Optimization of a Stochastically Simulated Gene Network Model via Simulated Annealing

Jonathan Tomshine and Yiannis N. Kaznessis
Department of Chemical Engineering & Materials Science, and the Digital Technology Center, University of Minnesota, Minneapolis, Minnesota

ABSTRACT By rearranging naturally occurring genetic components, gene networks can be created that display novel functions. When designing these networks, the kinetic parameters describing DNA/protein binding are of great importance, as these parameters strongly influence the behavior of the resulting gene network. This article presents an optimization method based on simulated annealing to locate combinations of kinetic parameters that produce a desired behavior in a genetic network. Since gene expression is an inherently stochastic process, the simulation component of simulated annealing optimization is conducted using an accurate multiscale simulation algorithm to calculate an ensemble of network trajectories at each iteration of the simulated annealing algorithm. Using the three-gene repressilator of Elowitz and Leibler as an example, we show that gene network optimizations can be conducted using a mechanistically realistic model integrated stochastically. The repressilator is optimized to give oscillations of an arbitrary specified period. These optimized designs may then provide a starting-point for the selection of genetic components needed to realize an in vivo system.

INTRODUCTION

Genetic networks have arisen naturally to sense and respond to environmental stimuli, as well as control circadian rhythms (1,2). By rearranging naturally occurring network components, new and novel functions can be obtained (3). Using only a handful of genes, researchers have constructed logic gates (4), switches (5,6), oscillators (7), and other signal processing motifs that are familiar from the field of electrical engineering. These networks are created by specifying the desired function of the circuit and designing a connectivity that might be reasonably expected to produce that functionality. When carrying out this rational design, it is critical that the genes involved have compatible kinetic parameters, as the parameters involved in regulation, transcription, and translation may strongly influence the behavior of the resulting gene network (8).

Previous simulation work in this field has used varying methodology. The models of gene expression and regulation used in prior simulations vary, but are often simplified, combining many distinct reaction events of the transcription and translation processes into single steps. Mechanisms for the evaluation of those models also vary widely. In many cases, a combination of ordinary differential equations and stochastic simulations are employed to explore the system dynamics and the effects of noise. Such studies include circadian rhythms (9–11) and a synthetic oscillator coupled to the bacterial cell cycle (12). Other researchers have used a statistical-mechanical approach to describe the probabilities of certain enumerated states (13), though this method does not capture system dynamics. Arkin et al. (14,15) were among the first to use a mechanistic model simulated using exclusively stochastic simulations, and our simulations follow in this tradition.

Past work in designing and optimizing these gene regulatory networks has focused primarily on a completely rational approach to design (3), or on optimization methods and bifurcation analysis utilizing deterministic mass-action kinetics (16,17). While the bifurcation theory of deterministic systems is convenient and well developed, these models suffer from an inability to accurately describe the truly stochastic nature of many of the regulatory species involved.

In a cell, some species such as operator or promoter sites may be present in single-molecule concentrations. Regulatory proteins may be present in small numbers also, often <100 per cell. Furthermore, these scarce reactants are involved in slow reactions, e.g., the dissociation of σ-factor from RNAp. Given the small numbers of these species and the low rates of some reactions, continuously-valued chemical Langevin equations (18), when used alone, are insufficient to describe the system. On the other hand, depending on system dynamics, regulatory proteins and enzymes may be present in quantities of many hundreds or thousands per cell, undergoing reactions such as dimerization at very high rates. Given these two extremes, gene expression is an inherently multiscale process, and should be treated as such (19).

This article presents an optimization method based on simulated annealing (SA) to locate combinations of kinetic parameters that produce a desired dynamic behavior in a genetic network of a specified connectivity. Simulated annealing is an optimization scheme first developed in the early 1980s by Kirkpatrick et al. (20,21) for systems with many degrees of freedom. In the years since its creation, it has become one of the most popular and widely-used optimization algorithms due to its versatility and wide applicability. Simulated annealing draws an analogy between a multidimensional

Submitted February 15, 2006, and accepted for publication August 2, 2006.
Address reprint requests to Y. N. Kaznessis, Tel.: 612-624-4197; E-mail: yiannis@cems.umn.edu.

0006-3495/06/11/3196/10 $2.00
doi: 10.1529/biophysj.106.083485

optimization problem and the minimization of energy that occurs within a metal as it cools and its atoms optimize their positions to minimize Gibbs free energy. In simulated annealing, perturbations to the model replace atomic vibration, a problem-specific quality metric takes the place of energy, and a virtual temperature is lowered to ''anneal'' the system toward the optimal value of that quality metric.

Since gene expression is an inherently stochastic process, the simulation component of SA optimization is conducted using an accurate multiscale simulation algorithm (22) to calculate an ensemble of network trajectories at each iteration of the SA algorithm. The optimization of simplified models using ordinary differential equations has been well studied (16,17). This article attempts to use a mechanistic, stochastically integrated model of a gene network as the foundation for a Metropolis Monte Carlo/simulated annealing optimization scheme. On the other hand, we recognize that locating the global optimum behavior of a gene network is of little value if the resulting set of optimum parameters does not correspond to the kinetic parameters of genetic components available to the experimentalist. Therefore, we will seek to use our optimization scheme in combination with a particular gene network model to locate many sets of parameters that correspond to many different optima. The experimentalist will then be presented with a larger menu of putative systems that yield a desired network dynamic behavior, within a certain tolerable error.

The network that will be used as an example will be the three-gene repressilator of Elowitz and Leibler (7). Using simplified models and ordinary differential equations, the bifurcation analysis of such systems have been investigated (23). Prior modeling investigations using mechanistic models and stochastic simulation have determined that this system gives rise to oscillations over certain ranges of kinetic parameters (8). These studies by Tuttle et al. (8) have also revealed which kinetic parameters of the model give the best control over the period of oscillations. This makes the repressilator an ideal candidate for testing optimization schemes that can then be applied to less well-studied systems. The goal of the optimizations will be to obtain an oscillator that oscillates at or near a specified period. The models obtained will be tested for quality by comparing the periods of their oscillations to the specified goal period. In applying simulated annealing to other gene networks, this is the only aspect of the described SA algorithm itself that would need to be altered. Indeed, the definition of fitness or quality will be different with each new network-function (switch, filter, etc.) under consideration and will depend on the use to which the network is to be put.

METHODS

Overview

The Metropolis Monte Carlo/simulated annealing algorithm (20,21) is a global optimization technique that is well-suited to complex systems with many parameters. In summary, the algorithm, as applied to a chemical kinetic system, consists of the following steps:

Step 1. Perturb the reaction kinetics of the current model, $\mathbf{k}^{i}$, to form a new $\mathbf{k}'$.
Step 2. Calculate concentration trajectories $\mathbf{x}'(t)$ based on kinetic constants $\mathbf{k}'$.
Step 3. Evaluate the fitness of the $\mathbf{x}'(t)$ trajectories.
Step 4.
(*a*) If the $\mathbf{x}'(t)$ trajectories are an improvement over the previous iteration, $\mathbf{x}^{i}(t)$, accept them and set $\mathbf{x}^{i+1}(t) = \mathbf{x}'(t)$ and $\mathbf{k}^{i+1} = \mathbf{k}'$.
(*b*) If the $\mathbf{x}'(t)$ trajectories are not an improvement over the previous iteration (subject to the Metropolis criterion), discard $\mathbf{x}'(t)$ and $\mathbf{k}'$.
Step 5. Return to Step 1, unless some stopping criterion is met.

We describe each of these steps, as they apply to the repressilator, below. In general, the algorithm is applicable to any gene network, provided that the desired behavior can be described quantitatively. This would amount to changing the model in Step 2 to describe the new network and modifying Step 3 to describe the desired function of the new network.

Gene expression as chemical reactions

The example network that is used here is a repressilator consisting of three genes (7). This work refers to the lactose (lac), arabinose (ara), and tetracycline (tet) operons, as these operons are well characterized and code for proteins that are not essential for cellular function. Only a limited subset of naturally occurring genetic components are available in constructing a gene network, since a circuit that relies on repressing or overexpressing critical proteins will be incompatible with living cells.

While the lac, ara, and tet operons function very differently, the theoretical framework that is used to express each gene in silico is quite similar. Each interaction between individual, distinct chemical species is described as a chemical reaction with a particular rate constant. In this network, the lac operator controls the expression of tet repressor, the tet operator controls the expression of the ara repressor, and the ara operator controls the expression of the lac repressor. Table 1 lists the full three-gene network used to dynamically model the repressilator.

The mechanism of the expression of a single gene, as embodied in our model, is also shown schematically in Fig. 1. With this model we have attempted to capture a reasonable amount of mechanistic detail. The DNA is modeled as having an RNAp binding site (labeled *P* in Fig. 1), one or more repressor binding sites (labeled *O2* in Fig. 1), and one or more coding regions that code for protein production (labeled *lac* in Fig. 1; in this case, for lac repressor monomer). When a repressor dimer or tetramer (*AraC* in Fig. 1) is bound to an operator site, it obstructs the RNAp from binding and prevents transcription. On the other hand, when no repressor is bound, RNAp may bind, initiate transcription, and produce protein. Additionally, most reactions are reversible—as indicated in Table 1.

Although specific kinetic and thermodynamic parameters are available for the wild-type lac (24–28), ara (29–31), and tet (32,33) systems, the reference-model that serves as a starting point for most optimizations in this work is constructed symmetrically. That is, kinetic parameters for repressor-operator binding, RNAp-promoter binding, repressor degradation, mRNA degradation, etc., are set to the same value across the three different gene systems. These initial parameters were chosen to be consistent with the order-of-magnitude of values reported for the wild-type forms of these genes, as referenced in Table 1. By using a model that is initially symmetric, we insure that the system will oscillate during the first round of optimization and provide a convenient base-line for observing the changes made during the progression of the optimization algorithm. Ultimately, as the optimization proceeds, the symmetry will break down among the parameters subject to optimization.

Determining which of these parameters are subject to optimization is critical. Only some of the rate constants in this model are experimentally

TABLE 1 The complete network of reactions used as a starting point for gene network optimizations

Reaction No.	Reaction	k	Ref.
Repressor protein dimerization/tetramerization			
1	2 araC → araC2	10^9	†
2	araC2 → 2 araC	10	†
3	2 LacI → LacI2	10^9	(26)
4	LacI2 → 2 LacI	10	(26)
5	2 LacI2 → LacI4	10^9	(26)
6	LacI4 → 2 LacI2	10	(26)
7	2 tetR → tetR2	10^9	†
8	tetR2 → 2 tetR	10	†
Repressor/operator binding‡			
9	LacI4 + lacO1 → LacI4:lacO1	10^8	(28)
10	LacI4:lacO1 → LacI4 + lacO1	10^{-2}	(28)
11	tetR2 + tetO2 → tetR2:tetO2	10^8	(32)
12	tetR2:tetO2 → tetR2 + tetO2	10^{-2}	(32)
13	araC2 + araI1/I2 → araC2:araI1/I2	10^8	(37)
14	araC2:araI1/I2 → araC2 + araI1/I2	10^{-2}	(37)
RNAp / promoter binding			
15	RNAp + lacP + lacO1 → RNAp:lacP:lacO1	2×10^8	(25)
16	RNAp:lacP:lacO1 → RNAp + lacP + lacO1	10^{-1}	(25)
17	RNAp + tetP + tetR2 → RNAp:tetP:tetR2	2×10^8	(33)
18	RNAp:tetP:tetR2 → RNAp + tetP + tetR2	10^{-1}	(33)
19	RNAp + araP + araI1/I2 → RNAp:araP:araI1/I2	2×10^8	(29)§
20	RNAp:araP:araI1/I2 → RNAp + araP + araI1/I2	10^{-1}	(29)§
Bound RNAp conformational change			
21	RNAp:lacP:lacO1 → RNAp:lacP*	10^{-2}	(25)
22	RNAp:tetP:tetR2 → RNAp:tetP*	10^{-2}	†
23	RNAp:araP:araI1/I2 → RNAp:araP*	10^{-2}	†
RNAp moving to coding DNA			
24	RNAp:lacP* → lacP + lacO1 + RNAp:DNAlac	30	(14)
25	RNAp:tetP* → tetP + tetR2 + RNAp:DNAtet	30	(14)
26	RNAp:araP* → araP + araI1/I2 + RNAp:DNAara	30	(14)
Transcription			
27	RNAp:DNAlac → RNAp + tet_mRNA	30 nt/s, 600 nt	(14)
28	RNAp:DNAtet → RNAp + ara_mRNA	30 nt/s, 600 nt	(14)
29	RNAp:DNAara → RNAp + lac_mRNA	30 nt/s, 600 nt	(14)
mRNA / ribosome binding			
30	lac_mRNA + rib → rib:lac_mRNA	10^5	¶
31	tet_mRNA + rib → rib:lac_mRNA	10^5	¶
32	ara_mRNA + rib → rib:lac_mRNA	10^5	¶
Ribosome moves off of ribosome binding site			
33	rib:lac_mRNA → rib:lac_mRNA_1 + lac_mRNA	33 aa/s	(14)
34	rib:tet_mRNA → rib:lac_mRNA_1 + lac_mRNA	33 aa/s	(14)
35	rib:ara_mRNA → rib:lac_mRNA_1 + lac_mRNA	33 aa/s	(14)
Translation			
36	rib:lac_mRNA_1 → rib + lacR + Dlac	33 aa/s, 220	(14)
37	rib:lac_mRNA_1 → rib + lacR + Dlac	33 aa/s, 220	(14)
38	rib:lac_mRNA_1 → rib + lacR + Dlac	33 aa/s, 220	(14)
Protein and mRNA degradation			
39	LacI →	5.78×10^{-4}	‖
40	LacI2 →	5.78×10^{-4}	‖
41	LacI4 →	5.78×10^{-4}	‖
42	tetR →	5.78×10^{-4}	‖
43	tetR2 →	5.78×10^{-4}	‖
44	araC →	5.78×10^{-4}	‖
45	$araC_2$ →	5.78×10^{-4}	‖
46	Dlac →	5.78×10^{-4}	‖
47	Dtet →	5.78×10^{-4}	‖
48	Dara →	5.78×10^{-4}	‖
49	lac_mRNA →	2×10^{-3}	¶
50	tet_mRNA →	2×10^{-3}	¶
51	ara_mRNA →	2×10^{-3}	¶

References are to the actual kinetic data; initial data depicted below has been reduced to order-of-magnitude estimates and made symmetric across all three gene systems. Units on k: first-order reaction, s^{-1}; second-order reaction $(M\ s)^{-1}$. Reactions that appear to be third- or higher order are treated as second-order. Reactions with two kinetic constants are γ-distributed events. In these cases, the first constant is the rate of each step (first-order) and the second constant is the total number of steps. Cell volume is taken to be 10^{-15} liters. Initial conditions are one molecule for all DNA species, 0 for all RNA and protein species.

†Values were estimated for *tet* and *ara* parameters based on literature values for the *lac* system.

‡Each of these reactions is duplicated as appropriate to give two or three operator sites per promoter region. Multiple operator sites are distinguishable.

§Values were adjusted to give ~20 proteins per mRNA.

¶The forward and reverse reaction rates were estimated from a given K_d value.

‖Constant is based on typical protein degradation half-lives.

accessible to modification, and of those, some will have little effect on the behavior of the system. From prior experiments, it is known that repressor-operator affinity has a marked effect on the period of oscillations (8). Furthermore, this parameter must be changed, as the perfect symmetry of the initial model discussed above is constructed far more easily in silico than in vivo. Since many DNA-protein reactions have forward rates near the diffusion limit of $\sim 10^8\ M^{-1}\ s^{-1}$ (34), this rate is considered to be fixed and inaccessible to optimization. Only the unbinding rate constants, reactions 9, 11, and 13 in Table 1, are modified. Since the model system has two operator sites controlling each gene (omitted in Table 1), this gives 6° of freedom in the optimization.

Generation of perturbations

Perturbation of the kinetic parameters of the most recent accepted model is accomplished by choosing with uniform probability a single parameter from the list of parameters that are subject to perturbation. A Gaussian-distributed

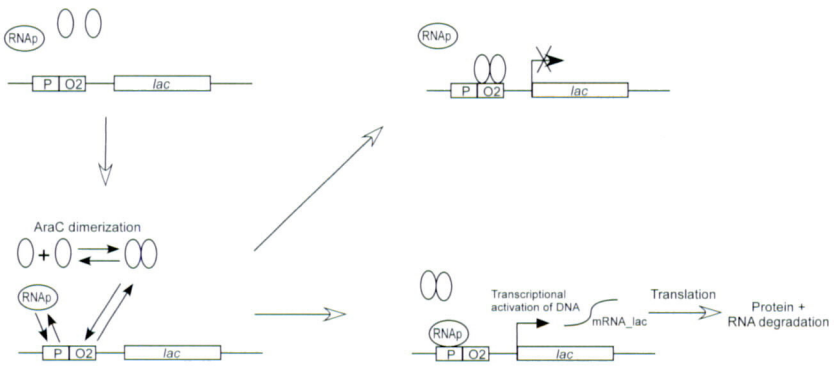

FIGURE 1 The model of the gene expression process used in this work.

random variable is then added to this parameter. If this results in a negative kinetic parameter, the move is discarded and a new random variable is chosen. This procedure may be repeated to create perturbations in more than one dimension of parameter-space. In fact, in all experiments conducted in this work, two of the six available parameters were perturbed during each optimization iteration.

Since unbiased perturbations are important (21), the mean value of these random variables is always zero. The variance is determined by multiplying the original kinetic parameter by some constant. For instance, $\boldsymbol{\sigma} = 0.2\ \boldsymbol{k_0}$, where both σ and k are vectors whose dimension corresponds to the number of kinetic parameters subject to perturbation. Prior experiments have revealed that repressor-operator affinities may be varied over roughly two orders of magnitude without quenching oscillations. Setting the perturbation standard deviation at $\pm 20\%$ of a parameter's original value gives a good coverage of the parameter space without making moves that are so aggressive as to destroy the system's oscillations in one move. Ultimately, σ is a vector whose values must be empirically determined to yield convergence with the fewest number of iterations. This effect is among those explored below.

Stochastic simulation algorithm

The calculation of the concentration trajectories is the most processor-intensive operation involved in optimizing a stochastic dynamical system. The original stochastic-simulation algorithm developed by Gillespie (18) models discrete concentration trajectories as a jump Markov process. The algorithm used in this work is a hybrid jump/continuous Markov process due to Salis and co-workers (22,40) and accelerates the original Gillespie algorithm in cases where some reactions occur at high rates and involve relatively plentiful reactants.

In a cell of volume V containing N species S_i ($i = 1 \ldots N$) engaging in M reactions R_j ($j = 1 \ldots M$), the following quantities are defined:

- $x_i(t)$—the state vector of the system contains the number of molecules of species i in the cell at time t.
- ν_{ij}—the $M \times N$ reaction matrix describes the change in the number of species S_i after the execution of each reaction R_j.
- $a_j(\boldsymbol{x})dt$—the probability that reaction R_j will occur in the cell in a differential unit of time, dt, given the current concentrations of all reactants, $\boldsymbol{x}$.

In the traditional Gillespie stochastic-simulation algorithm, the "next reaction" time would be calculated for each reaction. In Eq. 1, $URN(0,1)$ refers to a random number distributed uniformly between 0 and 1:

$$\tau_j = \frac{\ln\left(\frac{1}{URN(0,1)}\right)}{a_j} + t. \qquad (1)$$

The lowest value of τ_j would be the time until the next reaction and the value of j would give the next reaction's identity. One would then execute that reaction by adding the j^{th} row of $\boldsymbol{\nu}$ to $\boldsymbol{x}$ and adding τ to t.

In Salis' algorithm, the system is partitioned into fast and slow reactions. To be considered "fast," a reaction must meet two criteria:

1) the reaction occurs many times in a short time interval; and
2) the species involved in the reaction are present in relatively large numbers, so that the effect of each reaction event on the total number of reactant molecules is small.

If these conditions are met, the reaction is classified as "fast." Fast reactions are assumed to occur in a continuous state space rather than a discrete state space. That is, due to condition *2*, it is possible to allow concentrations that take values that would correspond to noninteger numbers of reactant molecules without introducing significant error. The M^{fast} reactions that meet these requirements may then be simulated with a Langevin equation:

$$dX_i(t) = \sum_{j=1}^{M_{fast}} \nu_{ji} a_j^f(\underline{X}(t))dt + \sum_{j=1}^{M^{fast}} \nu_{ji}\sqrt{a_j^f(\underline{X}(t))}dW_j. \qquad (2)$$

Here, $\boldsymbol{W}$ is a Wiener process (Gaussian random process, W_t–$W_s \sim N(0,t–s)$) of dimension M^{fast} and $\boldsymbol{\nu}$ is altered to contain only the fast reactions. The M–M^{fast} reactions that do not meet the requirements above are simulated by a modified version of the Gillespie algorithm. Salis does this by computing the integral of Eq. 3:

$$\int_{t_0}^{t_0+t} a_j^s(t')dt' + \log(URN_j) = R_j|^t \; j = 1 \ldots M^{slow}. \qquad (3)$$

As the fast-reaction stochastic differential equations described in Eq. 2 are integrated, the slow-reaction integrals of Eq. 3 are integrated alongside. When one of the slow-reaction residuals, R_j, crosses zero, reaction j is deemed to have occurred. The state vector, $\boldsymbol{x}$, is then updated by adding vector $\boldsymbol{\nu}_j$ and the integrations continue. Calculation of the propensities (which are based on the state vector, $\boldsymbol{x}$, and are thus coupled to both fast and slow dynamics) and evaluation of the fast/slow categorization criteria are performed continuously during the integration. This scheme may generate great time savings depending on the number of reactions classified as fast, especially since the simulation must be repeated multiple times to yield an ensemble of trajectories at each iteration of the optimization algorithm.

Evaluation of the trajectories

Once a set of reaction trajectories have been calculated, they must be evaluated for fitness. In this work, a discrete Fourier transform of protein concentration is used to compute the dominant period of the oscillator. Protein Dlac, a marker protein coexpressed with lacR monomer, is used to compute this period. Since the genes of the repressilator are expressed in sequence, the oscillator as a whole can have only one period, so this choice of a representative protein is somewhat arbitrary.

The discrete (forward) Fourier transform is given by Eq. 4 (35):

$$f_j = \sum_{k=0}^{N-1} x_k e^{-\frac{2i\pi}{N}jk} \quad j = 0, \ldots, N-1. \tag{4}$$

In the code developed for this work, the FFTW package was used to perform Fourier transforms (36).

Additionally, before computing the discrete Fourier transform, $\boldsymbol{x}$ is transformed by subtracting its mean value and then padded with zeros to increase the length and smoothness of the resulting transform. In the case of a biochemical oscillator, the vector $\boldsymbol{x}$ is of length N (plus the length of the padding zeros) and contains a single protein concentration as a function of time, spaced evenly by time Δt. The output vector, $\boldsymbol{f}$, is complex and of the same length as the input vector $\boldsymbol{x}$. After computing the Fourier transform, the power spectrum is obtained as the real vector $\boldsymbol{P}$ given in Eq. 5:

$$P_j = \frac{f_j f_j^*}{N}. \tag{5}$$

The indices of this vector, j, correspond to periods of oscillation $N \times \Delta t/j$, and the value of P_j describes the contribution of this period to the total signal. To locate the dominant period of the oscillator, λ, the index of the maximum value of $\boldsymbol{P}$ is determined and the dominant period $\lambda = N \times \Delta t/j_{max}$ is calculated.

Once the dominant period is obtained, it is used to calculate the model's energy. This may be accomplished in a variety of ways, as the choice of energy function is an element that is not prescribed by the simulated annealing algorithm, but must be defined by the practitioner or determined empirically (in fact, one can define alternate energy functions that do not rely on Fourier transforms at all). In this work, the L1 norm of the difference between the calculated dominant period and the desired period is taken to be the energy, E_i, of the model. If η-trials are computed at each iteration of the optimization algorithm, the overall energy of the model consists of an ensemble average of these η individual energies, E_i. The accept/reject decision is based on this ensemble average energy:

$$\langle E \rangle = \frac{1}{\eta}\sum_{i=1}^{\eta} E_i = \frac{1}{\eta}\sum_{i=1}^{\eta} |\lambda_i - \lambda_{goal}|. \tag{6}$$

If a parameter perturbation should happen to produce a system that does not oscillate, there will be no well-defined dominant period. In these cases, the Fourier transform will be relatively flat, and locating the maximum will give a nonsensical period value far from the desired goal. These cases result in very high energies and are rejected with extremely high probability—a self-correcting situation.

Simulated annealing

The decision of whether to accept or reject the model in question is made using the Metropolis criterion with simulated annealing (21). Once the ensemble average energy has been obtained, a random number is drawn, $U \sim$ Uniform[0,1]. The model is then accepted and recorded if Eq. 7 is satisfied:

$$U \leq e^{\frac{-(\langle E \rangle_k - \langle E \rangle_{k-1})}{T}}. \tag{7}$$

If Eq. 7 is not satisfied, the model is discarded.

While the temperature is not defined in its usual physical sense, it is used in Eq. 7 to adjust the tolerance for accepting unfavorable moves; a temperature of 0 would require each move in parameter space to decrease the ensemble average energy of the system. In that sense, it plays the same role as the thermal energy, kT, in a physical system of particles. While many annealing schedules are possible, one that is commonly used is the proportional scheme where $T_{k+1} = \alpha \times T_k$. With this schedule, T_0 and α are empirically determined with $\alpha < 1$. Ideally, whatever the initial value of T, it should go to zero as the optimization concludes. Of course, the proportional scheme will never reach zero temperature, so the number of iterations required (and thus the final temperature) is empirically determined by the quality of the solutions obtained. Once the energy fails to move significantly up or down (i.e., freezing), the algorithm may be halted.

To implement simulated annealing, the thermal energy value, T, is decreased monotonically according to an empirically determined annealing schedule. This step contains one of the most significant differences between the optimization of stochastic dynamics and deterministic dynamics. Since the quantity $\langle E \rangle_k$ is a random variable, the accept/reject decision is made without knowledge of the exact fitness of the current model. If the dynamics were simulated deterministically, E_k could be calculated exactly and the ensemble average would be unnecessary.

Multiple identical optimization experiments

Since multiple combinations of kinetic parameters will produce the same overall system period, multiple independent optimizations must be performed to collect a representative sample of optimized parameter sets. Each optimization, if started with identical kinetic parameters and using identical initial conditions, will proceed differently due to the stochastic nature of the SA algorithm and may reach different a set of kinetic parameters.

For the results of the optimization to be useful in constructing a network in vivo, it is critical that all of the optimized parameters match those of the genetic components used by the experimentalist. Since the number of real, available genetic components is finite, not every optimization will yield a system that would be easy to construct. By performing the optimization many times, we seek not just one set of kinetic parameters that yield the desired oscillation period, but many sets of kinetic parameters that may be employed to give oscillations at or within some tolerance of the desired period. For this reason, the majority of the data described below deal with sets of optimizations that start from identical locations in state and parameter space and have identical goals.

RESULTS AND DISCUSSION

Optimization algorithm behavior

In the trials that follow, the rate constants under investigation are the rates of the dissociation of repressor complex and operator site, shown in Table 1 as Reaction types 9, 11, and 13. These constants were chosen because they are known to have a significant effect on the period of the system when varied within reasonable bounds (8). Since two operator sites are available in each of the three genes present, there are six parameters that are independently subject to perturbation. All six dissociation rate constants are initially equal. At each step of the optimization algorithm, two of these six parameters are altered independently by 10–30% of their original value to quickly explore the parameter space.

The first aspect of the stochastic-model simulated annealing algorithm to be investigated is the size, η, of the ensemble of trials that is used to compute the average energy of the model. This parameter is one that is unique to the optimization of stochastic dynamical systems. When optimizing a system of ordinary differential equations, one computes the fitness of the model after only a single integration.

To investigate this parameter, a period of 6 h was first chosen as the goal of the optimizations. This value is known to be well within the envelope of achievable oscillations (8).

The initial period of the oscillator reference-model is 4.3 h. At each iteration of the simulated annealing algorithm, η trajectories were calculated for the same set of kinetic parameters and used to compute the ensemble average energy $\langle E \rangle_k$ at that iteration. The parameter η was varied, and 20 identical optimizations were conducted at each η-value. The results are summarized in Table 2 and depicted in Fig. 2.

Each of these optimizations was given $24 \cdot \eta$ CPU hours to run, after which they were terminated. The ensembles for each optimization were computed in parallel using η Intel Itanium 2 CPUs at 1.5 GHz. The energies listed in Table 2 are computed with the L1 norm, i.e., the absolute value of the difference between the goal period and the ensemble-average period. Fig. 3, *A* and *C*, shows the oscillator before and Fig. 3, *B* and *D*, after optimization using the initial kinetics and a set of optimized kinetics obtained using $\eta = 2$.

As the value of η increases, the uncertainty in $\langle E \rangle_k$ at each iteration of the SA algorithm is reduced, with the greatest reductions in errors occurring at small η-values. Table 1 also shows that acceptance ratio (the number of moves in parameter space that are accepted, divided by the total number that are attempted) monotonically increases with ensemble size. In essence, using a given amount of CPU time, one may sample many points in parameter space, moving from point to point with only a cursory examination of the resulting network quality at each point. Alternatively, one may use the same amount of CPU time to sample fewer points in parameter space while determining the fitness of each point more thoroughly. In Fig. 2 and Table 2, the number of optimizations at each value of η was held constant, so the amount of CPU time expended increases linearly with the value of η. As Fig. 2 depicts, as the size of the ensemble increases, the mean best energy of the optimizations generally decreases and the error in the estimate shrinks.

Additionally, a single optimization was performed using an ensemble size of $\eta = 100$. This would consume the same amount of CPU time as 10 optimizations using an ensemble size of $\eta = 10$. By computing a larger ensemble of trials with the same kinetics, this optimization has a more accurate estimate of $\langle E \rangle_k$ at each iteration of the SA algorithm than $\eta = 10$ would, but may only sample a relatively small number of points in parameter space, relative to the 10 optimizations of $\eta = 10$. After 24η CPU hours, the lowest energy achieved is 0.353 h. In contrast, performing 10 independent optimizations

TABLE 2 The effect of varying ensemble size on performance of the optimization

Ensemble size [trials]	Avg. best energy [hours]	SD of best energy [hours]	Avg. acceptance ratio	CPU time expended per optimization [hours]
1	0.714	0.462	0.149	24
2	0.539	0.466	0.208	48
6	0.426	0.340	0.300	144
10	0.374	0.132	0.334	240
16	0.426	0.135	0.382	384

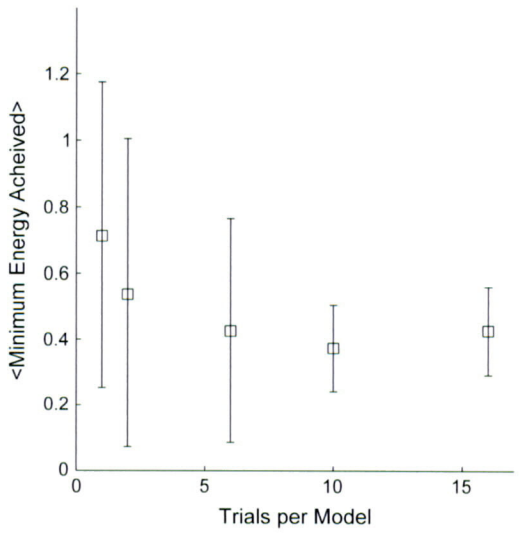

FIGURE 2 The mean value of the best energy obtained by 20 optimizations as a function of the size of the ensemble (η) used to compute mean energy within the SA algorithm. The error bars represent ± 1 standard deviation.

with $\eta = 10$ uses the same number of CPU hours, but provides a set of 10 optimized systems with minimum energies of 0.388 ± 0.144 h. The best energy obtained in these 10 optimizations is 0.258, better than the 0.353 of single large-ensemble optimization. Additionally, having 10 sets of parameters would provide a larger menu of possibilities for the experimentalist to select from. Therefore, it is more productive to devote a given amount of CPU time to running more trials, rather than evaluating each trial precisely.

Because of the large amount of CPU time required to compute ensembles of stochastic trajectories, the amount of CPU time consumed was used as the stopping criterion, rather than the number of iterations, accepted models, energy improvement, or other traditional metrics of convergence. This is a concession to practical necessity. On the other hand, it is important that each optimization run long enough to converge to a reasonable degree. To demonstrate that this is occurring within $24 \cdot \eta$ CPU hours, five of the optimizations described in Table 1 for $\eta = 6$ were continued for an additional $96 \cdot \eta$ CPU hours. These results are shown in Fig. 4 below. While improvement would (and does) continue to occur, the rate of improvement is substantially slower than during the initial $24 \cdot \eta$ CPU hours. This is reflected in the acceptance ratio, which drops from 0.300 in the initial interval to 0.077 in interval depicted in Fig. 4.

The parameters investigated next were the initial temperature and the annealing schedule. The simplest annealing method is the proportional method, whereby $T_{i+1} = \alpha \times T_i$, where α is an empirically chosen number with $0 < \alpha < 1$. Table 3 presents the results of trials with this optimization scheme. Again, the initial period was 4.3 h, the goal was 6.0 h, and an ensemble of six trials was used to evaluate the

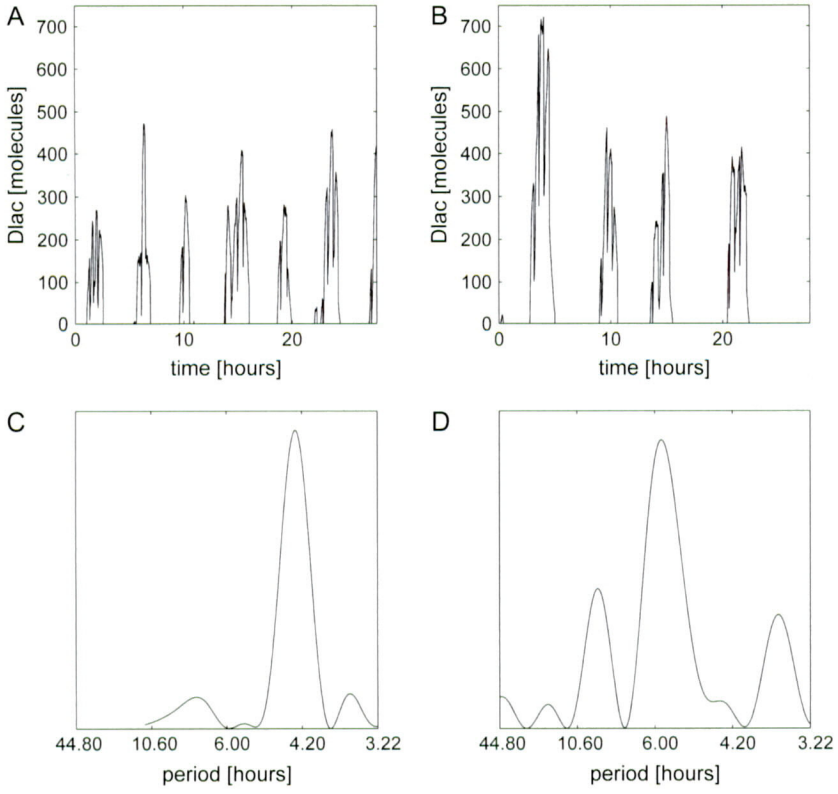

FIGURE 3 Plots of the number of molecules of Dlac (a marker protein coexpressed with lacR) in a sample cell in the time domain and their associated power spectra in the period domain. The plots at left (*A*,*C*) are the oscillator at its initial configuration while the plots at right (*B*,*D*) show it after optimization. The objective of the optimization was a period of 6 h.

average energy. Table 3 also depicts a data point at a low initial temperature and with no annealing (i.e., $\alpha = 1.000$). This series of trials generally performed more poorly than the simulated annealing trials.

The size of the attempted steps in parameter space is also critical to the optimization process—too large a step could cause the system to cease oscillations altogether (leading to rejection of the attempt with probability ~1), while steps that are too small will require an excessive number of iterations for convergence. As described above, the standard deviation of attempted step size is defined by multiplying the original value of the kinetic parameter in question with an empirical proportionality constant: $\sigma_i = c \times k_{0,j}$. This step size does not change as the optimization progresses. The proportionality constant, c, is investigated in Table 4; all other parameters, including an ensemble size of 6, were held constant. These data show that results improve as steps are larger and more aggressive, even up to standard-deviations of 3/10 of a parameter's original value.

The final series of trials investigates the effect of varying the initial rate constants, k_0. This also implies that the size of the steps in parameter space changes, since their standard

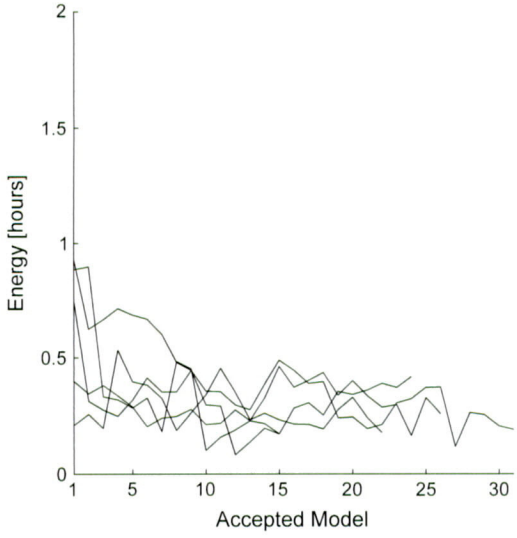

FIGURE 4 Optimization continued for 576 (96 · η) CPU h after the initial 144 (24 · η) CPU h. Initial period is ~1.6 h.

TABLE 3 Effects of initial temperature and annealing proportionality constant

T_0	α	Avg. best energy [hours]	SD of best energy [hours]	Initial acceptance ratio	Avg. acceptance ratio	No. of optimizations that reached $E \leq 0.25$ (of 20 total)
6×10^{-2}	0.900	0.47	0.31	0.85	0.27	3
6×10^{-2}	0.975	0.43	0.34	0.76	0.30	5
6×10^{-2}	0.990	0.37	0.16	0.81	0.32	3
9×10^{-2}	0.975	0.36	0.26	0.85	0.33	8
1×10^{-2}	1.000	0.63	0.48	0.67	0.18	4

TABLE 4 Variation of step size and its effect on convergence

c	Avg. best energy [hours]	SD of best energy [hours]	Initial acceptance ratio	Avg. acceptance ratio	No. of optimizations that reached $E \leq 0.25$ (of 20 total)
0.1	1.31	0.39	0.79	0.30	0
0.2	0.43	0.34	0.76	0.30	5
0.3	0.30	0.11	0.73	0.24	8

deviations were defined as $\sigma_i = c \cdot k_{0,i}$. The two sections of Table 5 show this effect. The trials in the upper section (Table 5, lines 1–3) allow the step-size to change with initial condition, while the trials in the lower section (Table 5, lines 4 and 5) use steps with the same standard deviation as line 2, $\sigma = 0.2 \cdot 10^{-2}$ 1/s.

The initial condition is furthest from the goal in lines 1 and 4, and indeed, neither choice of step-size reaches the goal in 144 (24 · 6) CPU hours. Lines 3 and 5, however, show how critical step-size may be. In line 3, none of the optimizations reach the goal period within ±0.25. Line 5 uses identical initial conditions, but all 20 optimizations reach the goal period given an identical amount of CPU time. In line 3, the steps in parameter space are $0.2 \cdot 10^{-3}$ 1/s = 0.0002 1/s. This is simply too small to sample the parameter space efficiently. Line 5, however, uses steps that are an order-of-magnitude larger, 0.002 1/s. By sampling the region around the solution more efficiently, the results are drastically improved. Curiously, the optimizations summarized on line 5 of Table 5 actually outperform those summarized on line 2, though the initial condition on line 5 is further from the goal and both sets use steps of the same size.

The kinetic constants obtained via optimization for the case of $\eta = 2$ are presented in Table 6 below. As discussed above, the parameters that are subject to perturbation are the repressor complex/operator site unbinding rate constants. Since each of the three operons is modeled as having two operator sites, this gives 6° of freedom.

Because simulated annealing is a stochastic optimization scheme, the kinetic constants obtained by this optimization are themselves random variables. It is also worth noting that there are many ways that this network may approach the optimized state. That is, there are multiple and very different combinations of kinetic parameters that yield a given period of oscillation, and in principle, multiple energy zeros.

TABLE 5 The effect of variation of initial conditions

	k_0 [1/s]	Avg. initial period [hours]	Avg. final energy [hours]	SD of final energy [hours]	Initial acceptance ratio	Avg. acceptance ratio	No. of optimizations that reached $E \leq 0.25$ (of 20 total)
1	10^{-1}	2.81	2.25	0.54	0.88	0.31	0
2	10^{-2}	4.33	0.43	0.34	0.76	0.30	5
3	10^{-3}	8.29	1.07	0.22	0.70	0.14	0
4	10^{-1}	2.80	2.93	0.07	0.80	0.30	0
5	10^{-3}	8.50	0.16	0.04	0.67	0.20	20

Lines 1–3 also vary step size, while lines 4 and 5 use step-sizes identical to those of line 2.

Finally, Table 7 gives the kinetic constants of a single optimization from the group of 20 optimizations represented in Table 6. The oscillator depicted in Table 7 had an energy of 0.0249; its period was ~2 min away from the desired goal of 6 h.

These are within the range of realistic values for a repressor/operator dissociation rate constant. Experiments with one of the best-studied systems, the wild-type Lac repressor, show a dissociation rate constant of $\sim 20 \times 10^{-3}$ s^{-1} (25). Assuming a repressor-operator forward binding rate near the diffusion limit of $\sim 10^8$ M^{-1} s^{-1} (34), this gives affinities in the range of 6.67×10^9 to 4.76×10^{11} M^{-1}. In the specific case of the Lac system, the forward binding rate constant is actually somewhat higher than the theoretical diffusion-limit; Riggs et al. (24) give a value of 7×10^9 M^{-1} s^{-1}. This discrepancy is thought to involve the repressor sliding along the DNA, thus reducing the dimensionality of the diffusion. The affinities of many mutant operator sites have also been studied. A selection is presented in Table 8.

Of course, affinities only give the ratio between binding and unbinding rate constants. As more true kinetic data is made available by experiments, the accuracy and usefulness of modeling will greatly benefit.

CONCLUSIONS

The simulated annealing algorithm illustrated here demonstrates that gene network optimizations can be conducted using a mechanistically realistic model integrated stochastically. While the process is computationally intensive, it

TABLE 6 Actual kinetic constants for repressor complex/operator site unbinding obtained for an oscillator using an ensemble size of $\eta = 2$ trials per set of kinetic parameters

	Mean [1/s]	Min [1/s]	Max [1/s]	Mean [s]	Min [s]	Max [min]
Lac 1	$(7.73 \pm 4.53) \times 10^{-3}$	0.15×10^{-3}	15.32×10^{-3}	344	45	76.12
Lac 2	$(8.54 \pm 3.47) \times 10^{-3}$	1.28×10^{-3}	16.06×10^{-3}	117	43	9.05
Tet 1	$(8.26 \pm 4.21) \times 10^{-3}$	0.16×10^{-3}	15.83×10^{-3}	365	44	70.66
Tet 2	$(7.73 \pm 4.40) \times 10^{-3}$	0.22×10^{-3}	13.96×10^{-3}	333	50	53.46
Ara 1	$(7.13 \pm 4.32) \times 10^{-3}$	0.61×10^{-3}	15.05×10^{-3}	243	46	19.02
Ara 2	$(7.07 \pm 5.01) \times 10^{-3}$	0.21×10^{-3}	17.14×10^{-3}	438	40	56.01

Twenty independent optimizations were conducted. Values are given in units of 1/s and as half-lives. All rate constants started at 10×10^{-3} s^{-1}.

TABLE 7 The results of a single-optimized oscillator

	k [1/s]
Lac 1	15.32×10^{-3}
Lac 2	8.66×10^{-3}
Tet 1	9.85×10^{-3}
Tet 2	6.73×10^{-3}
Ara 1	4.67×10^{-3}
Ara 2	0.21×10^{-3}

This single set of kinetics produces oscillations with a period of ~6 h.

does have the advantage of using some of the most realistic models of gene expression available—models whose parameters describe actual physical rate constants. Experiments conducted with the optimization parameters show that while the simulated annealing algorithm does require several empirical parameters, these parameters may be varied within a fairly wide range. Furthermore, most of these empirical parameters are not network-specific and could be applied to design and optimize other gene networks.

The experiments described in Table 2 and Fig. 2 show that the size of the ensemble of trials that must be calculated within the simulated annealing algorithm is not especially large. While computing just a single trial at each point in phase-space does produce somewhat erratic behavior, a significant improvement can be made by computing on the order of 10 trials, rather than hundreds or thousands. In fact, if a given amount of CPU time is available, it is shown to be more productive to compute a large number of independent optimizations using small ensembles than to compute a small number of optimizations using large ensembles.

In the end, however, the goal would be to actually construct these oscillators with a priori control over their behavior. This is somewhat more difficult due to the general lack of true kinetic data available. While the ordering of reaction-events required for gene expression is very well known, e.g., binding of the RNAp to the promoter, binding of the repressor to the operator, etc., in many cases our knowledge stops at this level. Only a subset of these binding or interaction events have been thermodynamically characterized in terms of binding affinity, and of those, an even smaller subset have been fully kinetically characterized in terms of binding and unbinding rate constants. Mutant forms are even more lacking in kinetic data than are their wild-type counterparts. It is this level of detailed knowledge, however, that is necessary to truly predict and model the expression of a single gene or a gene regulatory network.

TABLE 8 Experimental equilibrium data for actual lac and tet mutants (8)

lac operator:repressor variants (38)	*tet* operator:repressor variants (39)
O_{sym}:wt K_{eq} = 1e11 [M^{-1}]	wt:wt $r\beta^*$ = 0%
O_{4a}:wt K_{eq} = 1.1e8 [M^{-1}]	O_{2T}:KA33 $r\beta$ = 28%
O_{5a}:wt K_{eq}= 5.8e8 [M^{-1}]	O_{wt}:TA27 $r\beta$ = 45%
O_{5c}:wt K_{eq} = 8.1e7 [M^{-1}]	O_{3C}:wt $r\beta$ = 68%
O_{4a5c}:wt K_{eq} = 5.2e6 [M^{-1}]	O_{5G}:wt $r\beta$ = 87%

*The expression levels of *tet* operator:repressor variants are measured in relative β-galactosidase activity ($r\beta$), where 100% expression occurs when repressor is absent.

An experimentalist wishing to construct a specific network in vivo would select a set of wild-type or mutant genetic components whose kinetic parameters match, as closely as possible, those of an optimized model. This optimization scheme provides a means by which many potential sets of such parameters may be located. The next step toward application is the design or discovery of DNA binding components whose kinetics approach at least one of these sets.

Computational support from the Minnesota Supercomputing Institute is gratefully acknowledged.

This work was supported by National Science Foundation grant No. BES-0425882. This work was also supported by the National Computational Science Alliance under grant No. TG-MCA04N033.

REFERENCES

1. Goldbeter, A. 2002. Computational approaches to cellular rhythms. *Nature*. 420:238–245.
2. Stelling, J., U. Sauer, Z. Szallasi, F. J. Doyle 3rd, and J. Doyle. 2004. Robustness of cellular functions. *Cell*. 118:675–685.
3. Kaern, M., W. J. Blake, and J. J. Collins. 2003. The engineering of gene regulatory networks. *Annu. Rev. Biomed. Eng*. 5:179–206.
4. Guet, C. C., M. B. Elowitz, W. Hsing, and S. Leibler. 2002. Combinatorial synthesis of genetic networks. *Science*. 296:1466–1470.
5. Gardner, T. S., C. R. Cantor, and J. J. Collins. 2000. Construction of a genetic toggle switch in *Escherichia coli*. *Nature*. 403:339–342.
6. Atkinson, M. R., M. A. Savageau, J. T. Myers, and A. J. Ninfa. 2003. Development of genetic circuitry exhibiting toggle switch or oscillatory behavior in *Escherichia coli*. *Cell*. 113:597–607.
7. Elowitz, M. B., and S. Leibler. 2000. A synthetic oscillatory network of transcriptional regulators. *Nature*. 403:335–338.
8. Tuttle, L., H. Salis, J. Tomshine, and Y. N. Kaznessis. 2005. Model-driven designs of an oscillating gene network. *Biophys. J*. 89: 3873–3883.
9. Smolen, P., D. A. Baxter, and J. H. Byrne. 2002. A reduced model clarifies the role of feedback loops and time delays in the *Drosophila* circadian oscillator. *Biophys. J*. 83:2349–2359.
10. Vilar, J. M., H. Y. Kueh, N. Barkai, and S. Leibler. 2002. Mechanisms of noise-resistance in genetic oscillators. *Proc. Natl. Acad. Sci. USA*. 99:5988–5992.
11. Barkai, N., and S. Leibler. 2000. Circadian clocks limited by noise. *Nature*. 403:267–268.
12. Hasty, J., M. Dolnik, V. Rottschafer, and J. J. Collins. 2002. Synthetic gene network for entraining and amplifying cellular oscillations. *Phys. Rev. Lett*. 88:148101.
13. Shea, M. A., and G. K. Ackers. 1985. The OR control system of bacteriophage lambda. A physical-chemical model for gene regulation. *J. Mol. Biol*. 181:211–230.
14. Arkin, A., J. Ross, and H. H. McAdams. 1998. Stochastic kinetic analysis of developmental pathway bifurcation in phage λ-infected *Escherichia coli* cells. *Genetics*. 149:1633–1648.
15. McAdams, H. H., and A. Arkin. 1997. Stochastic mechanisms in gene expression. *Proc. Natl. Acad. Sci. USA*. 94:814–819.
16. Mendes, P., and D. Kell. 1998. Non-linear optimization of biochemical pathways: applications to metabolic engineering and parameter estimation. *Bioinformatics*. 14:869–883.

17. Francois, P., and V. Hakim. 2004. Design of genetic networks with specified functions by evolution in silico. *Proc. Natl. Acad. Sci. USA.* 101: 580–585.

18. Gillespie, D. T. 2000. The chemical Langevin equation. *J. Chem. Phys.* 113:297–306.

19. Elowitz, M. B., A. J. Levine, E. D. Siggia, and P. S. Swain. 2002. Stochastic gene expression in a single cell. *Science.* 297: 1183–1186.

20. Kirkpatrick, S., J. C. D. Gelatt, and M. P. Vecchi. 1983. Optimization by simulated annealing. *Science.* 220:671–680.

21. Liu, J. S. 2001. Monte Carlo Strategies in Scientific Computing. Springer-Verlag, New York, NY.

22. Salis, H., and Y. Kaznessis. 2005. Accurate hybrid stochastic simulation of a system of coupled chemical or biochemical reactions. *J. Chem. Phys.* 122:54103-1–54103-13.

23. Wang, R., Z. Jing, and L. Chen. 2005. Modelling periodic oscillation in gene regulatory networks by cyclic feedback systems. *Bull. Math. Biol.* 67:339–367.

24. Riggs, A. D., S. Bourgeois, and M. Cohn. 1970. The lac repressor-operator interaction. 3. Kinetic studies. *J. Mol. Biol.* 53:401–417.

25. Fickert, R., and B. Muller-Hill. 1992. How Lac repressor finds lac operator in vitro. *J. Mol. Biol.* 226:59–68.

26. Levandoski, M. M., O. V. Tsodikov, D. E. Frank, S. E. Melcher, R. M. Saecker, and M. T. Record, Jr. 1996. Cooperative and anticooperative effects in binding of the first and second plasmid Osym operators to a LacI tetramer: evidence for contributions of non-operator DNA binding by wrapping and looping. *J. Mol. Biol.* 260:697–717.

27. Beckwith, J. R., and D. Zipser, editors. 1970. The Lactose Operon. Cold Spring Harbor Laboratory, Cold Spring Harbor, NY.

28. McKnight, S. L., and K. R. Yamamoto, editors. 1992. Transcriptional Regulation. Cold Spring Harbor Laboratory Press, Cold Spring Harbor, NY.

29. Zhang, X., T. Reeder, and R. Schleif. 1996. Transcription activation parameters at ara pBAD. *J. Mol. Biol.* 258:14–24.

30. Hendrickson, W., and R. F. Schleif. 1984. Regulation of the *Escherichia coli* L-arabinose operon studied by gel electrophoresis DNA binding assay. *J. Mol. Biol.* 178:611–628.

31. Martin, K. J., and R. F. Schleif. 1987. Equilibrium DNA-binding of AraC protein. Compensation for displaced ions. *J. Mol. Biol.* 195:741–744.

32. Kleinschmidt, C., K. Tovar, W. Hillen, and D. Porschke. 1988. Dynamics of repressor-operator recognition: the Tn10-encoded tetracycline resistance control. *Biochemistry.* 27:1094–1104.

33. Bertrand-Burggraf, E., J. F. Lefevre, and M. Daune. 1984. A new experimental approach for studying the association between RNA polymerase and the tet promoter of pBR322. *Nucleic Acids Res.* 12:1697–1706.

34. Halford, S. E., and J. F. Marko. 2004. How do site-specific DNA-binding proteins find their targets? *Nucleic Acids Res.* 32:3040–3052.

35. Madisetti, V. K., and D. B. Williams. 1998. Signals and systems. *In* The Digital Signal Processing Handbook. V. K. Madisetti and D. B. Williams, editors. CRC Press, Boca Raton, FL.

36. Frigo, M., and S. G. Johnson. 1998. FFTW: an adaptive software architecture for the FFT. *In* Proceedings of the 1998 IEEE International Conference on Acoustics, Speech and Signal Processing, New York, NY. 1381–1384.

37. Stickle, D. F., K. M. Vossen, D. A. Riley, and M. G. Fried. 1994. Free DNA concentration in *E. coli* estimated by an analysis of competition for DNA binding proteins. *J. Theor. Biol.* 168:1–12.

38. Frank, D. E., R. M. Saecker, J. P. Bond, M. W. Capp, O. V. Tsodikov, S. E. Melcher, M. M. Levandoski, and M. T. Record, Jr. 1997. Thermodynamics of the interactions of lac repressor with variants of the symmetric lac operator: effects of converting a consensus site to a non-specific site. *J. Mol. Biol.* 267:1186–1206.

39. Wissmann, A., R. Baumeister, G. Muller, B. Hecht, V. Helbl, K. Pfleiderer, and W. Hillen. 1991. Amino acids determining operator binding specificity in the helix-turn-helix motif of Tn10 Tet repressor. *EMBO J.* 10:4145–4152.

40. Salis, H., V. Sotiropoulos, and Y. N. Kaznessis. 2006. Multiscale Hy3S: hybrid stochastic simulation for supercomputers. *BMC Bioinformatics.* 7:93.