well as confidence intervals for each parameter. Alas, the many-variable integrals are intractable. However, there is an approximation method known as Gibbs sampling.

## 9.2 Gibbs sampling

Gibbs sampling is a method for sampling from a joint probability distribution when you don't know the normalization constant. The idea is that you iterate through the variables one at a time, holding all the others fixed at their current values (which may be randomly chosen values at the beginning). You normalize the conditional distribution for that one variable with all the others fixed. That normalization process involves integrating over only one variable, which is computationally feasible. For example,

$$g'(c_{i,j}|\mathbf{c}\backslash c_{i,j},\mathbf{m},\mathbf{b}) = \frac{\exp\left(-\frac{1}{2\sigma^2}\sum_{i=1}^{n}\varepsilon_i(\mathbf{m},\mathbf{c}_i,b_i)^2\right)}{\int_{c_{i,j}}\exp\left(-\frac{1}{2\sigma^2}\sum_{i=1}^{n}\varepsilon_i(\mathbf{m},\mathbf{c}_i,b_i)^2\right)} \tag{41}$$

where $\mathbf{c}\backslash c_{i,j}$ means all the $c$'s except $c_{i,j}$. You then pick a new value for the current variable ($c_{i,j}$ in the example above) by sampling from the normalized conditional distribution. After picking a new value, you move on to updating the next variable. Once all the variables have been updated, the process is repeated. Each repetition produces a set of values that can be

viewed as a sample from the joint probability distribution. The randomly chosen initial values may turn out to be very unlikely under the true joint distribution. However, if this process is repeated long enough it is guaranteed to converge on a sampling of the joint probability distribution, as long as certain fairly general conditions are met. The sampling process can be viewed as a Markov chain which, if sampled long enough, converges on a limiting distribution of state frequencies.

Under the model (39) only $\varepsilon_i$ depends on the parameter $c_{i,j}$. (In other models there might be parameters that affect all the epsilons.) Thus, (41) can be simplified to

$$g'(c_{i,j}|\mathbf{c}\backslash c_{i,j}, \mathbf{m}, \mathbf{b}) = \frac{\exp\left(-\frac{1}{2\sigma^2}\varepsilon_i(\mathbf{m}, \mathbf{c}_i, b_i)^2\right)}{\int_{c_{i,j}} \exp\left(-\frac{1}{2\sigma^2}\varepsilon_i(\mathbf{m}, \mathbf{c}_i, b_i)^2\right)} \tag{42}$$

by factoring out the terms involving the other epsilons from both the numerator and the denominator. This shows that $c_{i,j}$ is independent (in the probability sense) of all $c_{k,j}$ for $k \neq i$. Thus, we can write the left hand side of (42) as $g'(c_{i,j}|c_{i,1}, \ldots, c_{i,j-1}, c_{i,j+1}, \ldots c_{i,n}, \mathbf{m}, b_i)$. In other words, we can infer the $c$ parameters for regulation of each gene independently of the $c$ parameters for all the other genes.

For this model, the Gibbs sampling algorithm would look something like the following:

```
gibbs(rnaLevels, b, p)
% p determines the resolution with which we discretize the c's.
discretizedCVals = 0:p ./ p
Let c be an nxn matrix of random numbers between 0 and 1
Repeat until some convergence criterion is met
   For each target gene i
     For each gene j, j not equal to i
       % gene j is a potential regulator of gene i
       cumNormalProb = ...
         cumnormal(i, j, rnaLevels, b(i), c(i, :), discretizedCVals)
       % See pseudocode for cumnormal below.
       c[i,j] = sample(cumNormalProb, discretizedCVals)
   Output c
End


cumnormal(i, j, rnaLevels, bi, ci, discretizedCVals)
  % cumnormal calculates the normalized cumulative distribution on c(i,j)
  p = numel(discretizedCVals)
  cumRelProb = zeros(p)
  For each k from 1 to p
    relativeProb = relprob(i, j, rnaLevels, bi, ci, discretizedCVals(k))
    % Calculates the unnormalized or relative probability of each
    % value of c(i,j) found in the vector discretizedCVals
    If k>1 cumRelProb(k) = cumRelProb(k-1) + relativeProb
```

```
    else cumRelProb(1) = relativeProb
  cumNormalProb = cumRelProb ./ cumRelProb(p)
  Return cumNormalProb
End
```

**Exercise 9.1.** New description: As a first step in implementing the Gibbs sampler, implement the function `relprob` that is called by the pseudocode. The arguments are specified in the pseudocode. This function returns the numerator of (42), calculating $\varepsilon_i$ from (39). It uses the rna levels ($m_i$'s) in the input vector `rnaLevels`, the input value of $b_i$, and the input values for each $c$ **except**: for the specific $c_{i,j}$ indicated by the input arguments `i` and `j`, it uses the final input argument, a value from `disretizedCVals`.

**Exercise 9.2.** Implement the function `cumnormal` from the pseudocode. Feel free to substitute vector or matrix operations for loops where appropriate.

`cumnormal` calculates the non-normalized conditional probabilities of $c_{i,j}$ given the values of the other variables. As before, we assume that $0 \leq c_{i,j} \leq 1$, $c_{i,i} = 0$, and $b_i$ has already been determined. To calculate the non-normalized conditional probabilities, it discretizes the interval $[0, 1]$ into $p$ equally sized subintervals and then calculates $e^{-\varepsilon_i(\mathbf{m},\mathbf{c}_i,b_i)^2}$ for the start of subinterval, using the `relprob` function from the previous exercise. Next, calculate the cumulative non-normalized probabilities by creating a vector whose $i^{th}$ element is the sum of the probabilities for subintervals 1 to $i$. The last element of the vector of cumulative non-normalized probabilities is the sum of all the non-normalized probabilities, which is to say the normalizing constant. Dividing each entry of the vector of cumulative non-normalized probabilities by the normalizing constant produces a vector of normalized cumulative probabilities.

**Exercise 9.3.** Implement the function `sample` from the pseudocode to output a randomly chosen value from a discrete cumulative probability function. The input matrix `discretizedCVals` contains the beginnings of the discretization intervals. For now these will be equally spaced in the interval [0, 1]. `cumNormProb(k)` contains the cumulative probability associated with the interval beginning at `discretizedCVals(k)`. Test your sampling function using a discretization into 3 intervals beginning at 0, 1/3, and 2/3. Try assigning them equal probability and make sure that, when you run *sample* they are produced with roughly equal probability. Now try assigning them very unequal probabilities and make sure the sampling frequencies are appropriate. Turn in tests and your code.

**Exercise 9.4.** Implement the reminder of the Gibbs sampling algorithm outline in the pseudocode, using the functions you wrote in earlier exercises. Rather than testing for convergence, you can use an arbitrary number of iterations such as 200 or 500. Keep track of the history of all sampled values for all the c parameters. You may want to exempt an initial set of samples, e.g. the first hundred, to allow the system to "burn in". That is, to get away from the random starting values.

**Exercise 9.5.** There are not enough constraints yet to estimate the c parameters from the wild type expression levels alone. Add a deletions matrix and a corresponding set of mRNA measurements from the various deletion experiments. This seems hard conceptually but really it's just a matter of adding more constraint equations like (39). The $m$ measurements in different experiments are really different measurements and they add additional $\varepsilon$'s that depend on the same $c$'s.

# 10 Metabolic modeling