IOP Publishing Phys. Biol. 14 (2017) 065001 https://doi.org/10.1088/1478-3975/aa868a

Physical Biology

PAPER

RECEIVED
2 December 2016

REVISED
14 August 2017

ACCEPTED FOR PUBLICATION
16 August 2017

PUBLISHED
3 November 2017

An application of the Krylov-FSP-SSA method to parameter fitting with maximum likelihood*

Khanh N Dinh and Roger B Sidje[1]

Department of Mathematics, University of Alabama, Tuscaloosa, AL 35487, United States of America

[1] Author to whom any correspondence should be addressed.

E-mail: kdinh@crimson.ua.edu **and** roger.b.sidje@ua.edu

Keywords: parameter fitting, chemical master equation, finite state Projection, gene regulation, systems biology, maximum likelihood, optimization

Abstract

Monte Carlo methods such as the stochastic simulation algorithm (SSA) have traditionally been employed in gene regulation problems. However, there has been increasing interest to directly obtain the probability distribution of the molecules involved by solving the chemical master equation (CME). This requires addressing the curse of dimensionality that is inherent in most gene regulation problems. The finite state projection (FSP) seeks to address the challenge and there have been variants that further reduce the size of the projection or that accelerate the resulting matrix exponential. The Krylov-FSP-SSA variant has proved numerically efficient by combining, on one hand, the SSA to adaptively drive the FSP, and on the other hand, adaptive Krylov techniques to evaluate the matrix exponential. Here we apply this Krylov-FSP-SSA to a mutual inhibitory gene network synthetically engineered in Saccharomyces cerevisiae, in which bimodality arises. We show numerically that the approach can efficiently approximate the transient probability distribution, and this has important implications for parameter fitting, where the CME has to be solved for many different parameter sets. The fitting scheme amounts to an optimization problem of finding the parameter set so that the transient probability distributions fit the observations with maximum likelihood. We compare five optimization schemes for this difficult problem, thereby providing further insights into this approach of parameter estimation that is often applied to models in systems biology where there is a need to calibrate free parameters.

1. Introduction

One of the important goals of systems biology is to understand the complex and stochastic dynamics of gene regulation. A challenge toward this goal is that there are usually many unknown reaction rates in the involved mathematical models.

Here we use a data-driven maximum likelihood approach [1–7] to search for and validate unknown parameters so that the distributions reported in the experimental data can be recreated in the models.

We apply this approach to synthetic data generated from the negative feedback model in Min Wu *et al* [8], where an inhibitory gene network was constructed using two synthetic promoters [9]. Their lab experiment involved TetR and LacI (figure 1), which are repressors that inhibit the expression of each other by binding to their corresponding operator sites, TetR operator (Otet) and LacI operator (Olac), placed in engineered GAL1 promoters. Anhydrotetracycline (ATc) was used to inhibit TetR. The abundance of each protein was recorded by flow cytometry with yeast-enhanced green fluorescent protein (yEGFP) and mCherry red fluorescent protein.

A mathematical model consisting of a set of two ordinary differential equations (ODEs) was proposed to explain the interaction of the two proteins involved [8, 9]. Experiments in [8] showed bimodality that is disruptive to the ODE model. This is because the dynamics of cellular processes with low copy numbers of molecules can be noisy events [10–12] and so the deterministic ODE formulation is not always ideal. This is supported by real time measurements of RNAs and proteins using fluoresent proteins made possible by recent advances in bio-imaging [13–17]. Because of this, stochastic models have arisen as a natural mode-

*Work supported by NSF grant DMS-1320849.

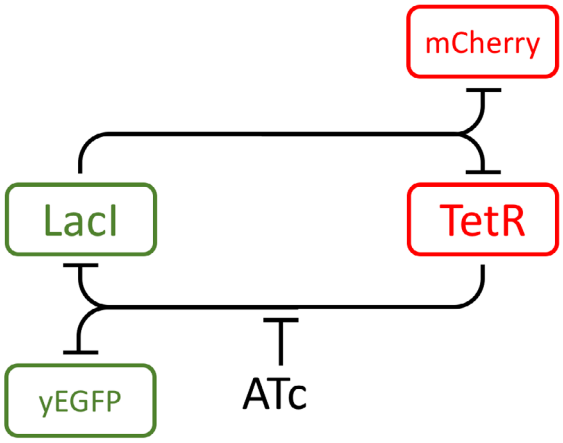

Figure 1. Schematic diagram of the network used in Min Wu *et al* [8], copyright 2013 National Academy of Sciences.

ling choice in many cases in systems biology [3, 18, 19]. The problem of integrating stochastic models with single-cell data is, therefore, of relevance to the systems biology community [20]. But in order to use either deterministic or stochastic biological models for analysis or for designing future lab experiments with confidence, unknown parameters need to be found so that the models can capture the qualitative and quantitative features of the distributions found in the data [21, 22].

Parameter fitting in stochastic models is harder [2–4], although it can offer important insights, particularly in systems biology where fitting of statistics or distributions can reveal some information about the underlying biological parameters or mechanisms. For example in [23], the probability distribution of nascent RNA was obtained by a stochastic model. It predicted hitherto unobserved discontinuities and periodic peaks in the distribution, which were then verified experimentally. Aslo in [24], parameter identification and cross-validation analyses were employed to choose, among many stochastic model hypotheses, the best model that fits the data without losing its predictive power because of overfitting. There have been other works that gained valuable insights from using stochastic models [11, 12]. Moreover, while rate constants derived from deterministic parameter values have been used in many published CME models, the noise in the system can generate dynamics that are different from the predictions of deterministic models [2, 25]. Using deterministic parameters in a stochastic model can thus be deceptive. Hence, parameter inference in stochastic models is a relevant problem.

This has kindled the interest of several recent efforts aimed at facilitating the task [2, 22, 25, 26]. In this study, we use the concept of maximum likelihood [1–7], which has been regarded as a natural approach given the probabilistic nature of stochastic models [20]. The general principle of maximum likelihood parameter estimation [7, 23, 24] is to find the parameters with which the mathematical model can reproduce the distributions in the experiments by using the likelihood of the data given a parameter set as the objective function for an optimization problem. Hence, fitting parameters in a stochastic model using maximum likelihood is essentially an optimization problem. What makes this problem challenging is that there can be confounded parameters, an identifiability issue or multimodality of the likelihood surface [20]. Although there have been comparisons of different derivative-free optimization schemes [27, 28], there is no single optimization scheme that performs best across all test problems [27]. Because of this, it is important to compare different optimization algorithms in the specific context of parameter fitting using the maximum likelihood. Here, we compare the performances of five optimization algorithms (three local and two global), representing some of the popular optimization techniques.

The rest of the paper is organized as follows: section 2 forms the CME for the particular gene regulation case under consideration. The likelihood of experimental observations given some parameter set is defined in section 3, as well as the parameter fitting scheme as an optimization process. The likelihood function is found by solving the CME, which is formidable and further compounded with the many function evaluations required for the optimization problem. Section 4 outlines the Krylov-FSP-SSA algorithm, which is a powerful numerical component that we use for this purpose. We report some numerical tests in section 5, followed by a discussion and some concluding remarks in section 6.

2. The chemical master equation (CME)

The application considered in this study is found in Min Wu *et al* [8, 9], where their mathematical model uses a set of two ODEs to characterize the interaction of the two proteins:

$$\frac{\mathrm{d}[\mathrm{LacI}]}{\mathrm{d}t} = c_{rl} + p_{e,\mathrm{tet}} \cdot (c_{il} - c_{rl}) - \delta \cdot [\mathrm{LacI}] \qquad (1)$$

$$\frac{\mathrm{d}[\mathrm{TetR}]}{\mathrm{d}t} = c_{rt} + p_{e,\mathrm{lac}} \cdot (c_{it} - c_{rt}) - \delta \cdot [\mathrm{TetR}]. \quad (2)$$

We will detail the variables $p_{e,\mathrm{tet}}$, $p_{e,\mathrm{lac}}$, and constants c_{il}, c_{rl}, c_{it}, c_{rt}, δ when describing the CME. We will see that there is a total of 11 parameters, 6 of which are estimated from previous experiments and the remaining 5 are to be fitted.

To describe the stochastic alternative based on the CME, we define the *state vector* consisting of 2 proteins species: TetR and LacI, and represented as

$$\mathbf{x} = ([\mathrm{TetR}], [\mathrm{LacI}])^T \quad (3)$$

where [TetR] and [LacI] can be any nonnegative integer counting the corresponding proteins. We model the interaction between the two species using the following reactions:

$$\varnothing \xrightarrow{\kappa_1} \mathrm{LacI} \quad (4)$$

$$\mathrm{LacI} \xrightarrow{\kappa_2} \varnothing \quad (5)$$

$$\varnothing \xrightarrow{\kappa_3} \mathrm{TetR} \quad (6)$$

$$\mathrm{TetR} \xrightarrow{\kappa_4} \varnothing, \quad (7)$$

where κ_i is the reaction rate of reaction i. The formula for these rates can be found in [8] and are summarized below.

The quantity $p_{e,\mathrm{tet}}$ is the probability of TX (the promoter for LacI) to not be bound by TetR. Given the state vector at the moment, it can be defined as

$$K_I = k_{\mathrm{ATc}} \cdot [\mathrm{TetR}] \quad (8)$$

$$f_I = \left(\frac{K_I}{K_I + [ATc] \cdot k_t}\right)^m \quad (9)$$

$$p_{e,\mathrm{tet}} = \frac{k_t^{n_t}}{k_t^{n_t} + ([\mathrm{TetR}] \cdot f_I)^{n_t}} \quad (10)$$

where the parameter k_{ATc}, nonlinearity constant n_t, and k_t (defined to be the active [TetR] needed so that $p_{e,\mathrm{tet}} = 50\%$) are to be fitted. Note that from fitting the Hill coefficient of induction of ATc to the dose response curves, we have

$$m \cdot n_t = 11.5 \quad (11)$$

and therefore only need to find n_t.

If TX is not bound by TetR (with probability $p_{e,\mathrm{tet}}$), the production rate of LacI is c_{il} (min^{-1}). However, if TetR does not bind to TX (with probability $1 - p_{e,\mathrm{tet}}$), the production rate of LacI is c_{rl} (min^{-1}). Therefore we have:

$$\kappa_1 = p_{e,\mathrm{tet}} \cdot c_{il} + (1 - p_{e,\mathrm{tet}}) \cdot c_{rl} \quad (12)$$

$$= c_{rl} + p_{e,\mathrm{tet}} \cdot (c_{il} - c_{rl}). \quad (13)$$

Similarly, $p_{e,\mathrm{lac}}$ is defined as the probability of LX (the promoter for TetR) to not be bound by LacI, and is given as

$$p_{e,\mathrm{lac}} = \frac{k_l^{n_l}}{k_l^{n_l} + [\mathrm{LacI}]^{n_l}} \quad (14)$$

where the nonlinearity constant n_l and parameter k_l (active [LacI] needed so that $p_{e,\mathrm{lac}} = 50\%$) are unknown. We then have

$$\kappa_3 = c_{rt} + p_{e,\mathrm{lac}} \cdot (c_{it} - c_{rt}) \quad (15)$$

where c_{rt} (min^{-1}) and c_{it} (min^{-1}) are TetR production rates when LX is repressed or induced, respectively.

Finally, because TetR and LacI are both very stable proteins, the decrease of intracellular abundance of these repressors is through cell division. Yeast cells grown in galactose media have doubling times of about 6 h [8, 9], corresponding to

$$\kappa_2 = \kappa_4 = \delta \approx 0.002\ (\mathrm{min}^{-1}). \quad (16)$$

We can then define the propensities of the four reactions given the state of the system, which are the probabilities of them occurring during an infinitesimal time interval $[t, t + \mathrm{d}t)$:

$$\alpha_1 = c_{rl} + p_{e,\mathrm{tet}} \cdot (c_{il} - c_{rl}) \quad (17)$$

$$= c_{rl} + \frac{k_t^{n_t} \cdot (c_{il} - c_{rl})}{k_t^{n_t} + \left[[\mathrm{TetR}] \cdot \left(\frac{k_{\mathrm{ATc}} \cdot [\mathrm{TetR}]}{k_{\mathrm{ATc}} \cdot [\mathrm{TetR}] + [ATc] \cdot k_t}\right)^m\right]^{n_t}} \quad (18)$$

$$\alpha_2 = \delta \cdot [\mathrm{LacI}] \quad (19)$$

$$\alpha_3 = c_{rt} + p_{e,\mathrm{lac}} \cdot (c_{it} - c_{rt}) \quad (20)$$

$$= c_{rt} + \frac{k_l^{n_l} \cdot (c_{it} - c_{rt})}{k_l^{n_l} + [\mathrm{LacI}]^{n_l}} \quad (21)$$

$$\alpha_4 = \delta \cdot [\mathrm{TetR}] \quad (22)$$

where the parameters k_{ATc}, k_t and k_l, and nonlinearity constants n_t and n_l are unknown and need to be fitted by the experimental data (note that since these are dimensionless quantities used to calculate $p_{e,\mathrm{tet}}$ and $p_{e,\mathrm{lac}}$, they do not require any unit).

The other parameters are fixed and come from experiments with promoters [9]:

$$c_{rl} = 7.46\ (\mathrm{min}^{-1}) \quad (23)$$

$$c_{il} = 918\ (\mathrm{min}^{-1}) \quad (24)$$

$$c_{rt} = 13.06\ (\mathrm{min}^{-1}) \quad (25)$$

$$c_{it} = 717.38\ (\mathrm{min}^{-1}) \quad (26)$$

$$m = \frac{11.5}{n_t} \quad (27)$$

where the last equation comes from (11).

We are interested in the probability that the system is at any given state $\mathbf{x}$ at time t, denoted as $P(\mathbf{x}t) = \mathrm{Prob}\{\mathbf{x}(t) = \mathbf{x}\}$. Knowing the number

of each species at $t = 0$ (from which $P(\mathbf{x}, 0)$ can be deduced), the CME [29] states that

$$\frac{dP(\mathbf{x}, t)}{dt} = \sum_{k=1}^{4} \alpha_k(\mathbf{x} - \boldsymbol{\nu}_k)P(\mathbf{x} - \boldsymbol{\nu}_k, t) - \sum_{k=1}^{4} \alpha_k(\mathbf{x})P(\mathbf{x}, t) \tag{28}$$

where the stoichiometric vector $\boldsymbol{\nu}_k$ represents the change in species numbers if reaction k occurs.

Assuming that there are n possible states, ordered as $\mathbf{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$, then equation (28) can be rewritten as

$$\dot{\mathbf{p}}(t) = \mathbf{A} \cdot \mathbf{p}(t) \tag{29}$$

where $\mathbf{p} = (P(\mathbf{x}_1, t), \ldots, P(\mathbf{x}_n, t))^T$ and the transition rate matrix $\mathbf{A} = [a_{ij}] \in \mathbb{R}^{n \times n}$ is defined as

$$a_{ij} = \begin{cases} -\sum_{k=1}^{4} \alpha_k(\mathbf{x}_j), & \text{if } i = j \\ \alpha_k(\mathbf{x}_j), & \text{if } \mathbf{x}_i = \mathbf{x}_j + \boldsymbol{\nu}_k \\ 0, & \text{otherwise} \end{cases}. \tag{30}$$

The solution to (29) is the probability vector at the end point t_f:

$$\mathbf{p}(t_f) = \exp(t_f \mathbf{A})\mathbf{p}(0) \tag{31}$$

where the exponential matrix is defined as

$$\exp(t_f \mathbf{A}) = \sum_{\ell=0}^{\infty} \frac{(t_f \mathbf{A})^\ell}{\ell!}. \tag{32}$$

The agreement between the dynamics described by the CME and the original ODEs [8] will be shown in section 5.

3. Parameter fitting

Parameter inference in stochastic biochemical systems has been less developed than in deterministic models [4]. Maximum likelihood [7, 23, 24] represents a natural approach for this problem because of the probabilistic nature of stochastic models. In this section we will define the likelihood function for the specific experimental data of Min Wu *et al* [8], but to help make the contrast with our approach clear, we first briefly describe their approach to parameter fitting using the ODE model.

There are two sets of lab experiments in [8]. In the first experiment, TXLX[2] cultures are treated with full ATc induction (250 ng ml^{-1}) for 48 h. In the second experiment, the cultures are treated with no ATc induction during the same time frame. Both cultures are then rediluted into media containing different ATc levels and the yEGFP measurements are recorded. The normalized GFP plots, as seen in figures 1(d)–(f) in [8] for the two experiments, do not coincide, which indicates bimodality. The ATc region where the plots are distinct is called the bistable region [8].

The goal of the fitting scheme in their work [8] is to find the parameters (k_{ATc}, n_t, k_t, n_l and k_k) so that the bistable region predicted by the mathematical model fits the experimental data. A range for each parameter is specified, so that they have biologically reasonable values. Random parameter sets are then generated uniformly from these regions. The bistable region for each set is then calculated, and only those whose bistable regions are within 10% relative error from the experimentally established region are kept.

We explore here a more general approach to fitting, in which the goal is not to fit the bistable region but to find the parameter set so that the frequencies shown in the experimental data can be captured in the mathematical model. Since in general, the flow cytometry measurements performed at different time points for different experiments produce histograms of the protein numbers, the goal of our approach of parameter fitting is to calibrate the parameters, which are k_{ATc}, n_t, k_t, n_l and k_k in this application, so that the probability distributions predicted by the mathematical model at these time points fit the experimental results. This can be formulated using the definition of likelihood function.

Suppose that N cells were under observation, and the ith cell was measured at time point t_i of experiment e_i and found to belong to the state $\mathbf{x}_i = ([\text{TetR}]_i, [\text{LacI}]_i)^T$. Assuming a parameter set $\theta = (k_{\text{ATc}}, n_t, k_t, n_l, k_k)^T$, we can solve the CME to compute the probability that a given cell is in that state, which is $p(\mathbf{x}_i|\theta, e_i, t_i)$. The total likelihood of all observations, $L(\mathbf{D}|\theta)$, is the product of the probabilities of all observed cells:

$$L(\mathbf{D}|\theta) = \prod_{i=1}^{N} p(\mathbf{x}_i|\theta, e_i, t_i). \tag{33}$$

The problem of parameter fitting is then to find the parameter set θ_{Fit} that maximizes this likelihood, or equivalently, the logarithm of the likelihood:

$$\theta_{\text{Fit}} = \arg\max_\theta(L(\mathbf{D}|\theta)) \tag{34}$$

$$= \arg\max_\theta(\log(L(\mathbf{D}|\theta))) \tag{35}$$

$$= \arg\max_\theta(\sum_{n=1}^{N} \log(p(\mathbf{x}_i|\theta, e_i, t_i))). \tag{36}$$

An optimization routine is required to find θ_{Fit}. To conduct parameter searches, we employ five different optimization algorithms: PRAXIS [30], NELMIN [31, 32] and NEWUOA [33, 34], representing local optimization approaches, together with GLOBAL [35–37] and SIMANN [38, 39] which are two global optimization algorithms.

We note that there have been other works that dealt with modeling and analyzing experimental data of a genetic toggle switch, some of them very similar to the

[2] In [8], the gene network constructed using the TX and LX synthetic promoters is called TXLX.

model discussed here [1, 21, 22] but with differences in the parameter fitting schemes. In [21] and [22], Munsky and Khammash fitted single-cell data using statistical quantities, such as the mean levels, marginal distributions, or full joint distributions, employing the FSP to compute the solutions to the CME. The fitted parameter arguments are then chosen to minimize the difference between the measured statistical quantity and the numerical solution of that quantity, using the 1-norm since the FSP naturally provides exact bounds on the 1-norm error of the solution. Their search was run using multiple iterations of fminsearch in MATLAB, which is a local optimization algorithm, and a simulated annealing algorithm. In this work, we use the concept of maximum likelihood to fit the parameters instead.

4. The Krylov-FSP-SSA algorithm

In the maximum likelihood approach, the CME is repeatedly solved over a large number of parameter sets, from which the likelihood of each parameter set can be computed and the parameters with the maximal likelihood is chosen. Generally, solving the CME is a formidable task. There are infinitely many states that the system can occupy when the copy numbers of species in the system are not bounded (as in the problem of interest here). Even when the copy numbers are bounded the size of the state space is prone to explosive growth (this is usually referred to as the 'curse of dimensionality'). The choice of the CME solver is therefore crucial to the effectiveness of this approach. The stochastic simulation algorithm (SSA) or other Monte Carlo methods were chosen in many maximum likelihood works [2–6]. They avoid the curse of dimensionality by drawing random trajectories of the system and using the resulting frequencies to indirectly approximate the true probability distributions.

Recently, there has been much interest in the finite state projection (FSP) [40–42], which presents a direct approach for solving the CME. It tackles the curse of dimensionality by applying upper bounds on each species number, thereby restricting the full state space $\mathbf{X}$ to a finite subspace $\mathbf{X}_J$ indexed by J. An advantage of solving the CME directly by the FSP is that unlike Monte Carlo methods, such as the SSA [43–45] or its many improved variants [46–55], the FSP possesses an analytical bound on the error of the resulting probability distributions. As the number of states taken into account in the FSP is increased, this bound is decreased and the probability of any given state of the system is more accurate. This contrasts with Monte Carlo methods, where the error is statistical.

Building on the original FSP [40–42], other works have sought improvements [56–61] (see [62] for an overview and detailed analysis of the current algorithms in the FSP family). Among these, the more successful variants transform the FSP method into an adaptive time-stepping algorithm by dividing $[0, t_f]$ into small intervals

$$0 = t_0 < t_1 < \cdots < t_{K+1} = t_f, \quad \tau_k = t_{k+1} - t_k. \tag{37}$$

At each time point $t_k, k = 0, 1, \ldots K$, one would:

(1) Pick a reduced state space $\mathbf{X}_{J_k}$ that contains the most likely states over $[t_k, t_k + \tau_k]$ and update $\mathbf{A}_{J_k}$ accordingly.
(2) Approximate (using a truncated $\mathbf{p}_{J_k}$ and padding the end result as necessary for consistency)

$$\mathbf{p}(t_{k+1}) \approx \exp(\tau_k \mathbf{A}_{J_k}) \mathbf{p}_{J_k}(t_k) \tag{38}$$

and then move on to the next iteration.

Performing both phases efficiently is crucial to the success of these time-stepping approaches. For phase (1), if $\mathbf{X}_{J_k}$ is too off the mark, the probability mass will escape and the error will be too large, but a too broad state space means a bigger $\mathbf{A}_{J_k}$ and its matrix exponential can be very expensive to compute. For phase (2), an efficient algorithm for calculating the action of the matrix exponential is necessary.

Among existing implementations, the Krylov-FSP-SSA method by Sidje and Vo [63] turns out to be reliable and efficient. The method uses SSA trajectories to find the likely states during the interval $[t_k, t_k + \tau_k]$ of small length τ_k, and Krylov techniques [64] for evaluating the matrix exponential, which are among the most effective strategies, especially when the matrix is large and sparse [65]. Since the state space is kept compact, containing only the most likely states of the system, $\mathbf{A}_{J_k}$ is usually considerably smaller than it would be in the original FSP algorithm, and therefore the time taken by the matrix exponential is even further reduced.

We performed a trial comparison between the Krylov-FSP-SSA and another FSP implementation [21]. We observed that, across 100 evaluations with randomized parameter sets, they achieved comparable accuracy but the former took on average 23s while the latter took 134s (section 5 has more details). It should be noted that there are many different FSP implementations, and choosing the right algorithm for any specific problem is not always obvious. However, the Krylov-FSP-SSA algorithm proved to be a powerful enough tool for our task that involves repeated solves of the CME during the optimization process.

5. Numerical tests

5.1. Computing platform

All tests reported here utilized resources of the Alabama Supercomputer Authority, which at the time of writing houses two supercomputers called SGI UV and DMC. The user can request a job to be executed on either of them, or can simply let the operating system select the more suitable system depending on the workload and availability. All codes were written in FORTRAN 77

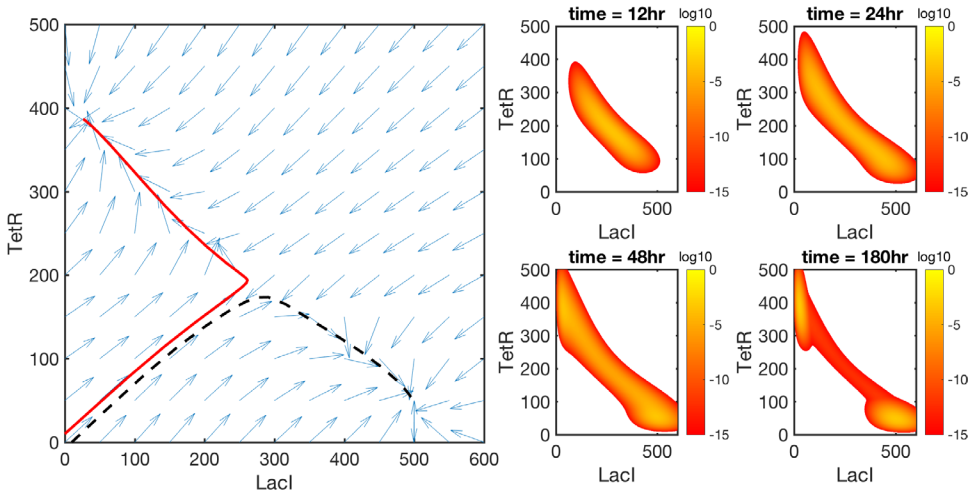

Figure 2. Left panel: vector field of the ODEs [8] when the parameters are $k_{\mathrm{ATc}} = 0.94, k_t = 11, n_t = 1.56, k_l = 264, n_l = 3.35$. The red solid line and the black dashed line represent two solutions of the ODEs when the initial state is ([TetR], [LacI]) = (10, 0) and (0, 10), respectively. Four panels on the right: evolution of the probability distribution at time 12, 24, 48 and 180 h from solving the CME with the same parameter set.

and were run on the large queue of the SGI UV with 1 processor core (Xeon E5-4640 CPU operating at 2.4 GHz), 360 h time limit and 1GB memory limit.

5.2. Comparison between the CME and ODE models

In section 2, we rewrote the deterministic ODE system in [8] into the stochastic CME. Therefore it is important to show that the two models indeed describe the same evolution in protein counts, which will be confirmed now.

Figure 2 shows the vector field of the ODE model, which dictates the evolution in numbers of TetR and LacI in any cell. The ODEs are solved for two different initial states:

$$([\mathrm{TetR}]\,,[\mathrm{LacI}]\,) = (0, 10) \tag{39}$$

and

$$(\,[\mathrm{TetR}]\,,[\mathrm{LacI}]\,) = (10, 0) \tag{40}$$

and the solutions are superimposed on the vector field (the black dashed line and the red solid line, respectively). Even though the initial states are close to each other, the trajectories branch out to two different steady states. This demonstrates the stochasticity of the system, where a small change in the protein counts at the beginning can lead to two different cell fates.

The CME is then solved with the Krylov-FSP-SSA algorithm for the same parameter set and the resulting transient probability distributions are also shown in figure 2. As predicted in the vector field of the ODEs, the probability mass first drifts toward the unstable steady state, then it is divided between the two modes of the bimodal distribution.

There are several observations to be drawn from this numerical test. First of all, the fact that the distributions resulting from the CME agree with the vector field of the ODEs implies that the two models describe the same problem, confirming the reliability of the CME. Second, even though the ODEs' vector field can predict both the unstable and stable states, it cannot produce the transient distributions which are required for computing the likelihood function. These transient distributions also give a clearer picture of the cell's fate. For example, solving the ODEs with initial state ([TetR] , [LacI]) = (0, 0) only results in one steady state. However, the stochastic nature of the system implies that the system might end up in the second steady state instead. The CME not only predicts that but also shows the probability for the cell to commit to either outcome. This is difficult to do using the ODE model.

5.3. Comparison between Krylov-FSP-SSA and SSA

Having checked that the CME model is compatible with the original ODE model in [8], we will now assess the choice of the Krylov-FSP-SSA as the CME solver for computing the likelihood function instead of Monte Carlo methods such as the SSA.

Figure 3 compares the probability distribution when the parameters are:

$$k_{\mathrm{ATc}} = 0.94, k_t = 11, n_t = 1.56, k_l = 264, n_l = 3.35 \tag{41}$$

at 48 h, when the system is in equilibrium, by using on one hand the Krylov-FSP-SSA algorithm, and on the other hand 100 000 trajectories of the classic SSA, which took 2 min in runtime. A clear contrast here is that solving the CME by the Krylov-FSP-SSA algorithm took about 10 s. Recall also the advantage of FSP algorithms to offer computations accurate to an *a priori* threshold (set here to be 10^{-5}).

In previous works of maximum likelihood estimation for parameter inference in stochastic models [1–6], the SSA or other Monte Carlo approaches were used for computing the likelihood function. However, the large number of different parameter sets to be examined means that it is not realistic to compute

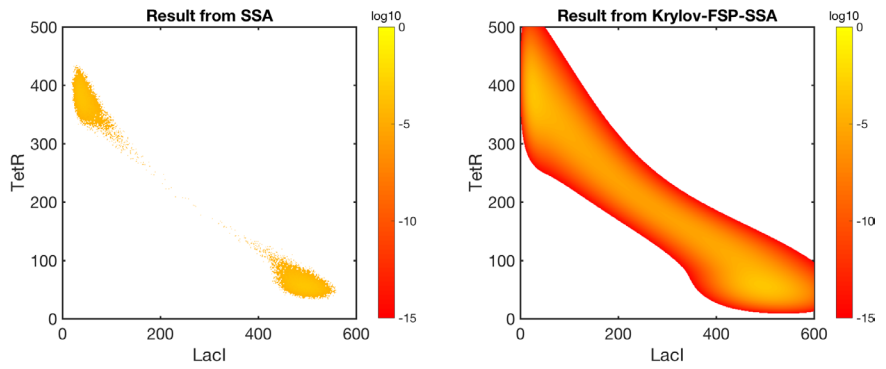

Figure 3. The probability distribution at 48 h, computed by 100 000 SSA runs (left plot) and by the Krylov-FSP-SSA algorithm (right plot). The parameters are $k_{\mathrm{ATc}} = 0.94$, $k_t = 11$, $n_t = 1.56$, $k_l = 264$, $n_l = 3.35$.

many realizations for each parameter set. The resulting distributions might therefore be incomplete. By contrast, for small gene regulation problems, where the state space is small enough and the integration interval is not too long, FSP algorithms can supply the probabilities for many more states in a reasonable short runtime.

5.4. Comparison between Krylov-FSP-SSA and original FSP

There has not been a systematic comparison between different FSP implementations in a wide range of biological problems. The Krylov-FSP-SSA algorithm was chosen here for several reasons. First, it can be used without prior assumptions on the model. Several other FSP algorithms require bounds on the state space, and this cannot be known without trial runs to find the areas on the state space that accumulate the most probability mass. The Krylov-FSP-SSA algorithm, however, finds the state space on the fly by following the direction of a few SSA runs and therefore does not need *a priori* bounds on the protein counts. Second, with its time-stepping feature, the Krylov-FSP-SSA algorithm can be more efficient than other FSP algorithms.

To check its effectiveness for the model under consideration in this study, we compare it to a FSP implementation by Munsky [21] with the parameter set (41). Munsky's FSP implementation was translated from its original MATLAB code to FORTRAN for a fair comparison, since it is well-known that FORTRAN is magnitudes faster than MATLAB.

The two algorithms were tested for 100 different parameter sets in the parameter space.

Each parameter set is randomly picked from the uniform distribution in its range, chosen to be the same as that used in [8]:

$$0.01 < k_{\mathrm{ATc}} < 1 \tag{42}$$

$$1 < k_t < 400 \tag{43}$$

$$1 < n_t < 5 \tag{44}$$

Table 1. Comparison between the Krylov-FSP-SSA [63] and Munsky's FSP implementation [21] using 100 evaluations with randomized parameter sets to compute the averages.

Average runtime of the Krylov-FSP-SSA	23 s
Average runtime of the FSP in [21]	134 s
Average relative 1-norm error (47)	1.23×10^{-2}

$$1 < k_l < 400 \tag{45}$$

$$1 < n_l < 5. \tag{46}$$

For each parameter set, we record the runtime for finding the probability distributions by either algorithm and then computing the likelihood function based on the distributions. The average runtime of each algorithm is shown in table 1. It is also crucial to check that the two algorithms give the same likelihood value. For this, we compute the relative 1-norm error for each parameter set θ:

$$relerr = \frac{|L_1(\mathbf{D}|\theta) - L_2(\mathbf{D}|\theta)|}{|L_2(\mathbf{D}|\theta)|} \tag{47}$$

where $L_1(\mathbf{D}|\theta)$ is the likelihood computed by the Krylov-FSP-SSA algorithm, and $L_2(\mathbf{D}|\theta)$ is computed by Munsky's FSP implementation. The average relative 1-norm error of the 100 parameter sets is shown in table 1.

We can clearly see that even though the resulting likelihoods are practically the same between the two algorithms, the Krylov-FSP-SSA has a much shorter average runtime. This is because there are a number of stark differences between them, notably the fact that the Krylov-FSP-SSA is a time-stepping algorithm, unlike Munsky's implementation in [21].

It is important to note that there are other FSP variants, many of which are time-stepping [57–59, 61] and some might be more efficient than the Krylov-FSP-SSA in some instances. However, there has not yet been an in-depth numerical comparison between the variants, and the Krylov-FSP-SSA was retained because of its availability and its satisfactory performance for our job.

Table 2. Input parameters of the local optimization algorithms.

	Input parameters	Value	Definition
PRAXIS	T_0	10^{-3}	Error tolerance
	MACHEP	2.22×10^{-16}	Machine precision
	H_0	0.1	Maximum stepsize
	FMIN	10^6	Estimate of minimum (used only for printing)
NELMIN	STEP	[0.1, 0.1, 1, 1, 0.05]	Size and shape of the initial simplex
	REQMIN	10^6	Terminating limit for the variance of the function values
	KONVGE	10	Convergence check
	KCOUNT	1000	Maximum number of function evaluations
NEWUOA	NPT	11	Number of interpolation conditions
	RHOBEG	0.8	Initial value of a trust region radius
	RHOEND	1	Final value of a trust region radius
	MAXFUN	1000	Maximum number of function evaluations

With the CME solver chosen, the final piece of the puzzle is to pick an optimization algorithm for finding the parameter set with maximum likelihood. There are many different derivative-free optimization schemes and their variants. There have also been works to compare these schemes in a variety of test problems, e.g. [27, 28]. Overall, the performance of the optimization schemes depends on the specific problems, and there is no single optimization scheme that is guaranteed to perform best in all circumstances [27].

Therefore, in this study we compare some popular local and global optimization algorithms for the specific task of stochastic parameter fitting with maximum likelihood. All of them are readily available online in FORTRAN and represent different optimization approaches.

Some of the optimization routines being investigated are maximization schemes, while others are minimization schemes. In the latter case, the sign of the likelihood function is simply reversed. Also, the routines originally written using the single precision REAL data type have all been changed to DOUBLE PRECISION for a fair comparison.

5.5. Local optimization schemes

We include three different local optimization algorithms for the fitting scheme:

(1) PRAXIS [30], one of the first derivative-free optimization solvers developed, using Powell's method of conjugate search directions
(2) NELMIN [31, 32], an implementation of the Nelder–Mead algorithm for derivative-free optimization
(3) NEWUOA [33, 34], implementing Powell's model-based algorithm using trust regions.

Aside from an initial guess for the parameter set, these routines require the input parameters shown in table 2. These values are recommended in the codes and are therefore used here.

As will be shown later, the performance of these optimization algorithms depends heavily on the starting guess: if the initial guess for the parameters is too far away from the correct parameters, the algorithms are less likely to provide a good output. The experimental data [8] consists of the protein numbers at different time points. To compare the performances of the optimization codes, we produce *synthetic data* by solving for the distribution vectors resulting from the CME with the initial state ([TetR], [LacI]) = (0, 0) with the true parameter set (41) at 5 different time points: 1 h, 6 h, 12 h, 24 h, and 48 h. For each time point, 100 000 samples are randomly drawn from the distribution vectors.

The frequencies of the protein counts at the 5 time points are the input for the fitting scheme and the goal is to find the parameter set that can recreate the distribution in the synthetic data. We remark here that synthetic data allows us to easily test our fitting method and the performance of the algorithms on a variety of input, knowing that the results on the synthetic data are indicative of the results to be expected on experimental data. There is a limit of maximum 1000 function evaluations, which is adequate for these algorithms to converge to some maxima.

The biological model [8], as is the case with most models, defines specific ranges for the parameters, which were given in (42)–(46). During the process, if the parameter set proposed from the optimization scheme is out of this range, its likelihood is defined to be a very small number (-10^{-21}) to dissuade the scheme from traveling in this direction.

To highlight how the initial guess plays a crucial role when using a local optimization scheme, we implement all three schemes from three different initial guesses. In the first test:

$$k_{\mathrm{ATc}} = 0.2, k_t = 250, n_t = 4, k_l = 55, n_l = 3 \quad (48)$$

in the second test:

$$k_{\mathrm{ATc}} = 0.9, k_t = 13, n_t = 1, k_l = 255, n_l = 3 \quad (49)$$

and finally in the third test:

$$k_{\mathrm{ATc}} = 0.8, k_t = 8, n_t = 2, k_l = 280, n_l = 4. \quad (50)$$

Table 3. Results of the local optimization algorithms.

		n_t	n_l	k_t	k_l	k_{ATc}	Number of function evalutations	Likelihood
Synthetic data		1.56	3.35	11	264	0.94		−1873 172.8
Test 1	Initial guess	4	3	250	55	0.20		−6329 761.9
	PRAXIS	4.02	5.00	249.93	55.01	0.08	131 (33 linear searches)	−6314 107.0
	NELMIN	2.26	4.98	245.05	76.39	0.02	193	−6294 412.4
	NEWUOA	4.00	3.80	250.00	55.00	0.60	14	−6321 248.2
Test 2	Initial guess	1	3	13	255	0.90		−3659 419.6
	PRAXIS	1.87	3.31	13.33	265.27	1.00	268 (83 linear searches)	−1884 653.9
	NELMIN	1.25	3.05	9.17	255.02	0.97	1007	−1923 230.3
	NEWUOA	1.20	3.00	13.00	255.00	0.90	13	−3549 849.8
Test 3	Initial guess	2	4	8	280	0.80		−5413 880.6
	PRAXIS	1.47	3.35	8.24	264.32	0.68	346 (121 linear searches)	−1873 192.5
	NELMIN	1.65	3.55	12.14	277.00	0.99	1004	−1905 593.2
	NEWUOA	2.00	4.00	8.00	280.00	0.40	13	−2037 143.6

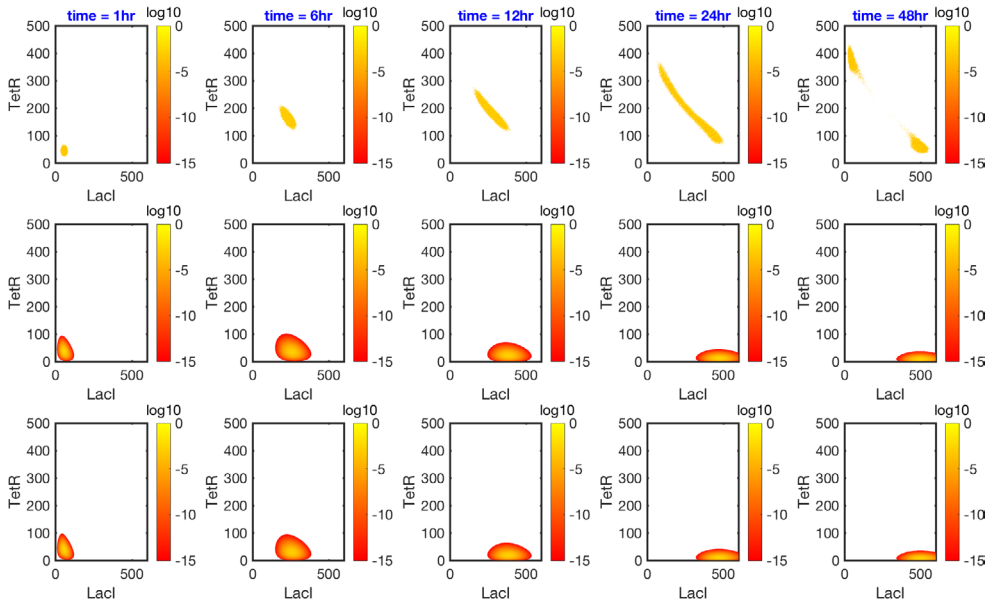

Figure 4. (Test 1) Result of the optimization algorithms with starting parameter guess: $k_{\mathrm{ATc}} = 0.2, k_t = 250, n_t = 4, k_l = 55, n_l = 3$. First row: the synthetic data consisting of protein counts for 100 000 cells per time point, computed by SSA with the true parameters $k_{\mathrm{ATc}} = 0.94, k_t = 11, n_t = 1.56, k_l = 264, n_l = 3.35$. Second row: the distributions at corresponding time points with the starting parameter guess. Third row: the distributions at corresponding time points with the final parameter guess.

The numbers of function evaluations that each scheme requires, the final optimal parameter set and its likelihood are shown in table 3.

As can be seen in table 3, the initial guess for the first test is very far from the true parameter set used to produce the synthetic data, resulting in a small likelihood. The final results from the local optimization schemes from this initial guess, therefore, only slightly improve the likelihood. The distributions they produce are similar and shown in figure 4. These distributions fail to recreate the distributions observed in the synthetic data, as expected.

The initial guess for the second test is closer to the true parameter set, and the final results from PRAXIS and NELMIN reflect this: their optimal likelihoods are greatly increased from the initial likelihood, and are very close to the true value. Between these two algorithms, the result from PRAXIS is better (larger final likelihood) and the algorithm converges after a much smaller number of function evaluations. On the other hand, NEWUOA converges after only 13 iterations and only marginally increases the likelihood. The distributions resulting from PRAXIS and NELMIN are shown in figure 5. As reflected in the large likelihood, the final result recreates the bimodality in the synthetic data and the evolution of probability distribution over time. Even though the initial guess produces distributions that are very different from the data, the optimization codes can easily calibrate the parameters to maximize the likelihood function and arrive at the distributions almost identical to the synthetic data.

In the third test, the small likelihood of the initial guess indicates that it is far from the optimal point. Nevertheless, the results from PRAXIS and NELMIN

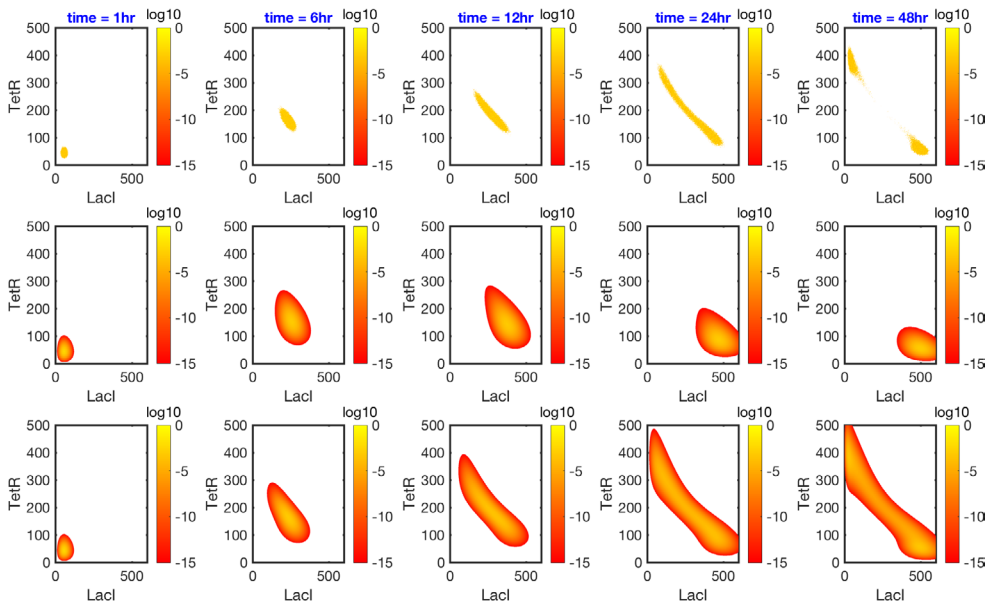

Figure 5. (Test 2) Result of the optimization algorithms with starting parameter guess: $k_{\mathrm{ATc}} = 0.9, k_t = 13, n_t = 1, k_l = 255, n_l = 3$. First row: the synthetic data consisting of protein counts for 100 000 cells per time point, computed by SSA with the true parameters $k_{\mathrm{ATc}} = 0.94, k_t = 11, n_t = 1.56, k_l = 264, n_l = 3.35$. Second row: the distributions at corresponding time points with the starting parameter guess. Third row: the distributions at corresponding time points with the final parameter guess.

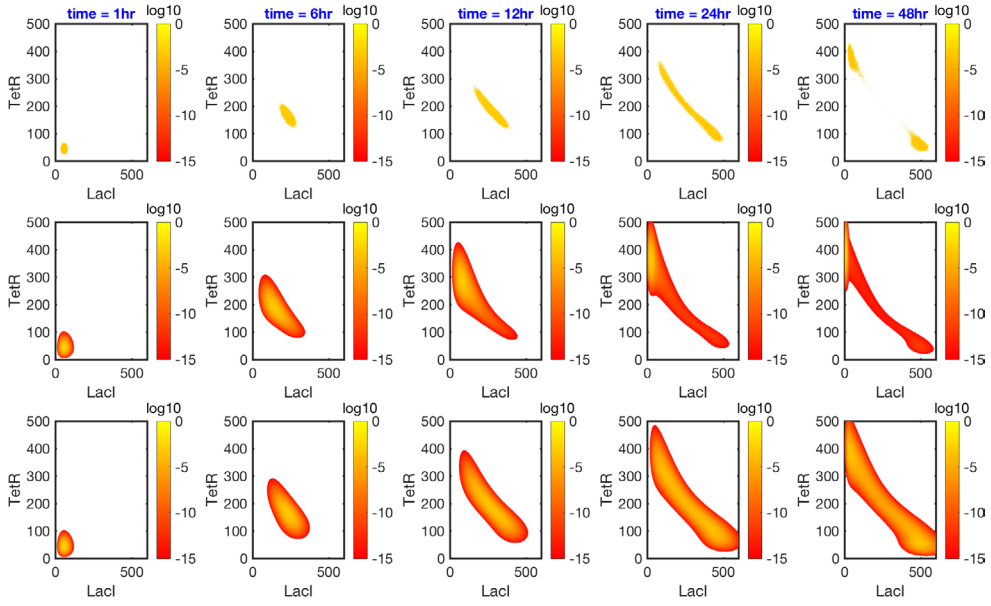

Figure 6. (Test 3) Result of the optimization algorithms with starting parameter guess: $k_{\mathrm{ATc}} = 0.8, k_t = 8, n_t = 2, k_l = 280, n_l = 4$. First row: the synthetic data consisting of protein counts for 100 000 cells per time point, computed by SSA with the true parameters $k_{\mathrm{ATc}} = 0.94, k_t = 11, n_t = 1.56, k_l = 264, n_l = 3.35$. Second row: the distributions at corresponding time points with the starting parameter guess. Third row: the distributions at corresponding time points with the final parameter guess.

are still very close to the true values. Similar to the second test, PRAXIS converges after less function evaluations to a better solution than NELMIN, and NEWUOA converges after only 13 iterations to the worst solution among the three algorithms. Figure 6 shows that the initial parameter guess produces distributions very different from the synthetic data, but the final result from the fitting scheme is similar to the frequency shown in the synthetic data.

We can conclude from these three tests that the performance of local optimization algorithms depends greatly on the initial guess. This reflects the fact that the likelihood function is difficult to optimize: it is likely not convex, and has many local maxima that these algorithms cannot escape from. Among the three algorithms, PRAXIS produces better results than NELMIN in spite of much smaller number of function evaluations needed for it to converge. NEWUOA converges quickly but its results are inferior to PRAXIS and NELMIN.

In real life applications, however, neither the true parameter set nor its likelihood are known in advance.

Table 4. Input parameters of the global optimization algorithms.

	Input parameters	Value	Definition
GLOBAL	NSIG	6	Convergence criterion
	M	1	Number of residual functions
	N100	500	Number of sample points to be drawn uniformly in one cycle
	NG0	10	Number of best points selected from the actual sample
	SEED	[1, 2, 3, 4, 5, 6]	Seeds for the random number generator
SIMANN	T_0		Initial temperature
	X	[4, 3, 250, 55, 0.2]	Initial guess for the parameter set
	RT	0.85	Temperature reduction factor
	EPS	10	Error tolerance for termination
	NS	20	Number of cycles
	NT	100	Number of iterations before temperature reduction
	NEPS	4	Number of final function values to decide upon termination
	MAXEVL	5000	Maximum number of function evaluations
	C	[2, 2, 2, 2, 2]	Vector controlling the step length adjustment
	ISEED1	1	First seed for the random number generator
	ISEED2	2	Second seed for the random number generator
	VM	[1, 1, 1, 1, 1]	Step length vector

This therefore casts doubt on employing these local optimization algorithms. A solution to this might be to start the local optimization scheme from different initial guesses, randomly chosen from the range, and select the best final result. This is the strategy of a number of global optimization algorithms, including GlobalSearch and MultiStart in MATLAB [66, 67]. An advantage of this is that each optimization run is independent from the others, and so it is possible to produce a parallel algorithm. Another strategy is to employ global optimization schemes. These algorithms focus on finding the maximum over the entire range and will be investigated in the next section.

5.6. Global optimization schemes

Two global optimization algorithms are investigated:

(1) GLOBAL [35–37], based on the Boender–Rinnooy–Stougie–Timmer algorithm [35, 36] and is a stochastic method involving sampling, clustering and local search. It was implemented in FORTRAN by Csendes [37], and the output contains up to 20 local maxima.
(2) SIMANN [38, 39], a simulated annealing algorithm [68–71].

The input parameters required by these routines are shown in table 4. Similarly to the tests with local optimization schemes, the values used here are recommended by the codes when available.

The comparison of these two algorithms uses the same synthetic data in the previous section as the data for fitting parameters, as well as the range for the parameters. A limit of 5000 function evaluations is applied on each algorithm. In practice this has been shown to be adequate for good results.

Unlike GLOBAL, the algorithm SIMANN as well as other simulated annealing algorithms depend on important input parameters to produce good results. The algorithm escapes from local optima, which is important to produce better results than local optimization schemes, by accepting downhill steps. This decision is made by the Metropolis criteria using T ('temperature') and the downhill move size in a probabilistic way. The downhill move is more likely to be accepted if T and the move size are smaller.

Therefore, the importance of the parameter T in the performance of SIMANN cannot be overstated. A smaller initial T_0 might result in a step length too small, and the function evaluations gathered by the algorithm are not enough to find the optima. The choice of an optimal initial temperature T_0, however, depends on the problem and trial runs usually have to be performed in order to find the right T_0. Because of this, we performed ten different tests with SIMANN, each with a different initial temperature:

$$T_0 = 10^k, \ k = 1, \ldots, 10. \tag{51}$$

SIMANN also requires an initial guess, and in our tests this is chosen to be

$$k_{\mathrm{ATc}} = 0.2, k_t = 250, n_t = 4, k_l = 55, n_l = 3. \tag{52}$$

Note that this initial guess was chosen for Test 1 in the previous section and was shown to result in unsatisfactory solutions from the local optimization schemes.

Table 5. Results of the global optimization algorithms.

	n_t	n_l	k_t	k_l	k_{ATc}	Number of function evalutations	Likelihood
Synthetic data	1.56	3.35	11	264	0.94		−1873 172.8
GLOBAL	1.50	3.35	9.24	264.00	0.77	4572	−1873 171.5
	1.55	3.35	10.63	264.01	0.90		−1873 172.2
	1.29	3.35	4.31	263.98	0.33		−1873 181.5
	3.40	3.17	21.41	267.17	1.00		−2002 408.4
	3.54	2.95	8.16	269.13	0.22		−2078 322.3
	4.35	2.32	22.18	283.77	0.83		−2321 101.8
	4.53	4.34	18.74	329.37	0.47		−2637 311.5
	2.14	1.88	7.67	310.07	0.38		−2668 259.4
	3.07	2.11	19.95	324.27	0.99		−2743 173.7
	4.15	2.65	18.98	342.64	0.59		−2916 667.9
	3.51	2.35	2.94	346.36	0.05		−3004 177.6
	3.32	2.91	381.46	268.37	0.02		−3740 211.5
	3.08	3.02	172.91	272.32	0.76		−3744 919.8
	3.75	2.76	29.49	264.93	0.03		−3745 244.7
	4.64	3.10	73.69	266.65	0.37		−3746 429.0
	3.22	3.01	270.00	261.40	0.22		−3748 192.9
	1.51	3.21	163.64	266.39	0.47		−3750 487.6
	2.90	2.64	283.87	262.79	0.59		−3757 378.6
	1.61	3.40	283.71	281.80	0.95		−3762 507.3
	4.60	3.47	334.80	280.11	0.69		−3767 111.1
SIMANN from $T_0 = 10^1$	2.64	1.00	247.53	81.58	0.59	5000 (limit exceeded)	−5375 590.2
SIMANN from $T_0 = 10^2$	4.01	1.00	248.27	72.83	0.23	5000 (limit exceeded)	−5495 580.3
SIMANN from $T_0 = 10^3$	3.17	1.00	246.87	77.95	0.40	5000 (limit exceeded)	−5425 506.7
SIMANN from $T_0 = 10^4$	1.46	3.63	9.76	269.04	0.83	5000 (limit exceeded)	−1873 618.3
SIMANN from $T_0 = 10^5$	2.54	4.27	15.10	271.02	0.77	5000 (limit exceeded)	−1880 111.8
SIMANN from $T_0 = 10^6$	3.86	4.96	213.19	284.03	0.78	5000 (limit exceeded)	−2142 633.2
SIMANN from $T_0 = 10^7$	4.35	3.90	31.67	257.66	0.51	5000 (limit exceeded)	−2034 590.3
SIMANN from $T_0 = 10^8$	1.86	1.53	80.26	254.45	0.81	5000 (limit exceeded)	−2146 360.6
SIMANN from $T_0 = 10^9$	4.04	1.90	374.31	75.31	0.88	5000 (limit exceeded)	−2041 629.5
SIMANN from $T_0 = 10^{10}$	2.64	2.52	180.20	342.90	0.46	5000 (limit exceeded)	−2086 480.4

The results from GLOBAL and SIMANN, with different initial temperatures, are shown in table 5.

The GLOBAL algorithm finished after 4572 function evaluations. By default, it outputs 20 local maxima found during the process, shown in table 5 with decreasing likelihoods. On the other hand, the result notes that there are too many clusters, implying that there are a large number of local maxima, confirming the reason for the failure of local optimization schemes: since the likelihood surface is multimodal, they converge to the nearest local maximum and cannot escape, which is why the results depend on the initial guesses.

Despite this, GLOBAL was able to find very good results for the parameter set. The first three results have virtually the same likelihoods as the true parameter set. The parameters themselves are also very close to the true values, except k_{ATc}, which might imply that the likelihood function is not very sensitive to this parameter.

On the other hand, all SIMANN runs exceeded the limit of 5000 function evaluations, which might be a result of the tight error tolerance (the variable EPS in table 4). As expected, the SIMANN runs starting with small initial temperature ($T_0 < 10^4$) result in parameter sets with very small likelihoods. When $T_0 = 10^4$, SIMANN converges to a good result, with likelihood only slightly smaller than that of the true parameter sets. With $T_0 > 10^4$, however, the results from SIMANN are worse.

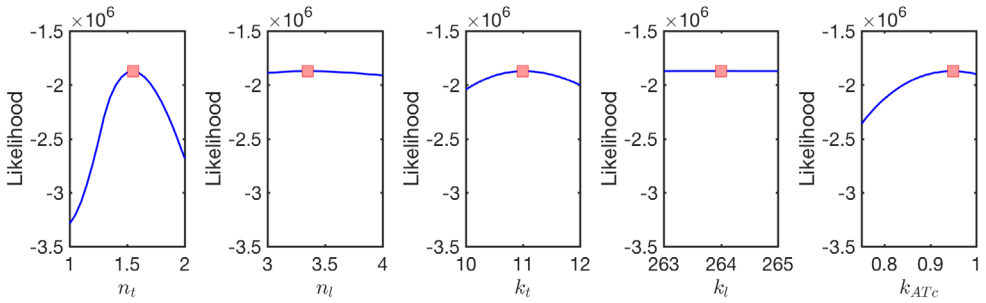

Figure 7. The sensitivity of the likelihood function of the synthetic data with respect to each parameter around the parameter set $k_{\mathrm{ATc}} = 0.94, k_t = 11, n_t = 1.56, k_l = 264, n_l = 3.35$. The panels show the change in the likelihood function when there is a change in n_t, n_l, k_t, k_l and k_{ATc}, respectively. The red squares correspond to the likelihood when the correct parameters are used.

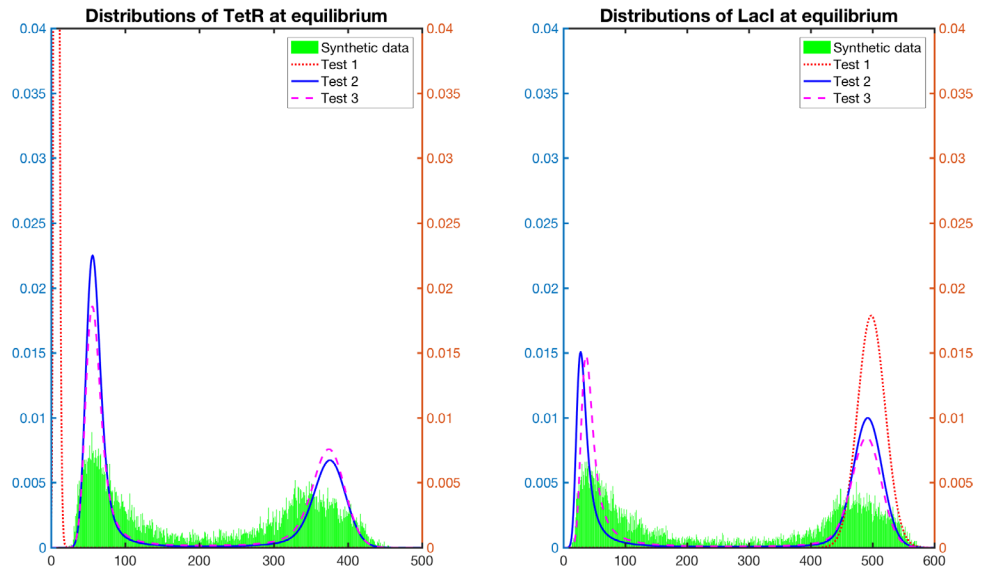

Figure 8. Comparison of the distributions at equilibrium (180 h) between the results from the different tests and the synthetic data that they are fitting.

It is important to point out, however, that even with the optimal initial temperature of $T_0 = 10^4$, the parameter set from SIMANN is not as good as the three best results from GLOBAL. GLOBAL also outputs the local maxima, which are important to draw conclusions about the likelihood function itself, as opposed to only one best parameter set as in SIMANN, and it does all this with less function evaluations. Importantly, the input parameters that GLOBAL requires are not significant to the final result, while SIMANN depends on some important input parameters which can only be set up by experience, knowledge of the problem, or trial optimization runs.

Although GLOBAL seems to be superior to SIMANN for this problem, the same conclusion cannot be drawn universally. GLOBAL in particular, and the Boender–Rinnooy–Stougie–Timmer algorithm in general, will first sample parameter sets in the given range, and then transform the parameter sets into groups around local maxima. Clustering techniques are then employed to find neighborhoods of each local maximum, and local optimization runs from each cluster can point to the global maximum. While effective for problems with few parameters, other optimization algorithms can be more efficient when there are hundreds or more parameters to be found.

5.7. Sensitivity effect

Finally, we investigate how sensitive the likelihood function of the synthetic data is with respect to each parameter in figure 7. The true parameter set (41) was used to generate the synthetic data, and the changes in the likelihood function when each parameter varies around its true value (with the other four parameters fixed to their exact numbers) are shown.

The likelihood function does not change much when k_l or n_l vary in their neighborhoods. In comparison, the likelihood responds more strongly to changes in k_t or k_{ATC}. When n_t increases from 1 to 1.56, however, the likelihood roughly increases two-fold in its value, implying that the model is most sensitive to this parameter. This sensitivity study may have important ramifications especially in parameter fitting, since not knowing the sensitivity of the parameters may lead to the codes spending time calibrating the non-sensitive parameters without getting a good result.

6. Conclusion

Synthetic biology is an effective approach to study how microbes and multicellular organisms regulate their cell fate determination when the environments change or proper development needs to be ensured. An investigation into the different properties of the network requires a mathematical model that illuminates possible regulatory mechanisms, for which a systematic approach to calibrating free parameters is required. The model can then be cross-validated using other datasets or used to predict/refine outcomes of new experiments.

Here we have investigated a mutual inhibitory gene network in Saccharomyces cerevisiae, using the model from [8, 9]. The likelihood function is evaluated by using the Krylov-FSP-SSA algorithm to solve the CME for the probability distributions given a parameter set. The likelihood function is then maximized by different optimization codes. This ensures that the solution results in a faithful portrayal of the evolution of the probability distribution over time and therefore confirms the mathematical model.

We compared for the first time different optimization algorithms for parameter fitting using maximum likelihood. There is still work to be done, as only one biological problem is considered here. There are also many different optimization approaches, and a more complete comparison between them for a variety of stochastic models will be beneficial to the systems biology community, given the importance of parameter inference in this field. The results in this work might be one step further towards establishing a parameter inference method of reference in stochastic models.

From the numerical tests in section 5, it is apparent that local optimization schemes do not perform well for our purpose. This underlines the difficulty of fitting stochastic models by maximum likelihood. The likelihood surface is often multimodal, and therefore it is difficult for the local optimization algorithms to escape a local maxima.

Figure 8 offers a comparison of the marginal distributions for both proteins from the three tests with local optimization algorithms at 180 h, where the system reaches equilibrium and clearly shows a bimodal pattern. Since the results from PRAXIS and NELMIN do not differ much in the resulted distributions even though the parameter sets are not the same, we only use the results from PRAXIS. The marginal distributions from test 1 do not match the data, with the TetR distribution being inconsistent with the synthetic data and the LacI distibution showing unimodality instead of bimodality. On the other hand, tests 2 and 3 show a bimodality in agreement with the synthetic data, although there is a slight difference in the height of the peaks of the probability distribution. Note that each peak represents one mode, or one possible fate that the cell can end up in.

In practice, choosing a starting parameter set for each local optimization run can be a challenging task. Usually, a range for each parameter is chosen so that they are biologically relevant. The parameter search is then conducted by randomly choosing different initial guesses for the parameters in these ranges, leading to thousands of function evaluations per optimization run, for which only the best solution is recorded at the end. This task can be done in an embarrassingly parallel code, since the optimization runs are independent from each other. As can be seen from table 3, the results are often satisfactory when the starting parameter guess is good. It is thus possible to have satisfactory results by employing the local optimization schemes in a parallel multi-start fashion.

On the other hand, the numerical comparison showed that global optimization algorithms produce better results than local optimization schemes, at the expense of more function evaluations. Only two global optimization schemes were considered in this study, of which GLOBAL [35–37] proved to be the better choice. This of course might change when a different biological model is tested, as GLOBAL is effective only for problems with few unknown parameters.

Our comparison considered only optimization algorithms in FORTRAN that are freely available. There have been other works comparing local and global optimization methods [27, 28] but the test cases in those works belong to different classes from that studied here. A broader and deeper comparison involving more biological models and more optimization algorithms might be essential to the biomathematical community given the importance of the parameter fitting problem.

Acknowledgments

Work supported by NSF grant DMS-1320849. The comments of the anonymous referees helped improve the paper and are gratefully acknowledged. KD would like to thank Huy Vo for his help with the codes and valuable conversations. He would also like to thank Riqi Su for his helpful information. Resources of the Alabama Supercomputer Authority were used to conduct the tests.

References

[1] Tian T, Xu S, Gao J and Burrage K 2007 Simulated maximum likelihood method for estimating kinetic rates in gene expression *Bioinformatics* **23** 84–91

[2] Poovathingal S and Gunawan R 2010 Global parameter estimation methods for stochastic biochemical systems *BMC Bioinform.* **11** 414

[3] Daigle B Jr, Roh M, Petzold L and Niemi J 2012 Accelerated maximum likelihood parameter estimation for stochastic biochemical systems *BMC Bioinform.* **13** 68

[4] Wang Y, Christley S, Mjolsness E and Xie X 2010 Parameter inference for discretely observed stochastic kinetic models using stochastic gradient descent *BMC Syst. Biol.* **4** 99

[5] Horvath A and Manini D 2008 Parameter estimation of kinetic rates in stochastic reaction networks by the em method *Proc. of the Int. Conf. on Biomedical Engineering and Informatics* vol 1 pp 713–7

[6] Reinker S, Altman R and Timmer J 2006 Parameter estimation in stochastic biochemical reactions *Syst. Biol.* **153** 168–78

[7] Fox Z, Neuert G and Munsky B 2016 Finite state projection based bounds to compare chemical master equation models using single-cell data *J. Chem. phys.* **145** 074101

[8] Wu M, Su R, Li X, Ellis T, Lai Y-C and Wang X 2013 Engineering of regulated stochastic cell fate determination *Proc. Natl Acad. Sci.* **110** 10610–5

[9] Ellis T, Wang X and Collins J J 2009 Diversity-based, model-guided construction of synthetic gene networks with predicted functions *Nat. Biotechnol.* **27** 465–71

[10] McAdams H and Arkin A 1999 It's a noisy business! genetic regulation at the nanomolar scale *Trends Genet.* **15** 65–9

[11] Fange D and Elf J 2006 Noise-induced min phenotypes in *E. Coli PLoS Comput. Biol.* **2** e80

[12] Samoilov M, Plyasunov S and Arkin A 2005 Stochastic amplification and signaling in enzymatic futile cycles through noise-induced bistability with oscillations *Proc. Natl Acad. Sci.* **102** 2310–5

[13] Elowitz M and Leibler S 2000 A synthetic oscillatory network of transcriptional regulators *Nature* **403** 335–8

[14] Colman-Lerner A, Gordon A, Serra E, Chin T, Resnekov O, Endy D, Pesce C and Brent R 2005 Regulated cell-to-cell variation in a cell-fate decision system *Nature* **437** 699–706

[15] Golding I, Paulsson J, Zawilski S and Cox E 2005 Real-time kinetics of gene activity in individual bacteria *Cell* **123** 1025–36

[16] Yu J, Xiao J, Ren X, Lao K and Lie X 2006 Probing gene expression in live cells, one protein molecule at a time *Science* **311** 1600–3

[17] Gardner T, Cantor C and Collins J 2000 Construction of a genetic toggle switch in escherichia coli *Nature* **403** 339–42

[18] Yang E, van Nimwegen E, Zavolan M, Rajewsky N, Schroeder M, Magnasco M and Darnell J Jr 2003 Decay rates of human mrnas: correlation with functional characteristics and sequence attributes *Genome Res.* **13** 1863–72

[19] Chou I and Voit E 2009 Recent developments in parameter estimation and structure identification of biochemical and genomic systems *Math. Biosci.* **219** 57–83

[20] Wilkinson D 2009 Stochastic modelling for quantitative description of heterogeneous biological systems *Nature* **10** 122–33

[21] Munsky B 2012 Chapter: Modeling cellular variability *Quantitative Biology From Molecular to Cellular Systems* (New York: Taylor and Francis) pp 234–66

[22] Munsky B and Khammash M 2010 Identification from stochastic cell-to-cell variation: a genetic switch case study *IET Syst. Biol.* **4** 356–66

[23] Xu H, Skinner S, Sokac A and Golding I 2016 Stochastic kinetics of nascent rna *Phys. Rev. Lett.* **117** 128101

[24] Neuert G, Munsky B, Tan R, Teytelman L, Khammash M and Oudenaarden A 2013 Systematic identification of signal-activated stochastic gene regulation *Science* **339** 584–7

[25] Boys R, Wilkinson D and Kirkwood T 2008 Bayesian inference for a discretely observed stochastic kinetic model *Stat. Comput.* **18** 125–35

[26] Golightly A and Wilkinson D 2006 Bayesian sequential inference for stochastic kinetic biochemical network models *J. Comput. Biol.* **13** 838–51

[27] Rios L and Sahinidis N 2013 Derivative-free optimization: a review of algorithms and comparison of software implementations *J. Glob. Optim.* **56** 1247–93

[28] Moles C, Mendes P and Banga J 2003 Parameter estimation in biochemical pathways: a comparison of global optimization methods *Genome Res.* **13** 2467–74

[29] Gillespie D 1992 A rigorous derivation of the chemical master equation *Phys. A: Stat. Mech. Appl.* **188** 404–25

[30] Brent R 1971 Algorithms for finding zeros and extrema of functions without calculating derivatives *PhD Thesis* Stanford University

[31] Nelder J and Mead R 1965 A simplex method for function minimization *Comput. J.* **7** 308–13

[32] O'Neill R 1971 Algorithm as 47: function minimization using a simplex procedure *J. R. Stat. Soc.* C **20** 338–45

[33] Powell M J D 2004 The newuoa software for unconstrained optimization without derivatives *Technical Report* NA05, Department of Applied Mathematics and Theoretical Physics, Cambridge University (https://doi.org/10.1007/0-387-30065-1_16)

[34] Powell M J D 2004 Least frobenius norm updating of quadratic models that satisfy interpolation conditions *Math. Program.* **100** 183–215

[35] Boender C, Kan A R, Timmer G and Stougie L 1982 A stochastic method for global optimization *Math. Program.* **22** 125–40

[36] Timmer G 1984 Global optimization: a stochastic approach *PhD Thesis* Erasmus University Rotterdam

[37] Csendes T 1988 Nonlinear parameter estimation by global optimization—efficiency and reliability *Acta Cybern.* **8** 361–70

[38] Goffe W, Ferrier G and Rogers J 1994 Global optimization of statistical functions with simulated annealing *J. Econ.* **60** 65–99

[39] Goffe W 2007 Simann: a global optimization algorithm using simulated annealing *Stud. Nonlinear Dyn. Econ.* **1** 169–76

[40] Munsky B and Khammash M 2006 The finite state projection algorithm for the solution of the chemical master equation *J. Chem. Phys.* **124** 044104

[41] Munsky B and Khammash M 2006 A reduced model solution for the chemical master equation arising in stochastic analyses of biological networks *45th IEEE Conf. on Decision and Control* (https://doi.org/10.1109/CDC.2006.377251)

[42] Munsky B, Peles S and Khammash M 2007 Stochastic analysis of gene regulatory networks using finite state projections and singular perturbation *American Control Conf.* (https://doi.org/10.1109/ACC.2007.4283077)

[43] Gillespie D 1976 A general method for numerically simulating the stochastic time evolution of coupled chemical reactions *J. Comput. Phys.* **22** 403–34

[44] Gillespie D 2001 Approximate accelerated stochastic simulation of chemically reacting systems *J. Chem. Phys.* **115** 1716

[45] Cao Y, Gillespie D and Petzold L 2006 Efficient step size selection for the tau-leaping simulation method *J. Chem. Phys.* **124** 044109

[46] Cao Y, Gillespie D and Petzold L 2005 The slow-scale stochastic simulation algorithm *J. Chem. Phys.* **122** 14116

[47] Cao Y, Gillespie D and Petzold L 2005 Avoiding negative populations in explicit poisson tau-leaping *J. Chem. Phys.* **123** 054104

[48] Chatterjee A, Vlachos D and Katsoulakis M 2005 Binomial distribution based tau-leap accelerated stochastic simulation *J. Chem. Phys.* **122** 24112

[49] Gibson M and Bruck J 2000 Efficient exact stochastic simulation of chemical systems with many species and many channels *J. Phys. Chem.* A **104** 1876–89

[50] Cao Y, Li H and Petzold L 2004 Efficient formulation of the stochastic simulation algorithm for chemically reacting systems *J. Chem. Phys.* **121** 4059

[51] McCollum J, Peterson G, Cox C, Simpson M and Samatova N 2006 The sorting direct method for stochastic simulation of biochemical systems with varying reaction execution behavior *Comput. Biol. Chem.* **30** 39–49

[52] Schulze T 2008 Efficient kinetic monte carlo simulation *J. Comput. Phys.* **227** 2455–62

[53] Slepoy A, Thompson A and Plimpton S 2008 A constant-time kinetic monte carlo algorithm for simulation of large biochemical reaction networks *J. Chem. Phys.* **128** 205101

[54] Thanh V and Zunino R 2012 Tree-based search for stochastic simulation algorithm *Proc. of ACM-SAC* (ACM) pp 1415–6

[55] Ramaswamy R, Gonzlez-Segredo N and Sbalzarini I 2009 A new class of highly efficient exact stochastic simulation algorithms for chemical reaction networks *J. Chem. Phys.* **130** 244104

[56] Peles S, Munsky B and Khammash M 2006 Reduction and solution of the chemical master equation using time scale separation and finite state projection *J. Chem. Phys.* **125** 204104
[57] Munsky B and Khammash M 2007 A multiple time interval finite state projection algorithm for the solution to the chemical master equation *J. Comput. Phys.* **226** 818–35
[58] Burrage K, Hegland M, MacNamara S and Sidje R 2006 A krylov-based finite state projection algorithm for solving the chemical master equation arising in the discrete modelling of biological systems *Markov Anniversary Meeting: an Int. Conf. to Celebrate the 150th Anniversary of the Birth of A. A. Markov* pp 21–38
[59] Wolf V, Goel R, Mateescu M and Henzinger T 2010 Solving the chemical master equation using sliding windows *BMC Syst. Biol.* **4** 42
[60] Sunkara V and Hegland M 2010 An optimal finite state projection method *Proc. Comput. Sci* **1** 1579–86
[61] Vo H D and Sidje R B 2017 An adaptive solution to the chemical master equation using tensors *J. Chem. Phys.* **147** 044192
[62] Dinh K and Sidje R 2016 Understanding the finite state projection and related methods for solving the chemical master equation *Phys. Biol.* **13** 035003
[63] Sidje R and Vo H 2015 Solving the chemical master equation by a fast adaptive finite state projection based on the stochastic simulation algorithm *Math. Biosci.* **269** 10–16
[64] Sidje R 1998 Expokit: a software package for computing matrix exponentials *ACM Trans. Math. Softw.* **24** 130–56
[65] Moler C and van Loan C 2003 Ninetine dubious ways to compute the exponential of a matrix, twenty-five years later *SIAM Rev.* **45** 1–46
[66] Ugray Z, Lasdon L, Plummer J, Glover F, Kelly J and Marti R 2007 Scatter search and local nlp solvers: a multistart framework for global optimization *INFORMS J. Comput.* **19** 328–40
[67] Glover F 1998 Chapter: A template for scatter search and path relinking *Artificial Evolution* vol 1363 (Berlin: Springer) pp 13–54
[68] Khachaturyan A, Semenovsovskaya S and Vainshtein B 1981 The thermodynamic approach to the structure analysis of crystals *Acta Crystallogr.* A **37** 742–54
[69] Kirkpatrick S, Gelatt C Jr and Vecchi M 1983 Optimization by simulated annealing *Science* **220** 671–80
[70] Cerny V 1985 Thermodynamical approach to the traveling salesman problem: an efficient simulation algorithm *J. Optim. Theory Appl.* **45** 41–51
[71] Semenovskaya S, Khachaturyan K and Khachaturyan A 1985 Statistical mechanics approach to the structure determination of a crystal *Acta Crystallogr.* A **41** 268–73