arXiv:1102.2819v1 [q-bio.QM] 14 Feb 2011

Parameter Identification for Markov Models of Biochemical Reactions

Aleksandr Andreychenko, Linar Mikeev, David Spieler, Verena Wolf

Saarland University, Saarbrücken, Germany

Abstract

We propose a numerical technique for parameter inference in Markov models of biological processes. Based on time-series data of a process we estimate the kinetic rate constants by maximizing the likelihood of the data. The computation of the likelihood relies on a dynamic abstraction of the discrete state space of the Markov model which successfully mitigates the problem of state space largeness. We compare two variants of our method to state-of-the-art, recently published methods and demonstrate their usefulness and efficiency on several case studies from systems biology.

1 Introduction

A widely-used strategy in systems biology research is to refine mathematical models of biological processes based on both computer simulations and wet-lab experiments. In this context, parameter estimation methods for quantitative models play a major role. Typically, time series data is analyzed to learn the structure of a biochemical reaction network and to calibrate the reaction rate parameters. Direct measurement of parameters through wet-lab experiments is often difficult or even impracticable. There are extensive research efforts to estimate the reaction rate parameters of ordinary differential equations (ODEs) that describe the evolution of the chemical concentrations over time (see, for instance, [5, 4, 1] and the references therein). The problem of finding parameters that minimize the difference between observed and predicted data is usually multimodal due to non-linear constraints and thus requires global optimization techniques.

The assumption that chemical concentrations change deterministically and continuously in time is not always appropriate for biological processes. In particular, if certain substances in the cell are present in small concentrations the resulting stochastic effects cannot be adequately described by deterministic models. In that case, discrete-state stochastic models are advantageous because they take into account the discrete random nature of chemical reactions. The theory of stochastic chemical kinetics provides a rigorously justified framework for the description of chemical reactions where the effects of molecular noise are taken into account [6]. It is based on discrete-state Markov processes that explicitly represent the reactions as state-transitions between population vectors. When the molecule numbers are large, the solution of the ODE description of a reaction network and the mean of the corresponding stochastic model agree up to a small approximation error. If, however, small populations are involved, then only a

stochastic description can provide probabilities of events of interest such as probabilities of switching between different expression states in gene regulatory networks or the distribution of gene expression products. Moreover, even the mean behavior of the stochastic model can largely deviate from the behavior of the deterministic model [12]. In such cases the parameters of the stochastic model rather then the parameters of the deterministic model have to be estimated [17, 15, 19].

Here, we consider noisy time series measurements of the system state as they are available from wet-lab experiments. Recent experimental imaging techniques such as high-resolution fluorescence microscopy can measure small molecule counts with measurement errors of less than one molecule [7]. We assume that the structure of the underlying reaction network is known but the rate parameters of the network are unknown. Then we identify those parameters that maximize the likelihood of the time series data. Maximum likelihood estimators are the most popular estimators since they have desirable mathematical properties. Specifically, they become minimum variance unbiased estimators and are asymptotically normal as the sample size increases.

Our main contribution consists in devising an efficient algorithm for the numerical approximation of the likelihood and its derivatives w.r.t. the reaction rate constants. Previous techniques are based on Monte-Carlo sampling [17, 19] because the discrete state space of the underlying model is typically infinite in several dimensions and a priori a reasonable truncation of the state space is not availabe. Our method is not based on sampling but directly calculates the likelihood using a dynamic truncation of the state space. More precisely, we first show that the computation of the likelihood is equivalent to the evaluation of a product of vectors and matrices. This product includes the transition probability matrix of the associated continuous-time Markov process, i.e., the solution of the Kolmogorov differential equations (KDEs). Since solving the KDEs is infeasible, we propose two iterative approximation algorithms during which the state space is truncated in an on-the-fly fashion, that is, during a certain time interval we consider only those states that significantly contribute to the likelihood. One approach exploits equidistant observation intervals while the other approach is particularly well suited for observation intervals that are not equidistant. Both approaches take into account measurement noise during the observations.

After introducing the stochastic model in Section 2, we discuss dynamic state space truncations for the transient probability distribution and its derivatives in Section 3. We introduce the maximum likelihood method in Section 4 and present the approximation methods in Section 5. Finally, we report on experimental results for two reaction networks (Section 6) and discuss related work in Section 7.

2 Discrete-state Stochastic Model

According to Gillespie's theory of stochastic chemical kinetics, a well-stirred mixture of n molecular species in a volume with fixed size and fixed temperature can be represented as a continuous-time Markov chain $\{\mathbf{X}(t), t \geq 0\}$ [6]. The random vector $\mathbf{X}(t) = (X_1(t), \ldots, X_n(t))$ describes the chemical populations at time t, i.e., $X_i(t)$ is the number of molecules of type $i \in \{1, \ldots, n\}$ at time t. Thus, the state space of $\mathbf{X}$ is $\mathbb{Z}_+^n = \{0, 1, \ldots\}^n$. The state changes of $\mathbf{X}$ are triggered by the occurrences of chemical reactions, which are of m different types. For $j \in \{1, \ldots, m\}$ let $\mathbf{v}_j \in \mathbb{Z}^n$ be the nonzero *change vector* of the j-th reaction type, that is, $\mathbf{v}_j = \mathbf{v}_j^- + \mathbf{v}_j^+$ where $\mathbf{v}_j^-$ contains only non-positive entries, which specify how many molecules of each species are consumed (*reactants*) if an instance of the reaction occurs. The vector $\mathbf{v}_j^+$ con-

tains only non-negative entries, which specify how many molecules of each species are produced (*products*). Thus, if $\mathbf{X}(t) = \mathbf{x}$ for some $\mathbf{x} \in \mathbb{Z}_+^n$ with $\mathbf{x} + \mathbf{v}_j^-$ being non-negative, then $\mathbf{X}(t + dt) = \mathbf{x} + \mathbf{v}_j$ is the state of the system after the occurrence of the j-th reaction within the infinitesimal time interval $[t, t + dt)$.

Each reaction type has an associated *propensity function*, denoted by $\alpha_1, \ldots, \alpha_m$, which is such that $\alpha_j(\mathbf{x}) \cdot dt$ is the probability that, given $\mathbf{X}(t) = \mathbf{x}$, one instance of the j-th reaction occurs within $[t, t + dt)$. The value $\alpha_j(\mathbf{x})$ is proportional to the number of distinct reactant combinations in state $\mathbf{x}$. More precisely, if $\mathbf{x} = (x_1, \ldots, x_n)$ is a state for which $\mathbf{x} + \mathbf{v}_j^-$ is nonnegative then, for reactions with at most two reactants,

$$\alpha_j(\mathbf{x}) = \begin{cases} c_j & \text{if } \mathbf{v}_j^- = (0, \ldots, 0), \\ c_j \cdot x_i & \text{if } \mathbf{v}_j^- = -\mathbf{e}_i, \\ c_j \cdot x_i \cdot x_\ell & \text{if } \mathbf{v}_j^- = -\mathbf{e}_i - \mathbf{e}_\ell, \\ c_j \cdot \binom{x_i}{2} = c_j \cdot \frac{x_i \cdot (x_i - 1)}{2} & \text{if } \mathbf{v}_j^- = -2 \cdot \mathbf{e}_i, \end{cases} \tag{1}$$

where $i \neq \ell$, $c_j > 0$, and $\mathbf{e}_i$ is the vector with the i-th entry 1 and all other entries 0.

Example 1 *We consider the simple gene expression model described in [15] that involves three chemical species, namely DNA$_{ON}$, DNA$_{OFF}$, and mRNA, which are represented by the random variables $X_1(t)$, $X_2(t)$, and $X_3(t)$, respectively. The three possible reactions are DNA$_{ON}$ $\rightarrow$ DNA$_{OFF}$, DNA$_{OFF}$ $\rightarrow$ DNA$_{ON}$, and DNA$_{ON}$ $\rightarrow$ DNA$_{ON}$+ mRNA. Thus, $\mathbf{v}_1^- = (-1, 0, 0)$, $\mathbf{v}_1^+ = (0, 1, 0)$, $\mathbf{v}_2^- = (0, -1, 0)$, $\mathbf{v}_2^+ = (1, 0, 0)$, $\mathbf{v}_3^- = (-1, 0, 0)$ and $\mathbf{v}_3^+ = (1, 0, 1)$. For a state $\mathbf{x} = (x_1, x_2, x_3)$, the propensity functions are $\alpha_1(\mathbf{x}) = c_1 \cdot x_1$, $\alpha_2(\mathbf{x}) = c_2 \cdot x_2$, and $\alpha_3(\mathbf{x}) = c_3 \cdot x_1$. Note that given the initial state $\mathbf{x} = (1, 0, 0)$, at any time, either the DNA is active or not, i.e. $x_1 = 0$ and $x_2 = 1$, or $x_1 = 1$ and $x_2 = 0$. Moreover, the state space of the model is infinite in the third dimension. For a fixed time instant $t > 0$, no upper bound on the number of mRNA is known a priori. All states $\mathbf{x}$ with $x_3 \in \mathbb{Z}_+$ have positive probability if $t > 0$ but these probabilities will tend to zero as $x_3 \to \infty$.*

In general, the reaction rate constants c_j refer to the probability that a randomly selected pair of reactants collides and undergoes the j-th chemical reaction. It depends on the volume and the temperature of the system as well as on the microphysical properties of the reactant species. Since reactions of higher order (requiring more than two reactants) are usually the result of several successive lower order reactions, we do not consider the case of more than two reactants.

The Chemical Master Equation. For $\mathbf{x} \in \mathbb{Z}_+^n$ and $t \geq 0$, let $p(\mathbf{x}, t)$ denote the probability $Pr(\mathbf{X}(t) = \mathbf{x})$ and let $\mathbf{p}(t)$ be the row vector with entries $p(\mathbf{x}, t)$.

Given $\mathbf{v}_1^-, \ldots, \mathbf{v}_m^-$, $\mathbf{v}_1^+, \ldots, \mathbf{v}_m^+$, $\alpha_1, \ldots, \alpha_m$, and some initial distribution $\mathbf{p}(0)$, the Markov chain $\mathbf{X}$ is uniquely specified and its evolution is given by the chemical master equation (CME)

$$\tfrac{d}{dt}\mathbf{p}(t) = \mathbf{p}(t)Q, \tag{2}$$

where Q is the infinitesimal generator matrix of $\mathbf{X}$ with $Q(\mathbf{x}, \mathbf{y}) = \alpha_j(\mathbf{x})$ if $\mathbf{y} = \mathbf{x} + \mathbf{v}_j$ and $\mathbf{x} + \mathbf{v}_j^- \geq 0$. Note that, in order to simplify our presentation, we assume here that all vectors $\mathbf{v}_j$ are distinct. All remaining entries of Q are zero except for the diagonal entries which are equal to the negative row sum. The ordinary first-order differential equation in (2) is a direct consequence of the Kolmogorov forward equation. Since $\mathbf{X}$ is a regular Markov process, (2) has the general solution $\mathbf{p}(t) = \mathbf{p}(0) \cdot e^{Qt}$, where e^A is the matrix exponential of a matrix A. If the state space of X is infinite, then we can only compute approximations of $\mathbf{p}(t)$. But even if Q is finite, its size is often large because it

grows exponentially with the number of state variables. Therefore standard numerical solution techniques for systems of first-order linear equations of the form of (2) are infeasible. The reason is that the number of nonzero entries in Q often exceeds the available memory capacity for systems of realistic size. If the populations of all species remain small (at most a few hundreds) then the CME can be efficiently approximated using projection methods [9, 14, 3] or fast uniformization methods [13, 16]. The idea of these methods is to avoid an exhaustive state space exploration and, depending on a certain time interval, restrict the analysis of the system to a subset of states.

Here, we are interested in the partial derivatives of $\mathbf{p}(t)$ w.r.t. the reaction rate constants $\mathbf{c} = (c_1, \ldots, c_m)$. In order to explicitly indicate the dependence of $\mathbf{p}(t)$ on the vector $\mathbf{c}$ we write $\mathbf{p}(\mathbf{c}, t)$ instead of $\mathbf{p}(t)$ and $p(\mathbf{x}, \mathbf{c}, t)$ instead of $p(\mathbf{x}, t)$ if necessary. We define the row vectors $\mathbf{s}_j(\mathbf{c}, t)$ as the derivative of $\mathbf{p}(\mathbf{c}, t)$ w.r.t. c_j, i.e.,

$$\mathbf{s}_j(\mathbf{c}, t) = \frac{\partial \mathbf{p}(\mathbf{c},t)}{\partial c_j} = \lim_{\Delta c \to 0} \frac{\mathbf{p}(\mathbf{c}+\boldsymbol{\Delta}\mathbf{c}_j,t)-\mathbf{p}(\mathbf{c},t)}{\Delta c},$$

where the vector $\boldsymbol{\Delta}\mathbf{c}_j$ is zero everywhere except for the j-th position that is equal to Δc. We denote the entry in $\mathbf{s}_j(\mathbf{c}, t)$ that corresponds to state $\mathbf{x}$ by $s_j(\mathbf{x}, \mathbf{c}, t)$. Using (2), we find that $\mathbf{s}_j(\mathbf{c}, t)$ is the unique solution of the system of ODEs

$$\frac{d}{dt}\mathbf{s}_j(\mathbf{c}, t) = \mathbf{s}_j(\mathbf{c}, t)Q + \mathbf{p}(\mathbf{c}, t)\frac{\partial}{\partial c_j}Q, \tag{3}$$

where $j \in \{1, \ldots, m\}$. The initial condition is $s_j(\mathbf{x}, \mathbf{c}, 0) = 0$ for all $\mathbf{x}$ and $\mathbf{c}$ since $p(\mathbf{x}, \mathbf{c}, 0)$ is independent of c_j.

3 Dynamic state space truncation

The parameter estimation method that we propose in Section 5.1 builds on the approximation of the transient distribution $\mathbf{p}(t)$ and the derivatives $\mathbf{s}_j(\mathbf{c}, t)$ for all j at a fixed time instant $t > 0$. Therefore we now discuss how to solve (2) and (3) simultaneously using an explicit fourth-order Runge-Kutta method and a dynamically truncated state space. This truncation is necessary because models of chemical reaction networks typically have a very large or infinite number of states $\mathbf{x}$ with nonzero values for $p(\mathbf{x}, t)$ and $s_j(\mathbf{x}, \mathbf{c}, t)$. For instance, the system in Example 1 is infinite in one dimension. In order to keep the number of states, that are considered in a certain step of the numerical integration, manageable we suggest a dynamic truncation of the state space that, for a given time interval, neglects those states being not relevant during that time, that is, we neglect states that have a probability that is smaller than a certain threshold.

First, we remark that the equation that corresponds to state $\mathbf{x}$ in (2) is given by

$$\frac{d}{dt}p(\mathbf{x}, t) = \sum_{j:\mathbf{x}-\mathbf{v}_j^- \geq 0} \alpha_j(\mathbf{x} - \mathbf{v}_j)p(\mathbf{x} - \mathbf{v}_j, t) - \alpha_j(\mathbf{x})p(\mathbf{x}, t). \tag{4}$$

and it describes the change of the probability of state $\mathbf{x}$ as the difference between inflow of probability at rate $\alpha_j(\mathbf{x} - \mathbf{v}_j)$ from direct predecessors $\mathbf{x} - \mathbf{v}_j$ and outflow of probability at rate $\alpha_j(\mathbf{x})$. Assume now that an initial distribution $\mathbf{p}(0)$ is given. We choose a small positive constant δ and define the set of significant states $S = \{\mathbf{x} \mid p(\mathbf{x}, 0) > \delta\}$. We then only integrate equations in (2) and (3) that belong to states in S. If h is the time step of the numerical integration, then for the interval $[t, t + h)$ we use the following strategy to modify S according to the probability flow. We check for all successors $\mathbf{x} + \mathbf{v}_j \notin S$ of a state $\mathbf{x} \in S$ whether $p(\mathbf{x} + \mathbf{v}_j, t + h)$ becomes greater than δ at time $t + h$ as they receive "inflow" from their direct predecessors

(see Eq. (4)). If the probability that $\mathbf{x} + \mathbf{v}_j$ receives is greater δ, then we add $\mathbf{x} + \mathbf{v}_j$ to S. Note that since we use a fourth-order method, states reachable within at most four transitions from a state in S can be added during one step of the integration. On the other hand, whenever $p(\mathbf{x}, t)$ becomes less or equal to δ for a state $\mathbf{x} \in S$ then we remove $\mathbf{x}$ from S. We approximate the probabilities and derivatives of all states that are not considered during $[t, t + h)$ with zero. In this way the computational costs of the numerical integration is drastically reduced, since typically the number of states with probabilities less than δ is large and the main part of the probability mass is concentrated on a small number of significant states. Due to the regular structure of $\mathbf{X}$, the probability of a state decreases exponentially with its distance to the "high probability" locations. If δ is small (e.g. 10^{-15}) and the initial distribution is such that the main part of the probability mass (e.g. 99.99%) distributes on a manageable number of states, then even for long time horizons the approximation of the transient distribution is accurate. For arbitrary Markov models, the approximation error of the derivatives could, in principle, be large. For biochemical reaction networks, however, the underlying Markov process is well-structured and the sensitivity of the transient distribution w.r.t. the rate constants is comparatively small, i.e., small changes of the rate constants result in a transient distribution that differs only slightly from the original distribution. Therefore, the derivatives of insignificant states are small and, in order to calibrate parameters, it is sufficient to consider the derivatives of probabilities of significant states. It is impossible to explore the whole state space and those parts containing most of the probability mass are most informative w.r.t. pertubations of the rate constants.

Example 2 *We consider a simple enzyme reaction with three reactions that involve four different species, namely enzymes (E), substrates (S), complex molecules (C), and proteins (P). The reactions are complex formation (E+S→ C), dissociation of the complex (C→E+S), and protein production (C→E+P). The corresponding rate functions are* $\alpha_1(\mathbf{x}) = c_1 \cdot x_1 \cdot x_2$, $\alpha_2(\mathbf{x}) = c_2 \cdot x_3$, *and* $\alpha_3(\mathbf{x}) = c_3 \cdot x_3$ *where* $\mathbf{x} = (x_1, x_2, x_3, x_4)$. *The change vectors are given by* $\mathbf{v}_1^- = (-1, -1, 0, 0)$, $\mathbf{v}_1^+ = (0, 0, 1, 0)$, $\mathbf{v}_2^- = (0, 0, -1, 0)$, $\mathbf{v}_2^+ = (1, 1, 0, 0)$, $\mathbf{v}_3^- = (0, 0, -1, 0)$, *and* $\mathbf{v}_3^+ = (1, 0, 0, 1)$. *We start initially with probability one in state* $\mathbf{x} = (1000, 200, 0, 0)$ *and compute* $\mathbf{p}(t)$ *and* $\mathbf{s}_j(\mathbf{c}, t)$ *for* $t = 10$, $\mathbf{c} = (1, 1, 0.1)$, *and* $j \in \{1, 2, 3\}$. *In Table 1 we list the results of the approximation of* $\mathbf{p}(t)$ *and* $\mathbf{s}_j(\mathbf{c}, t)$. *We chose this model because it has a finite state space and we can compare our approximation with the values obtained for* $\delta = 0$. *The column "Time" lists the running times of the computation. Obviously, the smaller* δ *the more time consuming is the computation. The remaining columns refer to the maximum absolute error of all entries in the vectors* $\mathbf{p}(t)$ *and* $\mathbf{s}_j(\mathbf{c}, t)$ *where we use as exact values those obtained by setting* $\delta = 0$. *Clearly, even if* $\delta = 0$ *we have an approximation error due to the numerical integration of* (2) *and* (3), *which is, however, very small compared to the error that originates from the truncation of the state space.*

A similar truncation effect can be obtained by sorting the entries of $\mathbf{p}(t)$ and successively removing the smallest entries until a fixed amount ε of probability mass is lost. If ε is chosen proportional to the time step, then it is possible to bound the total approximation error of the probabilities, i.e., $\varepsilon = \tilde{\varepsilon} h/t$ where $\tilde{\varepsilon}$ is the total approximation error for a time horizon of length t. If memory requirements and running time are more pressing then accuracy, then we can adjust the computational costs of the approximation by keeping only the k most probable states in each step for some integer k.

Table 1: Approximated transient distribution and derivatives of the enzyme reaction network.

δ	Time	Maximum absolute error			
		$\mathbf{p}(t)$	$\mathbf{s}_1(\mathbf{c},t)$	$\mathbf{s}_2(\mathbf{c},t)$	$\mathbf{s}_3(\mathbf{c},t)$
0	10 h	0	0	0	0
10^{-20}	47 sec	$1 \cdot 10^{-11}$	$1 \cdot 10^{-12}$	$1 \cdot 10^{-12}$	$4 \cdot 10^{-9}$
10^{-15}	25 sec	$1 \cdot 10^{-11}$	$8 \cdot 10^{-11}$	$9 \cdot 10^{-11}$	$2 \cdot 10^{-8}$
10^{-10}	10 sec	$7 \cdot 10^{-7}$	$3 \cdot 10^{-6}$	$4 \cdot 10^{-6}$	$2 \cdot 10^{-4}$

4 Parameter Inference

Following the notation in [15], we assume that observations of a biochemical network are made at time instances $t_1, \ldots, t_R \in \mathbb{R}_{\geq 0}$ where $t_1 < \ldots < t_R$. Moreover, we assume that $O_i(t_\ell)$ is the observed number of species i at time t_ℓ for $i \in \{1, \ldots, n\}$ and $\ell \in \{1, \ldots, R\}$. Let $\mathbf{O}(t_\ell) = (O_1(t_\ell), \ldots, O_n(t_\ell))$ be the corresponding vector of observations. Since these observations are typically subject to measurement errors, we assume that $O_i(t_\ell) = X_i(t_\ell) + \epsilon_i(t_\ell)$ where the error terms $\epsilon_i(t_\ell)$ are independent and identically normally distributed with mean zero and standard deviation σ. Note that $X_i(t_\ell)$ is the true population of the i-th species at time t_ℓ. Clearly, this implies that, conditional on $X_i(t_\ell)$, the random variable $O_i(t_\ell)$ is independent of all other observations as well as independent of the history of $\mathbf{X}$ before time t_ℓ.

We assume further that for the unobserved process $\mathbf{X}$ we do not know the values of the rate constants $c_1, \ldots, c_m$ and our aim is to estimate these constants. Similarly, the exact standard deviation σ of the error terms is unknown and must be estimated[1]. Let f denote the joint density of $\mathbf{O}(t_1), \ldots, \mathbf{O}(t_R)$. Then the likelihood of the observations is [11]

$$\begin{aligned} \mathcal{L} &= f\left(\mathbf{O}(t_1), \ldots, \mathbf{O}(t_R)\right) \\ &= \textstyle\sum_{\mathbf{x}_1} \ldots \sum_{\mathbf{x}_R} f\left(\mathbf{O}(t_1), \ldots, \mathbf{O}(t_R) \mid \mathbf{X}(t_1) = \mathbf{x}_1, \ldots, \mathbf{X}(t_R) = \mathbf{x}_R\right) \\ &\quad Pr(\mathbf{X}(t_1) = \mathbf{x}_1, \ldots, \mathbf{X}(t_R) = \mathbf{x}_R)\,, \end{aligned} \tag{5}$$

that is, $\mathcal{L}$ is the probability to observe $\mathbf{O}(t_1), \ldots, \mathbf{O}(t_R)$. Note that $\mathcal{L}$ depends on the chosen rate parameters $\mathbf{c}$ since the probability measure $Pr(\cdot)$ does. Furthermore, $\mathcal{L}$ depends on σ since the density f does. When necessary, we will make this dependence explicit by writing $\mathcal{L}(\mathbf{c}, \sigma)$ instead of $\mathcal{L}$. We now seek constants $\mathbf{c}^*$ and a standard deviation σ^* such that

$$\mathcal{L}(\mathbf{c}^*, \sigma^*) = \max_{\sigma, \mathbf{c}} \mathcal{L}(\mathbf{c}, \sigma) \tag{6}$$

where the maximum is taken over all $\sigma > 0$ and vectors $\mathbf{c}$ with all components strictly positive. This optimization problem is known as the maximum likelihood problem [11]. Note that $\mathbf{c}^*$ and σ^* are random variables because they depend on the (random) observations $\mathbf{O}(t_1), \ldots, \mathbf{O}(t_R)$.

If more than one sequence of observations is made, then the corresponding likelihood is the product of the likelihoods of all individual sequences. More precisely, if $\mathbf{O}^k(t_l)$ is the k-th observation that has been observed at time instant t_l where $k \in$

[1] We remark that it is straightforward to extend the estimation framework that we present in the sequel such that a covariance matrix for a multivariate normal distribution of the error terms is estimated. In this way, different measurement errors of the species can be taken into account as well as dependencies between error terms.

$\{1, \ldots, K\}$, then we define $\mathcal{L}_k(\mathbf{c}, \sigma)$ as the probability to observe $\mathbf{O}^k(t_1), \ldots, \mathbf{O}^k(t_R)$ and maximize

$$\prod_{k=1}^{K} \mathcal{L}_k(\mathbf{c}, \sigma). \tag{7}$$

In the sequel, we concentrate on expressions for $\mathcal{L}_k(\mathbf{c}, \sigma)$ and $\frac{\partial}{\partial c_j}\mathcal{L}_k(\mathbf{c}, \sigma)$. We first assume $K = 1$ and drop index k. We consider the case $K > 1$ later. In (5) we sum over all state sequences $\mathbf{x}_1, \ldots, \mathbf{x}_R$ such that $Pr(\mathbf{X}(t_\ell) = \mathbf{x}_\ell, 1 \leq \ell \leq R) > 0$. Since $\mathbf{X}$ has a large or even infinite state space, it is computationally infeasible to explore all possible sequences. In Section 5 we propose an algorithm to approximate the likelihoods and their derivatives. We truncate the state space in a similar way as in Section 3 and use the fact that (5) can be written as a product of vectors and matrices. Let ϕ_σ be the density of the normal distribution with mean zero and standard deviation σ. Then

$$\begin{aligned} & f\left(\mathbf{O}(t_1), \ldots, \mathbf{O}(t_R) \mid \mathbf{X}(t_1) = \mathbf{x}_1, \ldots, \mathbf{X}(t_R) = \mathbf{x}_R\right) \\ = \; & \prod_{\ell=1}^{R} \prod_{i=1}^{n} f\left(O_i(t_\ell) \mid X_i(t_\ell) = x_{i\ell}\right) \\ = \; & \prod_{\ell=1}^{R} \prod_{i=1}^{n} \phi_\sigma(O_i(t_\ell) - x_{i\ell}), \end{aligned}$$

where $\mathbf{x}_\ell = (x_{1\ell}, \ldots, x_{n\ell})$. If we write $w(\mathbf{x}_\ell)$ for $\prod_{i=1}^{n} \phi_\sigma(O_i(t_\ell) - x_{i\ell})$, then the sequence $\mathbf{x}_1, \ldots, \mathbf{x}_R$ has weight $\prod_{\ell=1}^{R} w(\mathbf{x}_\ell)$ and, thus,

$$\mathcal{L} = \sum_{\mathbf{x}_1} \cdots \sum_{\mathbf{x}_R} Pr(\mathbf{X}(t_1) = \mathbf{x}_1, \ldots, \mathbf{X}(t_R) = \mathbf{x}_R) \prod_{\ell=1}^{R} w(\mathbf{x}_\ell). \tag{8}$$

Moreover, for the probability of the sequence $\mathbf{x}_1, \ldots, \mathbf{x}_R$ we have

$$Pr(\mathbf{X}(t_1) = \mathbf{x}_1, \ldots, \mathbf{X}(t_R) = \mathbf{x}_R) = p(\mathbf{x}_1, t_1) P_2(\mathbf{x}_1, \mathbf{x}_2) \cdots P_R(\mathbf{x}_{R-1}, \mathbf{x}_R)$$

where $P_\ell(\mathbf{x}, \mathbf{y}) = Pr(\mathbf{X}(t_\ell) = \mathbf{y} \mid \mathbf{X}(t_{\ell-1}) = \mathbf{x})$. Hence, (8) can be written as

$$\mathcal{L} = \sum_{\mathbf{x}_1} p(\mathbf{x}_1, t_1) w(\mathbf{x}_1) \sum_{\mathbf{x}_2} P_2(\mathbf{x}_1, \mathbf{x}_2) w(\mathbf{x}_2) \ldots \sum_{\mathbf{x}_R} P_R(\mathbf{x}_{R-1}, \mathbf{x}_R) w(\mathbf{x}_R). \tag{9}$$

Let P_ℓ be the matrix with entries $P_\ell(\mathbf{x}, \mathbf{y})$ for all states $\mathbf{x}, \mathbf{y}$. Note that P_ℓ is the transition probability matrix of $\mathbf{X}$ for time step $t_\ell - t_{\ell-1}$ and thus the general solution $e^{Q(t_\ell - t_{\ell-1})}$ of the Kolmogorov forward and backward differential equations

$$\tfrac{d}{dt} P_\ell = Q P_\ell, \qquad\qquad \tfrac{d}{dt} P_\ell = P_\ell Q.$$

Using $\mathbf{p}(t_1) = \mathbf{p}(t_0) P_1$ with $t_0 = 0$, we can write (9) in matrix-vector form as

$$\mathcal{L} = \mathbf{p}(t_0) P_1 W_1 P_2 W_2 \cdots P_R W_R \mathbf{e}. \tag{10}$$

Here, $\mathbf{e}$ is the vector with all entries equal to one and W_ℓ is a diagonal matrix whose diagonal entries are all equal to $w(\mathbf{x}_\ell)$ with $\ell \in \{1, \ldots, R\}$, where W_ℓ is of the same size as P_ℓ. Since it is in general not possible to analytically obtain parameters that maximize $\mathcal{L}$, we use optimization techniques to find $\mathbf{c}^*$ and σ^*. Typically, such techniques iterate over values of $\mathbf{c}$ and σ and increase the likelihood $\mathcal{L}(\mathbf{c}, \sigma)$ by following the gradient. Therefore, we need to calculate the derivatives $\frac{\partial}{\partial c_j}\mathcal{L}$ and $\frac{\partial}{\partial \sigma}\mathcal{L}$. For $\frac{\partial}{\partial c_j}\mathcal{L}$ we obtain

$$\begin{aligned} \tfrac{\partial}{\partial c_j}\mathcal{L} &= \tfrac{\partial}{\partial c_j}\left(\mathbf{p}(t_0) P_1 W_1 P_2 W_2 \cdots P_R W_R \mathbf{e}\right) \\ &= \mathbf{p}(t_0) \left(\textstyle\sum_{\ell=1}^{R} \left(\tfrac{\partial}{\partial c_j} P_\ell \right) W_\ell \prod_{\ell' \neq \ell} P_{\ell'} W_{\ell'} \right) \mathbf{e}. \end{aligned} \tag{11}$$

The derivative of $\mathcal{L}$ w.r.t. the standard deviation σ is derived analogously. The only difference is that $P_1, \ldots, P_R$ are independent of σ but $W_1, \ldots, W_R$ depend on σ. It is also important to note that expressions for partial derivatives of second order can be derived in a similar way. These derivatives can then be used for an efficient gradient-based local optimization.

For $K > 1$ observation sequences we can maximize the log-likelihood

$$\log \prod_{k=1}^{K} \mathcal{L}_k = \sum_{k=1}^{K} \log \mathcal{L}_k, \tag{12}$$

instead of the likelihood in (7), where we abbreviate $\mathcal{L}_k(\mathbf{c}, \sigma)$ by $\mathcal{L}_k$. Note that the derivatives are then given by

$$\frac{\partial}{\partial \lambda} \sum_{k=1}^{K} \log \mathcal{L}_k = \sum_{k=1}^{K} \frac{\frac{\partial}{\partial \lambda} \mathcal{L}_k}{\mathcal{L}_k}, \tag{13}$$

where λ is either c_j or σ. It is also important to note that only the weights $w(\mathbf{x}_\ell)$ depend on k, that is, on the observed sequence $\mathbf{O}^k(t_1), \ldots, \mathbf{O}^k(t_R)$. Thus, when we compute $\mathcal{L}_k$ based on (10) we use for all k the same transition matrices $P_1, \ldots, P_R$ and the same initial conditions $\mathbf{p}(t_0)$, but possibly different matrices $W_1, \ldots, W_R$.

5 Numerical approximation algorithm

In this section, we focus on the numerical approximation of the likelihood and the corresponding derivatives w.r.t. the rate constants $c_1, \ldots, c_m$. We propose two approximation algorithms for the likelihood and its derivatives, a state-based likelihood approximation (SLA) and a path-based likelihood approximation (PLA). Both are based on a dynamic truncation of the state space as suggested in Section 3. They differ in that the PLA method exploits equidistant time series, that is, it is particularly efficient if $h = t_{\ell+1} - t_\ell$ for all ℓ and if σ is not too large. The SLA algorithm works for arbitrarily spaced time series and is efficient even if σ is large.

5.1 State-based likelihood approximation

The SLA algorithm calculates an approximation of the likelihood based on (10) by traversing the matrix-vector product from the left to the right. The main idea behind the algorithm is that instead of explicitly computing the matrices P_ℓ, we express the vector-matrix product $\mathbf{u}(t_{\ell-1})P_\ell$ as a system of ODEs similar to the CME (cf. Eq. (2)). Here, $\mathbf{u}(t_0), \ldots, \mathbf{u}(t_R)$ are row vectors obtained during the iteration over time points $t_0, \ldots, t_R$, that is, we define $\mathcal{L}$ recursively as $\mathcal{L} = \mathbf{u}(t_R)\mathbf{e}$ with $\mathbf{u}(t_0) = \mathbf{p}(t_0)$ and

$$\mathbf{u}(t_\ell) \quad = \quad \mathbf{u}(t_{\ell-1})P_\ell W_\ell \qquad \text{for all } 1 \leq \ell \leq R,$$

where $t_0 = 0$. Instead of computing P_ℓ explicitly, we solve R systems of ODEs

$$\frac{d}{dt}\tilde{\mathbf{u}}(t) = \tilde{\mathbf{u}}(t)Q \tag{14}$$

with initial condition $\tilde{\mathbf{u}}(t_{\ell-1}) = \mathbf{u}(t_{\ell-1})$ for the time interval $[t_{\ell-1}, t_\ell)$ where $\ell \in \{1, \ldots, R\}$. After solving the ℓ-th system of ODEs we set $\mathbf{u}(t_\ell) = \tilde{\mathbf{u}}(t_\ell)W_\ell$ and finally compute $\mathcal{L} = \mathbf{u}(t_R)\mathbf{e}$. Since this is the same as solving the CME for different initial conditions, we can use the dynamic truncation of the state space proposed in Section 3. Since the vectors $\tilde{\mathbf{u}}(t_\ell)$ do not sum up to one, we scale all entries by multiplication with $1/(\tilde{\mathbf{u}}(t_\ell)\mathbf{e})$. This simplifies the truncation of the state space using the significance

threshold δ since after scaling it can be interpreted as a probability. In order to obtain the correct (unscaled) likelihood, we compute $\mathcal{L}$ as $\mathcal{L} = \prod_{\ell=1}^{R}(\tilde{\mathbf{u}}(t_\ell)\mathbf{e})$. We handle the derivatives of $\mathcal{L}$ in a similar way. To shorten our presentation, we only consider the derivative $\frac{\partial}{\partial c_j}\mathcal{L}$ in the sequel. An iterative scheme for $\frac{\partial}{\partial \sigma}\mathcal{L}$ is derived analogously. From (11) we obtain $\frac{\partial}{\partial c_j}\mathcal{L} = \mathbf{u}_j(t_R)\mathbf{e}$ with $\mathbf{u}_j(t_0) = \mathbf{0}$ and

$$\mathbf{u}_j(t_\ell) \quad = \quad (\mathbf{u}_j(t_{\ell-1})P_\ell + \mathbf{u}(t_{\ell-1})\tfrac{\partial}{\partial c_j}P_\ell)W_\ell \qquad \text{for all } 1 \leq \ell \leq R,$$

where $\mathbf{0}$ is the vector with all entries zero. Thus, during the solution of the ℓ-th ODE in (14) we simultaneously solve

$$\tfrac{d}{dt}\tilde{\mathbf{u}}_j(t) = \tilde{\mathbf{u}}_j(t)Q + \tilde{\mathbf{u}}(t)\tfrac{\partial}{\partial c_j}Q \tag{15}$$

with initial condition $\tilde{\mathbf{u}}_j(t_{\ell-1}) = \mathbf{u}_j(t_{\ell-1})$ for the time interval $[t_{\ell-1}, t_\ell)$. As above, we set $\mathbf{u}_j(t_\ell) = \tilde{\mathbf{u}}_j(t_\ell)W_\ell$ and obtain $\frac{\partial}{\partial c_j}\mathcal{L}$ as $\mathbf{u}_j(t_R)\mathbf{e}$.

Solving (14) and (15) simultaneously is equivalent to the computation of the partial derivatives in (3) with different initial conditions. Thus, we can use the approximation algorithm proposed in Section 3 to approximate $\mathbf{u}_j(t_\ell)$. Experimental results of the finite enzyme reaction network (see Example 2) show that the approximation errors of the likelihood and its derivatives are of the same order of magnitude as those of the transient probabilities and their derivatives (not shown). Note, however, that, if σ is small only few states contribute significantly to the likelihood. In this case, truncation strategies based on sorting of vectors are more efficient without considerable accuracy losses since the main part of the likelihood concentrates on very few entries (namely those that correspond to states that are close to the observed populations).

In the case of K observation sequences we repeat the above algorithm in order to sequentially compute $\mathcal{L}_k$ for $k \in \{1, \ldots, K\}$. We exploit (12) and (13) to compute the total log-likelihood and its derivatives as a sum of individual terms. Obviously, it is possible to parallelize the SLA algorithm by computing $\mathcal{L}_k$ in parallel for all k.

5.2 Path-based likelihood approximation

If $\Delta t = t_\ell - t_{\ell-1}$ for all ℓ then the matrices $P_1, \ldots, P_R$ in (10) are equal to the Δt-step transition matrix $T(\Delta t)$ with entries $Pr(\mathbf{X}(t + \Delta t) = \mathbf{y} \mid \mathbf{X}(t) = \mathbf{x})$. Note that since we consider a time-homogeneous Markov process $\mathbf{X}$, the matrix $T(\Delta t)$ is independent of t. The main idea of the PLA method is to iteratively compute those parts of $T(\Delta t)$ that correspond to state sequences (paths) $\mathbf{x}_1, \ldots, \mathbf{x}_R$ that contribute significantly to $\mathcal{L}$. The algorithm can be summarized as follows, where we omit the argument Δt of T to improve the readability and refer to the entries of T as $T(\mathbf{x}, \mathbf{y})$:

1. We compute the transient distribution $\mathbf{p}(t_1)$ and its derivatives (w.r.t. $\mathbf{c}$ and σ) as outlined in Section 3 using a significance threshold δ.

2. For each state $\mathbf{x}_1$ with significant probability $p(\mathbf{x}_1, t_1)$ we approximate the rows of T and $\frac{\partial}{\partial c_j}T$ that correspond to $\mathbf{x}_1$ based on a transient analysis for Δt time units. More precisely, if $\mathbf{e}_{\mathbf{x}_1}$ is the vector with all entries zero except for the entry that corresponds to state $\mathbf{x}_1$ which is one, then we solve (2) with initial condition $\mathbf{e}_{\mathbf{x}_1}$ for Δt time units in order to approximate $T(\mathbf{x}_1, \mathbf{x}_2)$ and $\frac{\partial}{\partial c_j}T(\mathbf{x}_1, \mathbf{x}_2)$ for all $\mathbf{x}_2$. During this transient analysis we again apply the dynamic truncation of the state space proposed in Section 3 with threshold δ.

3. We then store for each pair $(\mathbf{x}_1, \mathbf{x}_2)$ the (partial) likelihood $a(\mathbf{x}_1, \mathbf{x}_2)$ and its derivatives:

$$\begin{aligned} a(\mathbf{x}_1, \mathbf{x}_2) &= p(\mathbf{x}_1, t_1) \cdot w(\mathbf{x}_1) \cdot T(\mathbf{x}_1, \mathbf{x}_2) \cdot w(\mathbf{x}_2) \\ \tfrac{\partial}{\partial c_j} a(\mathbf{x}_1, \mathbf{x}_2) &= \tfrac{\partial}{\partial c_j} p(\mathbf{x}_1, t_1) \cdot w(\mathbf{x}_1) \cdot T(\mathbf{x}_1, \mathbf{x}_2) \cdot w(\mathbf{x}_2) \\ &\quad + p(\mathbf{x}_1, t_1) \cdot w(\mathbf{x}_1) \cdot \tfrac{\partial}{\partial c_j} T(\mathbf{x}_1, \mathbf{x}_2) \cdot w(\mathbf{x}_2). \end{aligned}$$

4. We reduce the number of considered pairs by sorting $a(\mathbf{x}_1, \mathbf{x}_2)$ for all pairs $(\mathbf{x}_1, \mathbf{x}_2)$ calculated in the previous step and keep the most probable pairs (see also Section 3).

5. Next, we repeat steps 2-4, where in step 2 we start the analysis from all states $\mathbf{x}_2$ that are the last element of a pair kept in the previous step. In step 3 we store triples of states, say, $(\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3)$ and recursively compute their likelihood and the corresponding derivatives by multiplication with $T(\mathbf{x}_2, \mathbf{x}_3)$ and $w(\mathbf{x}_3)$, i.e., for the likelihood we compute

$$\begin{aligned} a(\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3) &= a(\mathbf{x}_1, \mathbf{x}_2) \cdot T(\mathbf{x}_2, \mathbf{x}_3) \cdot w(\mathbf{x}_3) \\ \tfrac{\partial}{\partial c_j} a(\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3) &= \tfrac{\partial}{\partial c_j} a(\mathbf{x}_1, \mathbf{x}_2) \cdot T(\mathbf{x}_2, \mathbf{x}_3) \cdot w(\mathbf{x}_3) \\ &\quad + a(\mathbf{x}_1, \mathbf{x}_2) \cdot \tfrac{\partial}{\partial c_j} T(\mathbf{x}_2, \mathbf{x}_3) \cdot w(\mathbf{x}_3). \end{aligned}$$

   Note that we may reuse some of the entries of T since they already have been calculated in a previous step. In step 4 we again reduce the number of triples $(\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3)$ by sorting them according to their likelihood. We then keep the most probable triples, and so on. Note that in step 4 we cannot use a fixed truncation threshold δ to reduce the number of state sequences (or paths) since their probabilities may become very small as the sequences become longer.

6. We stop the prolongation of paths $\mathbf{x}_1, \ldots, \mathbf{x}_\ell$ when the time instance $t_R = \Delta t \cdot R$ is reached and compute an approximation of $\mathcal{L}$ and its derivatives by summing up the corresponding values of all paths (cf. Eq. (8)).

If we have more than one observation sequence, i.e., $K > 1$, then we repeat the procedure to compute $\mathcal{L}_k$ for all k and use (12) to calculate the total log-likelihood. Note that the contribution of each path $\mathbf{x}_1, \ldots, \mathbf{x}_R$ to $\mathcal{L}_k$ may be different for each k. It is, however, likely that the entries of T can be reused not only during the computation of each single $\mathcal{L}_k$ but also for different values of k. If many entries of T are reused during the computation, the algorithm performs fast compared to other approaches. For our experimental results in Section 6, we keep the ten most probable paths in step 4. Even though this enforces a coarse approximation, the likelihood is approximated very accurately if σ is small, since in this case only few paths contribute significantly to $\mathcal{L}_k$. On the other hand, if σ is large, then the approximation may become inaccurate depending on the chosen truncation strategy. Another disadvantage of the PLA method is that for non-equidistant time series, the performance is slow since we have to compute (parts of) different transition matrices and, during the computation of $\mathcal{L}_k$, the transition probabilities cannot be reused.

6 Experimental results

In this section we present experimental results of the SLA and PLA method. For equidistant time series, we compare our approach to the approximate maximum like-

lihood (AML) and the singular value decomposition (SVDL) method described by Reinker et al. [15] (compare also Section 7). Since an implementation of the AML and SVDL method was not available to us, we chose the same examples and experimental conditions for the time series as Reinker et al. and compared our results to those listed in the results section in [15]. We also consider non-equidistant time series. To the best of our knowledge there exists no direct numerical approach for non-equidistant time series with measurement error that is based on the maximum likelihood method.

We generated time series data for two different examples from systems biology using Monte-Carlo simulation [6] and added error terms $\epsilon_i(t_\ell)$ to the population of the i-th species at time t_ℓ. Besides the simple network described in Example 1 we consider a more complex network with eight reactions and five species for the transcription regulation of a repressor protein [15]:

1:	mRNA	$\to$	mRNA + M	5:	DNA + D	$\to$	DNA.D
2:	M	$\to$	$\emptyset$	6:	DNA.D	$\to$	DNA+D
3:	DNA.D	$\to$	mRNA + DNA.D	7:	M + M	$\to$	D
4:	mRNA	$\to$	$\emptyset$	8:	D	$\to$	M + M

The initial molecular populations are $(2, 4, 2, 0, 0)$ for M, D, DNA, mRNA, and DNA.D. The reachable state space of the model is infinite in three dimensions since the populations of mRNA, M, and D are unbounded. The rate constants are $\mathbf{c} = (0.043, 0.0007, 0.0715, 0.00395, 0.02, 0.4791, 0.083, 0.5)$. For the network in Example 1 we chose the same parameters as Reinker et al., namely $\mathbf{c} = (0.0270, 0.1667, 0.40)$.

For the generation of time series data we fix the (true) constants $\mathbf{c}$ and the standard deviation σ of the error terms. We use the SLA and PLA method to estimate $\mathbf{c}$ and σ such that the likelihood of the time series becomes maximal under these parameters. Since in practice only few observation sequences are available, we estimate the parameters based on $K = 5$ observation sequences. As suggested by Reinker et al., we repeat the generation of batches of five observation sequences and the estimation of parameters 100 times to approximate the mean and the standard deviation of the estimators.

Our algorithms for the approximation of the likelihood are implemented in C++ and we run them on an Intel Core i7 at 2.8 Ghz with 8 GB main memory. They are linked to MATLAB's optimization toolbox which we use to minimize the negative log-likelihood. Since we use a global optimization method (MATLAB's global search), the running time of our method depends on the tightness of the intervals that we use as constraints for the unknown parameters as well as on the number of starting points of the global search procedure. We chose intervals that correspond to the order of magnitude of the parameters, i.e., if $c_j \in O(10^n)$ for some $n \in \mathbb{Z}$ then we use the interval $[10^{n-1}, 10^{n+1}]$ as constraint for c_j. E.g. if $c_j = 0.1$ then $n = -1$ and we use the interval $[10^{-2}, 10^0]$. Moreover, for global search we used 20 starting points for the gene expression example and 5 for the transcription regulation example. Note that this is the only difference of our experimental conditions compared to Reinker et al. who use a local optimization method and start the optimization with the true parameters.

In both algorithms we choose a significance threshold of $\delta = 10^{-15}$. Since the PLA method becomes slow if the number of paths that are considered is large, in step 4 of the algorithm we reduce the number of paths that we consider by keeping only the 10 most probable paths. In this way, the computational effort of the PLA method remains tractable even in the case of the transcription regulation network.

Table 2: Estimates for the simple gene expression model using equidistant time series.

Δt (R)	σ	Method	Time	Average (standard deviation) of parameter estimates			
				$c_1 = 0.027$	$c_2 = 0.1667$	$c_3 = 0.4$	σ
1.0 (300)	0.1	AML	–	0.0268(0.0061)	0.1523(0.0424)	0.3741(0.0557)	0.1012(0.0031)
		SVDL	–	0.0229(0.0041)	0.1573(0.0691)	0.4594(0.1923)	–
		SLA	29.4	0.0297(0.0051)	0.1777(0.0361)	0.3974(0.0502)	0.1028(0.0612)
		PLA	2.2	0.0300(0.0124)	0.1629(0.0867)	0.3892(0.0972)	0.1010(0.0792)
	1.0	AML	–	0.0257(0.0054)	0.1409(0.0402)	0.3461(0.0630)	1.0025(0.0504)
		SVDL	–	0.0295(0.0102)	0.1321(0.0787)	0.3842(0.2140)	–
		SLA	8.3	0.0278(0.0047)	0.1868(0.0339)	0.3946(0.0419)	0.9976(0.0476)
		PLA	1.8	0.0278(0.0041)	0.1810(0.0294)	0.3938(0.0315)	0.9938(0.0465)
	3.0	AML	–	0.0250(0.0065)	0.1140(0.0337)	0.3160(0.0674)	3.0292(0.1393)
		SVDL	–	–	–	–	–
		SLA	11.1	0.0285(0.0043)	0.1755(0.0346)	0.3938(0.0508)	2.9913(0.0733)
		PLA	1.7	0.0275(0.0086)	0.1972(0.0902)	0.3894(0.0722)	3.0779(0.0887)
10.0 (30)	0.1	AML	–	–	–	–	–
		SVDL	–	–	–	–	–
		SLA	40.9	0.0273(0.0069)	0.1788(0.04786)	0.3931(0.0599)	0.1086(0.0630)
		PLA	5.2	0.0277(0.0080)	0.1782(0.0517)	0.4057(0.0678)	0.1234(0.0523)
	1.0	AML	–	–	–	–	–
		SVDL	–	–	–	–	–
		SLA	10.2	0.0283(0.0070)	0.1787(0.0523)	0.4018(0.0681)	0.9898(0.0829)
		PLA	3.5	0.0243(0.0057)	0.1665(0.0400)	0.4031(0.0638)	1.0329(0.0859)
	3.0	AML	–	–	–	–	–
		SVDL	–	–	–	–	–
		SLA	12.3	0.0300(0.0110)	0.1960(0.0788)	0.4025(0.0689)	2.9402(0.1304)
		PLA	4.2	0.0210(0.0054)	0.1511(0.0534)	0.4042(0.0616)	3.0629(0.2249)

6.1 Equidistant time series

In the equidistant case, the length of the observation intervals is $\Delta t = t_\ell - t_{\ell-1}$ for all $\ell \in \{1, \ldots, R\}$. In Table 2 and 3 we list the results given in [15] as well as the results of our methods. Reinker et al. do not evaluate the AML method for larger intervals than $\Delta t = 1$ because, as we will discuss in Section 7, the approximation error of the AML method becomes huge in that case. Also, the SVDL method performs poor if σ is large since it does not include measurement errors in the likelihood. Therefore, no results for $\sigma > 1.0$ are provided in [15] for SVDL. In the first three columns we list Δt, the number R of observation points and the true standard deviation σ of the error terms. In column "Time" we compare the average running time (in seconds) of one parameter estimation (out of 100) for SLA and PLA, i.e., the average running time of the maximization of the likelihood based on $K = 5$ observation sequences. It is not meaningful to compare the running times with those in [15] since different optimization methods are used and experiments were run on different machines. Finally, we list estimation results for all four methods (if available). We give the true parameters in the column headings and list the average of 100 estimations and the standard deviation of the estimates (in brackets).

For the simple gene expression (Table 2) and $\Delta t = 1.0$, we find that SLA and PLA have a similar accuracy for the estimation of σ but are consistently more accurate than AML and SVDL for estimating the rate constants. If $\sigma = 0.1$, then the total absolute error for the estimation of $\mathbf{c}$ is 0.041, 0.073, 0.016, 0.018 for AML, SVDL, SLA, PLA, respectively. For $\sigma = 1.0$ we have total absolute errors of 0.081, 0.053, 0.026, 0.021

for AML, SVDL, SLA, PLA. Finally, for $\sigma = 3.0$, AML has a total error of 0.139 while the error for SLA and PLA is 0.017 and 0.041. For $\Delta t = 10$, the results of the SLA and PLA method are accurate even though only 30 observation points are given. Since PLA gives a much coarser approximation, its running time is always shorter (about three to ten times shorter). If σ is large, SLA gives more accurate results than PLA.

In Table 3 we compare results of the transcription regulation for $\sigma = 0$. Note that, for this example, Reinker et al. only give results for the SVDL method with $\Delta t \leq 1.0$ and $\sigma = 0$. Here, we compare results for $\Delta t = 1.0$ since in this case the SVDL method performs best compared to smaller values of Δt. The SLA and PLA method consistently perform better than the SVDL method since they approximate the likelihood more accurately. If $\sigma = 0$, then the accuracy of SLA and PLA is the same (up to the fifth digit). Therefore the results of SLA and PLA are combined in Table 3. The running time of SLA is, however, much slower since it does not reuse the entries of the transition probability matrix T. For $\Delta t = 1.0$, one parameter estimation based on $K = 5$ observations takes about 30 minutes for SLA and only about 2.4 minutes for PLA. For $\Delta t = 10.0$ we have running times about 5 hours(SLA) and 27 minutes (PLA). As for the gene expression example, we expect for larger values of σ the results of SLA to be more accurate than those of PLA.

6.2 Non-equidistant time series

Finally, we consider non-equidistant time series, which can only be handled by the SLA method. During the Monte-Carlo simulation, we generate non-equidistant time series by iteratively choosing $t_{\ell+1} = t_\ell + \mathcal{U}(0,5)$, where $\mathcal{U}(0,5)$ is a random number that is uniformly distributed on $(0,5)$ and $t_0 = 0$. Note that the intervals are not only different within an observation sequence but also for different k, i.e., the times $t_1, \ldots, t_R$ depend on the number k of the corresponding sequence. We consider the transcription regulation model with $\sigma = 1.0$ and $K = 5$ as this is our most complex example. Note that, since the accuracy of the estimation decreases as σ increases, we cannot expect a similar accuracy as in Table 3. For a time horizon of $t = 500$ the average number of observation points per sequence is $R = 500/2.5 = 200$. The estimates computed by SLA are $c_1^* = 0.0384(0.0343)$, $c_2^* = 0.0010(0.0001)$, $c_3^* = 0.0642(0.0249)$, $c_4^* = 0.0044(0.0047)$, $c_5^* = 0.0273(0.0073)$, $c_6^* = 0.5498(0.1992)$, $c_7^* = 0.0890(0.0154)$, $c_8^* = 0.5586(0.0716)$, and $\sigma^* = 0.9510(0.0211)$, where we averaged over 100 repeated estimations and give the standard deviation in brackets. Recall that the true constants are $c_1 = 0.043$, $c_2 = 0.0007$, $c_3 = 0.0715$, $c_4 = 0.00395$, $c_5 = 0.02$, $c_6 = 0.4791$, $c_7 = 0.083$, and $c_8 = 0.5$. The average running time of one estimation was 19 minutes.

7 Related work

In the context of stochastic chemical kinetics, parameter inference methods are either based on Bayesian inference [2, 18, 20] or maximum likelihood estimation [15, 19, 17]. The advantage of the latter method is that the corresponding estimators are, in a sense, the most informative estimates of unknown parameters [10] and have desirable mathematical properties such as unbiasedness, efficiency, and normality [11]. On the other hand, the computational complexity of maximum likelihood estimation is high. If an analytic solution of (6) is not possible, then, as a part of the nonlinear optimization problem, the likelihood and its derivatives have to be calculated. Monte-Carlo simu-

Table 3: Estimates for the transcription regulation model using equidistant time series.

Δt (R)	Method	Average (standard deviation) of parameter estimates			
		$c_1 = 0.043$	$c_2 = 0.0007$	$c_3 = 0.0715$	$c_4 = 0.00395$
1.0 (500)	SVDL	0.0477(0.0155)	0.0006(0.0004)	0.0645(0.0190)	0.0110(0.0195)
	PLA/SLA	0.0447(0.0036)	0.0007(0.0001)	0.0677(0.0115)	0.0034(0.0014)
10.0 (50)	PLA/SLA	0.0417(0.0069)	0.0005(0.0002)	0.0680(0.0075)	0.0038(0.0026)

Δt (R)	Method	Average (standard deviation) of parameter estimates			
		$c_5 = 0.02$	$c_6 = 0.4791$	$c_7 = 0.083$	$c_8 = 0.5$
1.0 (500)	SVDL	0.0159(0.0107)	0.2646(0.0761)	0.0149(0.0143)	0.0615(0.0332)
	PLA/SLA	0.0193(0.0008)	0.4592(0.0169)	0.0848(0.0024)	0.5140(0.0166)
10.0 (50)	PLA/SLA	0.0188(0.0039)	0.4359(0.0822)	0.0836(0.0016)	0.4892(0.0164)

lation has been used to estimate the likelihood [17, 19]. During the repeated random sampling it is difficult to explore those parts of the state space that are unlikely under the current rate parameters. Thus, especially if the rates are very different from the true parameters, many simulation runs are necessary to calculate an accurate approximation of the likelihood. To the best of our knowledge, Reinker et al. provide the first maximum likelihood estimation that is not based on Monte-Carlo simulation but calculates the likelihood numerically [15]. They propose the AML method during which the matrices P_ℓ are approximated. In order to keep the computational effort low, they allow at most two jumps of the Markov process during $[t_\ell, t_{\ell+1})$. Moreover, they ignore all states for which $|O_i(t_\ell) - x_{i\ell}|$ is greater than $3\sqrt{\sigma}$. This has the disadvantage that $\mathcal{L}$ is zero (and its derivative) if the values for the rate constants are far off the true values. If $\mathcal{L}$ is zero, then the derivatives provide no information about how the rate constants have to be altered in order to increase the likelihood. Thus, initially very good estimates for the rate constants must be known to apply this kind of truncation. On the other hand, the method that we propose neglects only insignificant terms of the likelihood. For this reason the likelihood and its derivatives do not become zero during the computation and it is always possible to follow the gradient in order to obtain higher likelihoods. Another disadvantage of the AML method is that, if the observation intervals are longer, the likelihood may not be approximated accurately since the assumption that only two reactions occur within an observation interval is not valid. Extending the AML approach to more than two steps would result in huge space requirements and perform slow since the state space is explored in a breath-first search manner and too many states would be considered even though their contribution to the likelihood is very small. In our approach we allow an arbitrary number of reactions during $[t_\ell, t_{\ell+1})$[2]. Therefore, our method is not restricted to reaction networks where the speed of all reactions is at most of the same time scale as the observation intervals. The second approach proposed by Reinker et al., called SVDL method, is based on the assumption that the propensities α_j stay constant during $[t_\ell, t_{\ell+1})$. Again, this assumption only applies to small observation intervals. Moreover, the SVDL method does not take into account measurement errors and is thus only appropriate if σ is very small. Further differences between the approach of Reinker et al. and our approach are that we use a global optimization technique (MATLAB's global search) while Reinker et al. use a local solver, namely the quasi-Newton method. Finally, the approach in [15]

[2]During one step of our numerical integration, we assume that only four reactions are possible. The time step h of the numerical integration does, however, not depend on the $[t_\ell, t_{\ell+1})$ but is dynamically chosen in such a way that performing more than four steps is very unlikely.

requires observations at equidistant time instances, which is not necessary for the SLA method.

8 Conclusion

Parameter inference for stochastic models of cellular processes demands huge computational resources. We proposed two numerical methods, called SLA and PLA, that approximate maximum likelihood estimators for a given set of observations. Both methods do not make any assumptions about the number of reactions that occur within an observation interval. The SLA method allows for an estimation based on arbitrarily spaced intervals while the PLA method requires equidistant intervals.

Many reaction networks involve both small populations and large populations. In this case stochastic hybrid models are most appropriate since they combine the advantages of deterministic and stochastic representations. We plan to extend our algorithms to the stochastic hybrid setting proposed in [8] to allow inference for more complex networks. Further future work also includes more rigorous truncations for the SLA method and the parallelization of the algorithm.

References

[1] J. R. Banga and E. Balsa-Canto. Parameter estimation and optimal experimental design. *Essays Biochem.*, 45:195–209, 2008.

[2] R. Boys, D. Wilkinson, and T. Kirkwood. Bayesian inference for a discretely observed stochastic kinetic model. *Statistics and Computing*, 18:125–135, 2008.

[3] K. Burrage, M. Hegland, F. Macnamara, and B Sidje. A Krylov-based finite state projection algorithm for solving the chemical master equation arising in the discrete modelling of biological systems. In *Proceedings of the Markov 150th Anniversary Conference*, pages 21–38. Boson Books, 2006.

[4] I. Chou and E. Voit. Recent developments in parameter estimation and structure identification of biochemical and genomic systems. *Mathematical Biosciences*, 219(2):57 – 83, 2009.

[5] D. Fernández Slezak, C. Suárez, G. Cecchi, G. Marshall, and G. Stolovitzky. When the optimal is not the best: Parameter estimation in complex biological models. *PLoS ONE*, 5(10):e13283, 10 2010.

[6] D. T. Gillespie. Exact stochastic simulation of coupled chemical reactions. *J. Phys. Chem.*, 81(25):2340–2361, 1977.

[7] I. Golding, J. Paulsson, S. Zawilski, and E. Cox. Real-time kinetics of gene activity in individual bacteria. *Cell*, 123(6):1025–1036, December 2005.

[8] T. Henzinger, M. Mateescu, L. Mikeev, and V. Wolf. Hybrid numerical solution of the chemical master equation. In *Proc. of CMSB'10*. ACM Digital Library, 2010.

[9] T. Henzinger, M. Mateescu, and V. Wolf. Sliding window abstraction for infinite Markov chains. In *Proc. CAV*, volume 5643 of *LNCS*, pages 337–352. Springer, 2009.

[10] J. J. Higgins. Bayesian inference and the optimality of maximum likelihood estimation. *Int. Stat. Rev.*, 45(1):9–11, 1977.

[11] L. Ljung. *System Identification: Theory for the User (2nd Edition)*. Prentice Hall PTR, 1998.

[12] A. Loinger, A. Lipshtat, N. Q. Balaban, and O. Biham. Stochastic simulations of genetic switch systems. *Physical Review E*, 75:021904, 2007.

[13] M. Mateescu, V. Wolf, F. Didier, and T.A. Henzinger. Fast adaptive uniformisation of the chemical master equation. *IET Systems Biology*, 4(6):441–452, 2010.

[14] B. Munsky and M. Khammash. The finite state projection algorithm for the solution of the chemical master equation. *J. Chem. Phys.*, 124:044144, 2006.

[15] S. Reinker, R.M. Altman, and J. Timmer. Parameter estimation in stochastic biochemical reactions. *IEEE Proc. Syst. Biol*, 153:168–178, 2006.

[16] R. Sidje, K. Burrage, and S. MacNamara. Inexact uniformization method for computing transient distributions of Markov chains. *SIAM J. Sci. Comput.*, 29(6):2562–2580, 2007.

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

[18] T. Toni, D. Welch, N. Strelkowa, A. Ipsen, and M. Stumpf. Approximate Bayesian computation scheme for parameter inference and model selection in dynamical systems. *J. R. Soc. Interface*, 6(31):187–202, 2009.

[19] B. Uz, E. Arslan, and I. Laurenzi. Maximum likelihood estimation of the kinetics of receptor-mediated adhesion. *J. Theor. Biol.*, 262(3):478 – 487, 2010.

[20] D. J. Wilkinson. *Stochastic Modelling for Systems Biology*. Chapman & Hall, 2006.