Stochastic Rate Parameter Inference using the Cross-Entropy Method

Jeremy Revell and Paolo Zuliani

School of Computing, Newcastle University, UK
{j.d.revell1,paolo.zuliani}@ncl.ac.uk

Abstract. We present a new, efficient algorithm for inferring, from time-series data or high-throughput data (*e.g.*, flow cytometry), stochastic rate parameters for chemical reaction network models. Our algorithm combines the Gillespie stochastic simulation algorithm (including approximate variants such as tau-leaping) with the cross-entropy method. Also, it can work with incomplete datasets missing some model species, and with multiple datasets originating from experiment repetitions. We evaluate our algorithm on a number of challenging case studies, including bistable systems (Schlögl's and toggle switch) and experimental data.

1 Introduction

In this paper we are concerned with the inference of biochemical reaction stochastic rate parameters from data. Reactions are discrete events that can occur randomly at any time with a rate dependent on the chemical kinetics [40]. It has recently become clear that stochasticity can produce dynamics profoundly different from the corresponding deterministic models. This is the case, *e.g.*, in genetic systems where key species are present in small numbers or where key reactions occur at a low rate [23], resulting in transient, stochastic bursts of activity [4, 24]. The standard model for such systems is the Markov jump process popularised by Gillespie [13, 14]. Given a collection of reactions modelling a biological system and time-course data, the *stochastic parameter inference problem* is to find parameter values for which the Gillespie model's temporal behaviour is most consistent with the data. This is a very difficult problem, much harder, both theoretically and computationally, than the corresponding problem for deterministic kinetics — see, *e.g.*, [41, Section 1.3]. One simple reason is because stochastic models can behave widely differently from the same initial conditions. (The related issue of parameter non-identifiability is outside the scope of this paper, but the interested reader can find more in, *e.g.*, [37, 38] and references therein.) Additionally, experimental data is usually sparse and most often involves only a limited subset of a model's species; and the system under study might exhibit multimodal behaviour. Also, data might not directly relate to a species, it might be measured in arbitrary units (*e.g.*, fluorescence measurements), thus requiring the estimation of scaling factors, or it might be described by frequency distributions (*e.g.*, high-throughput data such as flow cytometry). Stochastic parameter inference is thus a fundamental and challenging problem in systems biology, and it is crucial for obtaining validated and predictive models.

In this paper we propose an approach for the parameter inference problem that combines Gillespie's Stochastic Simulation Algorithm (SSA) with the cross-entropy (CE) method [27]. The CE method has been successfully used in optimisation, rare–event probability estimation, and other domains [29]. For parameter inference, Daigle *et al.* [8] combined a stochastic Expectation–Maximisation (EM) algorithm with a modified cross-entropy method. We instead develop the cross-entropy method in its own right, discarding the costly EM algorithm steps. We also show that our approach can utilise approximate, faster SSA variants such as tau-leaping [15]. Summarising, the main contributions of this paper are:

- we present a new, cross entropy-based algorithm for the stochastic parameter inference problem that outperforms previous, state–of–the–art approaches;
- our algorithm can work with multiple, incomplete, and distribution datasets;
- we show that tau-leaping can be used within our technique;
- we provide a thorough evaluation of our algorithm on a number of challenging case studies, including bistable systems (Schlögl model and toggle switch) and experimental data.

2 Background

Notation Given a system with n chemical species, the state of the system at time t is represented by the vector $\boldsymbol{x}(t) = (x_1(t), \ldots, x_n(t))$, where x_i represents the number of molecules of the ith species, S_i, for $i \in \{1, \ldots, n\}$. A well-mixed system within a fixed volume at a constant temperature can be modelled by a continuous-time Markov chain (CTMC) [13, 14]. The CTMC state changes are triggered by the (probabilistic) occurrences of chemical reactions. Given m chemical reactions, let $\mathcal{R}_j$ denote the jth reaction of type:

$$\mathcal{R}_j \quad : \quad \nu^-_{j,1} S_1 + \ldots + \nu^-_{j,n} S_n \xrightarrow{\theta_j} \nu^+_{j,1} S_1 + \ldots + \nu^+_{j,n} S_n,$$

where the vectors $\boldsymbol{\nu}^-_j$ and $\boldsymbol{\nu}^+_j$ represent the stoichiometries of the underlying chemical kinetics for the reactants and products, respectively. Let $\boldsymbol{\nu}_j \in \mathbb{Z}^n$ denote the overall (non-zero) state-change vector for the jth reaction type, specifically $\boldsymbol{\nu}_j = \boldsymbol{\nu}^+_j - \boldsymbol{\nu}^-_j$, for $j \in \{1, \ldots, m\}$. Assuming mass action kinetics (and omitting time dependency for $\boldsymbol{x}(t)$), the reaction $\mathcal{R}_j$ leads to the propensity [41]:

$$h_j(\boldsymbol{x}, \boldsymbol{\theta}) = \theta_j \alpha_j(\boldsymbol{x}) = \theta_j \prod_{i=1}^{n} \binom{x_i}{\nu^-_{j,i}}, \tag{1}$$

where $\boldsymbol{\theta} = (\theta_1, \ldots, \theta_m)^\intercal$ is the vector of rate constants. In general, $\boldsymbol{\theta}$ is unknown and must be estimated from experimental data — that is the aim of our work. Our algorithm can work with propensity functions factorisable as in (1), but it is not restricted to mass action kinetics (*i.e.*, the functions α_j's can be arbitrary).

Cross-Entropy Method For Optimisation The Kullback-Leibler divergence [20] or cross-entropy (CE) between two probability densities g and h is:

$$\mathcal{D}(g, h) = \mathbb{E}_g \left[\ln \frac{g(\boldsymbol{X})}{h(\boldsymbol{X})} \right] = \int g(\boldsymbol{x}) \ln \frac{g(\boldsymbol{x})}{h(\boldsymbol{x})} d\boldsymbol{x}$$

where $\boldsymbol{X}$ is a random variable with density g, and $\mathbb{E}_g$ is expectation w.r.t. g. Note that $\mathcal{D}(g,h) \geq 0$ with equality iff $g = h$ (almost everywhere). (However, $\mathcal{D}(g,h) \neq \mathcal{D}(h,g)$.) The CE has been successfully adopted for a wide range of hard problems, including rare event simulation for biological systems [7], discrete, and continuous optimisation [29, 28]. Consider the minimisation of an *objective* function J over a space χ (assuming such minimum exists), $\gamma^* = \min\limits_{x \in \chi} J(x)$. The CE method performs a Monte Carlo search over a parametric family of densities $\{f(\cdot;\boldsymbol{v}), \boldsymbol{v} \in \mathcal{V}\}$ on χ that contains as a limit the (degenerate) Dirac density that puts its entire mass on a value $x^* \in \chi$ such that $J(x^*) = \gamma^*$ — the so called *optimal* density. The key idea is to use the CE to measure how far a candidate density is from the optimal density. In particular, the method solves a sequence of optimisation problems of the type below for different values of γ by minimising the CE between a putative optimal density $g^*(\boldsymbol{x}) \propto I_{\{J(\boldsymbol{x}) \leq \gamma\}} f(\boldsymbol{x}, \boldsymbol{v}^*)$ for some $\boldsymbol{v}^* \in \mathcal{V}$, and the density family $\{f(\cdot;\boldsymbol{v}), \boldsymbol{v} \in \mathcal{V}\}$

$$\min_{\boldsymbol{v} \in \mathcal{V}} \mathcal{D}(g^*, f(\cdot;\boldsymbol{v})) = \max_{\boldsymbol{v} \in \mathcal{V}} \mathbb{E}_u \left[I_{\{J(\boldsymbol{X}) \leq \gamma\}} \ln f(\boldsymbol{X};\boldsymbol{v}) \right] \tag{2}$$

where I is the indicator function and $\boldsymbol{X}$ has density $f(\cdot;\boldsymbol{u})$ for $\boldsymbol{u} \in \mathcal{V}$. The definition of density g^* above essentially means that, for a given γ, we only consider densities that are positive only for arguments $\boldsymbol{x}$ for which $J(\boldsymbol{x}) \leqslant \gamma$. The generic CE method involves a 2-step procedure which alternates solving (2) for a candidate g^* with adaptively updating γ. In practice, problem (2) is solved approximately via a Monte Carlo adaptation, *i.e.*, by taking sample averages as estimators for $\mathbb{E}_u$. The output of the CE method is a sequence of putative optimal densities identified by their parameters $\hat{\boldsymbol{v}}_0, \hat{\boldsymbol{v}}_1, \ldots, \hat{\boldsymbol{v}}^*$, and performance scores $\hat{\gamma}_0, \hat{\gamma}_1, \ldots, \hat{\gamma}^*$, which improve with probability 1. For our problem, a key benefit of the CE method is that an analytic solution for (2) can be found when $\{f(\cdot;\boldsymbol{v}), \boldsymbol{v} \in \mathcal{V}\}$ is the exponential family of distributions. (More details in [29].)

Cross-Entropy Method for the SSA We denote by r_j the number of firings of the jth reaction channel, τ_i the time between the ith and $(i-1)$th reaction, and τ_{r+1} the final time interval at the end of the simulation in which no reaction occurs. It can be shown that an exact SSA trajectory $\boldsymbol{z} = (\boldsymbol{x}_0, \ldots, \boldsymbol{x}_r)$, where r is the total number of reaction events $r = \sum_{j=1}^m r_j$, belongs to the exponential family of distributions [41] — whose optimal CE parameter can be found analytically. Daigle *et al.* [8] showed that the solution of (2) for the SSA likelihood yields the following Monte Carlo estimate of the optimal CE parameter v_j^*,

$$\hat{\theta}_j = \hat{v_j^*} = \frac{\sum_{k=1}^K r_{jk} I_{\{J(\boldsymbol{z}_k) \leq \gamma\}}}{\sum_{k=1}^K I_{\{J(\boldsymbol{z}_k) \leq \gamma\}} \left(\sum_{i=1}^{r_k+1} \alpha_j(\boldsymbol{x}_{i-1,k}) \tau_{ik} \right)} \tag{3}$$

where K is the number of SSA trajectories of the Monte Carlo approximation of (2), $\boldsymbol{z}_k$ is the kth trajectory, r_{jk} and τ_{ik} are as before but w.r.t. the kth trajectory, $\boldsymbol{x}_{i,k}$ denotes the state after the $(i-1)$th reaction in the kth trajectory, and the fraction is defined only when the denominator is nonzero (*i.e.*, there is at least one trajectory $\boldsymbol{z}_k$ for which $J(\boldsymbol{z}_k) \leq \gamma$ — so-called *elite* samples). Note for $\gamma = 0$, the CE estimator (3) coincides with the maximum likelihood estimator (MLE) for θ_j over the same trajectory. Following [7] and [26, Section 5.3.4], it is easy to show that a Monte Carlo estimator of the covariance matrix of the optimal parameter estimators (3) is given (written in operator style) by the matrix:

$$\hat{\Sigma}^{-1} = \Bigg[- \frac{1}{K_E} \sum_{k \in E} \frac{\partial^2}{\partial \theta^2} - \frac{1}{K_E} \sum_{k \in E} \frac{\partial}{\partial \theta} \cdot \frac{\partial}{\partial \theta}^{\mathrm{T}} + \frac{1}{K_E^2} \left(\sum_{k \in E} \frac{\partial}{\partial \theta} \right) \cdot \left(\sum_{k \in E} \frac{\partial}{\partial \theta} \right)^{\mathrm{T}} \Bigg] (\log f(\theta | \boldsymbol{x}, \boldsymbol{z}_k)) \quad (4)$$

where E is the set of elite samples, $K_E = |E|$, the operator $\frac{\partial^2}{\partial \theta^2}$ returns a $m \times m$ matrix, $\frac{\partial}{\partial \theta}$ returns an m-dimensional vector ($m \times 1$ matrix), and $\frac{\partial}{\partial \theta}^{\mathrm{T}}$ denotes matrix transpose. From Eq. (4) parameter variance estimates can be readily derived. However, a more numerically stable option is to approximate the variance of the jth parameter estimator using the sample variance

$$\hat{\sigma}_j^2 = \frac{1}{K_E} \sum_{k \in E} \left(\frac{r_{jk}}{\sum_{i=1}^{r_k+1} \alpha_j(\boldsymbol{x}_{i-1,k}) \tau_{ik}} - \hat{\theta}_j \right)^2 . \quad (5)$$

3 Methods

In this section, we present our *stochastic rate parameter inference with cross-entropy* (SPICE) algorithm.

Overview To efficiently sample the parameter space, we treat each stochastic rate parameter as being log-normally distributed, *i.e.*, $\theta_j \sim \mathrm{Lognormal}(\omega_j, \mathrm{var}(\omega_j))$, where $\omega_j = \log(\theta_j)$ is the log-transformed parameter calculated analagously to (3) and (4), respectively. For the initial iteration, we sample the parameter vector $\boldsymbol{\theta}$ from the (log-transformed) desired parameter search space $[\boldsymbol{\theta}^{(0)}_{\mathrm{MIN}}, \boldsymbol{\theta}^{(0)}_{\mathrm{MAX}}]$ using a Sobol low-discrepancy sequence [33] to ensure adequate coverage. Subsequent iterations then generate a sequence of distribution parameters $\{(\gamma_n, \boldsymbol{\theta}_n, \boldsymbol{\Sigma}_n)\}$ which aim to converge to the optimal parameters as follows:

1. **Updating of γ_n**: Generate K sample trajectories using the SSA, $\boldsymbol{z}_1, \dots, \boldsymbol{z}_K$, from the model $f(\cdot; \boldsymbol{\theta}^{(n-1)})$ with $\boldsymbol{\theta}^{(n-1)}$ sampled from the lognormal distribution, and sort them in order of their performances $J_{1'} \leq \cdots \leq J_{K'}$ (see Eqs. (7) and (6) for the actual definition of the performance, or score, function we adopt). For a fixed small ρ, say $\rho = 10^{-2}$, let $\hat{\gamma}_n$ be defined as the ρth quantile of $J(\boldsymbol{z})$, *i.e.*, $\hat{\gamma}_n = J_{(\lceil \rho K \rceil)}$.
2. **Updating of $\boldsymbol{\theta}_n$**: Using the estimated level $\hat{\gamma}_n$, use the same K sample trajectories $\boldsymbol{z}_1, \dots, \boldsymbol{z}_K$ to derive $\hat{\boldsymbol{\theta}}_n$ and $\hat{\boldsymbol{\sigma}}_n^2$ from the solution of Eqs. (3) and (4). In case of numerical issues (or undersampling) in our implementation we switch to (5) for updating the variance.

The SPICE algorithm's pseudocode is shown in Algorithm 1. This 2-step approach provides a simple iterative scheme which converges asymptotically to the optimal density. A reasonable termination criteria to take would be to stop if $\hat{\gamma}_n \not\lesssim \hat{\gamma}_{n-1} \not\lesssim \dots$ for a fixed number of iterations. In general, more samples are required as the mean and variance of the estimates approach their optima.

Adaptive Sampling We adaptively update the number of samples K_n taken at each iteration. The reasoning is to ensure the parameter estimates improve with statistical significance at each step. Thus, our method allows the algorithm to make faster evaluations early on in the iterative process, and concentrate simulation time on later iterations, where it becomes increasingly hard to distinguish significant improvements of the estimated parameters. We update our parameters based on a fixed number of elite samples, K_E, satisfying $J(\boldsymbol{z}) \leq \gamma$. The performance of the 'best' elite sample is denoted J_n^*, while the performance of the 'worst' elite sample — previously given by the ρth quantile of $J(\boldsymbol{z})$ — is $\hat{\gamma}_n$. The quantile parameter ρ is adaptively updated each iteration as $\rho_n = K_E/K_n$, where K_E is typically taken to be 1–10% of the base number of samples K_0. At each iteration, a check is made for improvement in either of the best or worst performing elite samples, *i.e.*, if, $J_n^* < J_{n-1}^*$ or $\hat{\gamma}_n < \hat{\gamma}_{n-1}$, then we can update our parameters and proceed to the next iteration. If no improvement in either values are found, the number of samples K_n in the current iteration is increased in increments, up to a maximum $K_{\max}$. If we hit the maximum number of samples $K_{\max}$ for c iterations (*e.g.*, $c = 3$), then this suggests no further significant improvement can be made given the restriction on the number of samples.

Objective Function The SPICE algorithm has been developed to handle an arbitrary number of datasets. Given N time series datasets, SPICE associates N objective function scores with each simulated trajectory. Each objective value corresponds to the standard sum of L^2 distances of the trajectory across all time points in the respective dataset:

$$J_n(\boldsymbol{z}) = \sum_{t=1}^{T} (\boldsymbol{y}_{n,t} - \boldsymbol{x}_t)^2 \qquad 1 \leq n \leq N \tag{6}$$

where $\boldsymbol{x}_t = \boldsymbol{x}(t)$ and $\boldsymbol{y}_{n,t}$ is the datapoint at time t in the nth dataset. To ensure adequate coverage of the data, we choose our elite samples to be the best performing quantile of trajectories for each individual dataset (with scores J_n).

In the absence of temporal correlation within the data (*e.g.*, when measurements between time points are independent or individual cells cannot be tracked as in flow cytometry data), we instead construct an empirical Gaussian mixture model for each time point within the data. Each mixture model at time t is comprised of N multivariate normal distributions, each with a vector of mean values $\boldsymbol{y}_{n,t}$ corresponding to the *observed* species in the nth dataset, and diagonal covariance matrix $\boldsymbol{\sigma}_n^2$ corresponding to an error estimate or variance of the measurements on the species. In our experiments we used a 10% standard deviation, as we did not have any information about measurement noise. We then take the objective score function to be proportional to the negative log-likelihood of the simulated trajectory w.r.t. the data:

$$J_n(\boldsymbol{z}) = -\sum_{t=1}^{T} \ln \left(\sum_{n=1}^{N} \exp \left[-\frac{1}{2} (\boldsymbol{y}_{n,t} - \boldsymbol{x}_t)^\intercal \boldsymbol{\sigma}_n^{-2} (\boldsymbol{y}_{n,t} - \boldsymbol{x}_t) \right] \right). \tag{7}$$

Smoothed Updates We implement the parameter smoothing update formula

$$\hat{\boldsymbol{\theta}}^{(n)} = \lambda\tilde{\boldsymbol{\theta}}^{(n)} + (1-\lambda)\hat{\boldsymbol{\theta}}^{(n-1)}, \quad \hat{\boldsymbol{\sigma}}^{(n)} = \beta_n\tilde{\boldsymbol{\sigma}}^{(n)} + (1-\beta_n)\hat{\boldsymbol{\sigma}}^{(n-1)}$$

where $\beta_n = \beta - \beta\left(1-\frac{1}{n}\right)^q$, $\lambda\in(0,1]$, $q\in\mathbb{N}^+$ and $\beta\in(0,1)$ are smoothing constants, and $\tilde{\boldsymbol{\theta}}, \tilde{\boldsymbol{\sigma}}$ are outputs from the solution of the cross-entropy in equation (2), approximated by (3) and (4), respectively. Parameter smoothing between iterations has three important benefits: (i) the parameter estimates converge to a more stable value, (ii) it reduces the probability of a parameter value tending towards zero within the first few iterations, and (iii) it prevents the sampling distribution from converging too quickly to a degenerate point probability mass at a local minima. Furthermore, [6] provide a proof that the CE method converges to an optimal solution with probability 1 in the case of smoothed updates.

Multiple Shooting and Particle Splitting SPICE can optionally utilise these two techniques for trajectory simulation between time intervals. For multiple shooting we construct a sample trajectory comprised of T intervals matching the time stamps within the data $\boldsymbol{y}$. Originally [42], each segment from $\boldsymbol{x}_{t-1}$ to $\boldsymbol{x}_t$ was simulated using an ODE model with the initial conditions set to the previous time point of the dataset, *i.e.*, $\boldsymbol{x}_{t-1} = \boldsymbol{y}_{t-1}$. We instead treat the data as being mixture-normally distributed, thus we sample our initial conditions $\boldsymbol{x}_{t-1} \sim \mathcal{N}(\boldsymbol{y}_{n,t-1}, \boldsymbol{\sigma}^2_{n,t-1})$, where the index of the time series n is first uniformly sampled. Using the SSA, each piecewise section of a trajectory belonging to sample k is then simulated with the same parameter vector $\boldsymbol{\theta}$. For particle splitting we adopt a multilevel splitting approach as in [8], and the objective function is calculated after the simulation of each segment from $\boldsymbol{x}_{t-1}$ to $\boldsymbol{x}_t$. The trajectories $\boldsymbol{z}_k$ satisfying $J(\boldsymbol{z}_k) \leq \hat{\gamma}$ are then re-sampled with replacement K_n times before simulation continues (recall K_n is the number of samples in the nth iteration). This process aims at discarding poorly performing trajectories in favour of those 'closest' to the data. This will in turn create an enriched sample, at the cost of introducing an aspect of bias propagation.

Hyperparameters SPICE allows for the inclusion of hyperparameters $\boldsymbol{\phi}$ (*e.g.*, scaling constants, and non kinetic-rate parameters), which are sampled (logarithmically) alongside $\boldsymbol{\theta}$. These hyperparameters are updated at each iteration via the standard CE method.

Tau-Leaping With inexact, faster methods such as tau-leaping [15] a degree of accuracy is traded off in favour of computational performance. Thus, we are interested in replacing the SSA with tau-leaping in our SPICE algorithm. The next Proposition shows that with a tau-leaping trajectory we get the same form for the optimal CE estimator as in (3).

Proposition 1. *The CE solution for the optimal rate parameter over a tau-leaping trajectory is the same as that for a standard SSA trajectory.*

Proof. We shall use the same notation of Section 2 and further assume a trajectory in which state changes occur at times t_l, for $l \in \{0, 1, \ldots, L\}$. For each given time interval of size τ_l of the tau-leaping algorithm, $k_{jl} \in \mathbb{Z}^+$ firings of each reaction channel $\mathcal{R}_j$ are sampled from a Poisson process with mean $\lambda_{jl} = \theta_j \alpha_j(\boldsymbol{x}_{t_l})\tau_l$. Thus, the probability of firing k_{jl} reactions, in the interval $[t_l, t_l + \tau_l)$, given the initial state $\boldsymbol{x}_{t_l}$ is $P(k_{jl}|\boldsymbol{x}_{t_l}, \lambda_{jl}) = \exp\{-\lambda_{jl}\}(\lambda_{jl})^{k_{jl}}/k_{jl}!$, where $P(0|\boldsymbol{x}_{t_l}, 0) = 1$. Therefore, the combined probability across all reaction channels is:

$$\prod_{j=1}^{m} P(k_{jl}|\boldsymbol{x}_{t_l}, \lambda_{jl}) = \prod_{j=1}^{m} \frac{\exp\{-\lambda_{jl}\}(\lambda_{jl})^{k_{jl}}}{k_{jl}!} \ .$$

Extending for the entire trajectory, the complete likelihood is given by:

$$\mathcal{L} = \prod_{l=0}^{L} \prod_{j=1}^{m} P(k_{jl}|\boldsymbol{x}_{t_l}, \lambda_{jl}) = \prod_{l=0}^{L} \prod_{j=1}^{m} \frac{\exp\{-\lambda_{jl}\}(\lambda_{jl})^{k_{jl}}}{k_{jl}!} \ .$$

We can conveniently factorise the likelihood into component likelihoods associated with each reaction channel as $\mathcal{L} = \prod_{j=1}^{m} \mathcal{L}_j$, where each component $\mathcal{L}_j$ is given by $\mathcal{L}_j = \prod_{l=0}^{L} \frac{\exp\{-\lambda_{jl}\}(\lambda_{jl})^{k_{jl}}}{k_{jl}!}$. Expanding λ_{jl}:

$$\begin{aligned}\mathcal{L}_j &= \prod_{l=0}^{L} \frac{\exp\{-\theta_j \alpha_j(\boldsymbol{x}_{t_l})\tau_l\}(\theta_j \alpha_j(\boldsymbol{x}_{t_l})\tau_l)^{k_{jl}}}{k_{jl}!} \\ &= \theta_j^{r_j} \exp\left\{-\theta_j \sum_{l=0}^{L} \alpha_j(\boldsymbol{x}_{t_l})\tau_l\right\} \prod_{l=0}^{L} \frac{(\alpha_j(\boldsymbol{x}_{t_l})\tau_l)^{k_{jl}}}{k_{jl}!},\end{aligned}$$

where $r_j = \sum_{l=0}^{L} k_{jl}$, *i.e.*, the total number of firings of reaction channel $\mathcal{R}_j$. From [29], the solution to (2) can be found by solving:

$$\mathbb{E}_u \left[I_{\{J(\boldsymbol{X}) \geq \gamma\}} \nabla \ln \mathcal{L}_j \right] = 0,$$

given that the differentiation and expectation operators can be interchanged. Expanding $\ln \mathcal{L}_j$ and simplifying, we get:

$$\mathbb{E}_u \left[I_{\{J(\boldsymbol{X}) \geq \gamma\}} \nabla \left(\ln \theta_j^{r_j} - \theta_j \sum_{l=0}^{L} \alpha_j(\boldsymbol{x}_{t_l})\tau_l + \ln \left\{ \prod_{l=0}^{L} \frac{(\alpha_j(\boldsymbol{x}_{t_l})\tau_l)^{k_{jl}}}{k_{jl}!} \right\} \right) \right] = 0.$$

We can then take the derivative, ∇, with respect to θ_j,

$$\mathbb{E}_u \left[I_{\{J(\boldsymbol{X}) \geq \gamma\}} \left(\frac{r_j}{\theta_j} - \sum_{l=0}^{L} \alpha_j(\boldsymbol{x}_{t_l})\tau_l \right) \right] = 0.$$

It is simple to see that the previous entity holds when $r_j/\theta_j = \sum_{l=0}^{L} \alpha_j(\boldsymbol{x}_{t_l})\tau_l$, yielding the Monte Carlo estimate,

$$\hat{\theta}_j = \frac{\sum_{k=1}^{K} I_{\{J(\boldsymbol{z}_k) \leq \gamma\}} r_{jk}}{\sum_{k=1}^{K} I_{\{J(\boldsymbol{z}_k) \leq \gamma\}} \sum_{l=0}^{L} \alpha_j(\boldsymbol{x}_{t_l,k})\tau_{l,k}}.$$

$\square$

Algorithm 1: SPICE — Stochastic Rate Parameter Inference with Cross-Entropy

```
input : Datasets represented by mixture models $\Phi_i$ at times $t_i$ for $0 \leqslant i \leqslant L$,
        initial parameter bounds (log-transformed) $\left[\boldsymbol{\theta}^{(0)}_{\text{MIN}}, \boldsymbol{\theta}^{(0)}_{\text{MAX}}\right]$, quantile $\rho$.
output: Estimate of parameters $\hat{\boldsymbol{\theta}}^{(n)}$, and their variances $\hat{\boldsymbol{\Sigma}}^{(n)}$.
1  Iteration $n \leftarrow 1$
2  Generate Sobol sequence $S \leftarrow \left[\hat{\boldsymbol{\theta}}^{(0)}_{\text{MIN}}, \hat{\boldsymbol{\theta}}^{(0)}_{\text{MAX}}\right]$ hypercube
3  Initial sample size $K_1 \leftarrow K_{\min}$
4  Initialise $\gamma_1 \leftarrow \infty$
5  repeat
6      for $i \leftarrow 1$ to $L$ do
7          Set initial time point $t_0 \leftarrow t_{i-1}$
8          for $k \leftarrow 1$ to $K_n$ do
9              if $i = 1$ then
10                 Set initial state $\boldsymbol{x} \leftarrow \boldsymbol{y}_0$
11                 if $n = 1$ then
12                     Sample parameters from Sobol sequence $\boldsymbol{\theta}_k \leftarrow S(k)$
13                 else
14                     Sample parameters from the parameter distribution
                         $\boldsymbol{\theta}_k \sim \text{Lognormal}\left(\hat{\boldsymbol{\theta}}^{(n-1)}, \hat{\boldsymbol{\Sigma}}^{(n-1)}\right)$
15             else
16                 if Method = Multiple Shooting then
17                     Sample the starting state from the distribution of the data
                         $\boldsymbol{x} \sim \Phi_i$
18                 else
19                     Continue from the end state of the current simulation
                         $\boldsymbol{x} \leftarrow \boldsymbol{z}_{i-1,k}$
20             Forward simulate $\boldsymbol{z}_{i,k} \leftarrow$SSA$(\boldsymbol{x}, t_0, t_i, \boldsymbol{\theta}_k)$
21             if (Method = Splitting) or $(i = L)$ then
                   // depending on type of data, use (6) or (7)
22                 Calculate the cost function $d_k \leftarrow J(\boldsymbol{z}_k)$
23         if Method = Splitting then
24             Sample with replacement weighted trajectories satisfying $d_k < \gamma_n$
25     $\gamma_n \leftarrow \rho$th quantile of $(d_1, \ldots, d_{K_n})$
26     Compute $\hat{\boldsymbol{\theta}}^{(n)}$ and $\hat{\boldsymbol{\sigma}}^{(n)}$ by means of Eqs. (3) and (4) (or (5)), using the
         elite trajectories satisfying $d_k < \gamma_n$ (and taking log appropriately in (3)
         and (4) )
27     Increment $n \leftarrow n + 1$
28     Adaptively update $K_n$
29 until convergence detected in $\{\gamma_1, \ldots, \gamma_{n-1}\}$
```

4 Experiments

We utilise our SPICE algorithm on four commonly investigated systems: (i) the Lotka-Volterra predator–prey model, (ii) a Yeast Polarization model, (iii) the bistable Schlögl system, and (iv) the Genetic Toggle Switch. We present results for each system obtained using both the standard SSA and optimised tau-leaping (with an error control parameter of $\varepsilon = 0.1$) to drive our simulations.

For each run of the algorithm we set the sample parameters $K_E = 10$, $K_{\min} = 1,000$, $K_{\max} = 20,000$, and set an upper limit on the number of iterations to 250. The smoothing parameters (λ, β, q) were set to $(0.7, 0.8, 5)$ respectively. For our analysis, we define the mean relative error (MRE) between a parameter estimate $\hat{\boldsymbol{\theta}}$ and the truth $\boldsymbol{\theta}^*$ as $\mathrm{MRE}(\%_{\mathrm{ERR}}) = M^{-1} \sum_j^M |\hat{\theta}_j - \theta_j^*|/\theta_j^* \times 100$. All our experiments were performed on a Intel Xeon 2.9GHz Linux system *without* using multiple cores — all reported CPU times are single-core. SPICE has been implemented in Julia and is open source (`https://github.com/pzuliani/SPICE`).

For models (i)–(iii), we use synthetic data where the true solution is known, and compare the results of SPICE against some commonly used parameter estimation techniques implemented in COPASI 4.16 [17]. Specifically, we check the performance of SPICE against the genetic algorithm (GA), evolution strategy (ES), evolutionary programming (EP), and particle swarm (PS) implementations. For the ES and EP algorithms we allow 250 generations with a population of 1,000 particles. For the GA, we run 500 generations with 2,000 particles. For the PS, we allow 1,000 iterations with 1,000 particles[1]. For model (iv), the Genetic Toggle Switch, we show results for SPICE using *real* experimental data.

All statistics presented are based on 100 runs of each algorithm using fixed datasets. For each approach we also compared the performance of using the standard SSA versus tau-leaping, alongside multiple-shooting and particle splitting approaches. However, for the models tested, neither multiple shooting nor particle splitting helped in reducing CPU times or improving the estimates accuracy.

Lotka-Volterra Predator–Prey Model We implement the standard Lotka-Volterra model below with real parameters $(\theta_1, \theta_2, \theta_3) = (0.5,\ 0.0025,\ 0.3)$, and initial population $(X_1, X_2) = (50,\ 50)$

$$X_1 \xrightarrow{\theta_1} X_1 + X_1 \qquad X_1 + X_2 \xrightarrow{\theta_2} X_2 + X_2 \qquad X_2 \xrightarrow{\theta_3} \emptyset$$

We artificially generated 5 datasets each consisting of 40 timepoints using Gillespie's SSA, and performed parameter estimation based on these datasets. For the initial iteration, we placed bounds on the Sobol sequence parameter search space of $\theta_j \in [1\mathrm{e}{-6}, 10]$, for $j = 1, 2, 3$. The minimum, maximum, and average MRE between the true parameters and their estimates across all 100 runs of each algorithm (using the standard SSA) are summarised in Table 1, together with corresponding CPU run times. Box plots summarising the obtained parameter estimates across all runs of each method are displayed in Fig. 1.

[1]NB: we also tested the COPASI implementations using greater populations and more iterations (not shown), but found little improvement for the significant increase in computational cost.

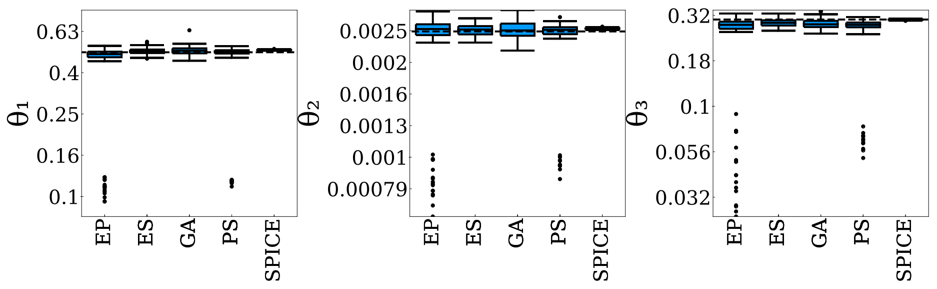


Fig. 1. Lotka-Volterra Model: box plots showing the summary statistics across 100 runs of COPASI and SPICE for each of the 3 parameter estimates. We note SPICE consistently has the least variance.

In the previous Lotka-Volterra predator–prey example, SPICE was provided with the complete data for both species X_1, X_2. However, we are also concerned with cases where the data is not fully observed, *i.e.*, when we have latent species. To compare the effects of latent species on the quality of parameter estimates, we ran SPICE again (averaging across 100 runs), this time supplying information about species X_1 alone. The results are presented in Table 1.

Yeast Polarization Model We implement the Yeast Polarization model (see below) with real parameters $(\theta_1, \ldots, \theta_8)$ = (0.38, 0.04, 0.082, 0.12, 0.021, 0.1, 0.005, 13.21), and initial population $(R, L, RL, G, G_a, G_{bg}, G_d)$ = (500, 4, 110, 300, 2, 20, 90). The reactions of the model are [8]:

$$\begin{aligned} \emptyset &\xrightarrow{\theta_1} R & RL + G &\xrightarrow{\theta_5} G_a + G_{bg} \\ R &\xrightarrow{\theta_2} \emptyset & G_a &\xrightarrow{\theta_6} G_d \\ L + R &\xrightarrow{\theta_3} RL + L & G_d + G_{bg} &\xrightarrow{\theta_7} G \\ RL &\xrightarrow{\theta_4} R & \emptyset &\xrightarrow{\theta_8} RL \end{aligned}$$

We artificially generated 5 datasets each consisting of 17 timepoints using Gillespie's SSA, and performed parameter estimation based on these datasets. For the initial iteration, we placed bounds on the parameter search space of $\theta_j \in [1\text{e}{-6}, 10]$ for $1 \leqslant j \leqslant 7$, and $\theta_8 \in [1\text{e}{-6}, 100]$. The average relative errors between the estimated and the real parameters across 100 runs of the algorithm are summarised in Table 1, along with the corresponding CPU run times. The variability of the estimates obtained using SPICE (and other methods) are shown in Fig 2.

Schlögl System We use the Schlögl model [30] with parameters $(\theta_1, \theta_2, \theta_3, \theta_4) = (3\text{e}{-7}, 1\text{e}{-4}, 1\text{e}{-3}, 3.5)$, and initial population $(X, A, B) = (250, 1\text{e}5, 2\text{e}5)$. This model is well known to produce bistable dynamics (see Fig. 4).

$$\begin{aligned} 2X + A &\xrightarrow{\theta_1} 3X & B &\xrightarrow{\theta_3} X \\ 3X &\xrightarrow{\theta_2} 2X + A & X &\xrightarrow{\theta_4} B \end{aligned}$$

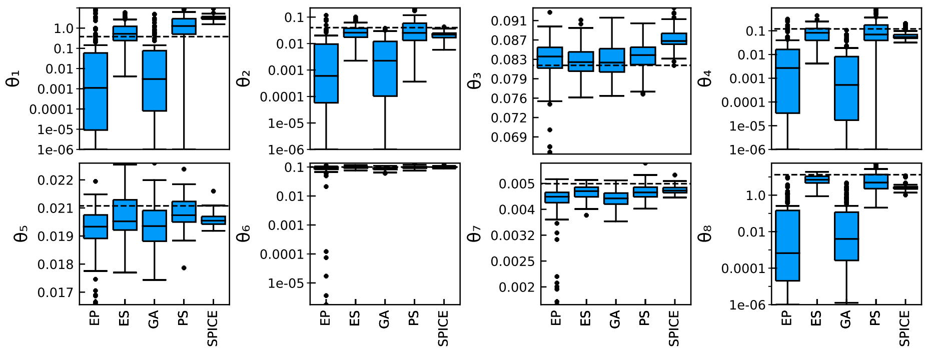


Fig. 2. Yeast Polarization parameter estimates: box plots showing the summary statistics of all 8 parameter estimates across 100 runs of COPASI's methods and SPICE. We note once again SPICE produces the least variation of obtained estimates.

We artificially generated 10 datasets (in order to partially capture a degree of the bistable dynamics) each consisting of 100 timepoints, and performed parameter estimation based on these datasets (also see Fig. 4). For the initial iteration, we placed bounds on the parameter search space of $\theta_1 \in [1e{-}9, 1e{-}5]$, $\theta_2 \in [1e{-}6, 0.01]$, $\theta_3 \in [1e{-}5, 10]$, $\theta_4 \in [0.01, 100]$. Unlike the previous models, we explicitly ran the Schlögl System using tau-leaping for *all* algorithms, due to the computation time being largely infeasible under the same conditions (4.5 hours in SPICE, 48+ hours in COPASI). The MRE of all the estimated parameters, together with CPU times for each algorithm are summarised in Table 1. Box plots of the SPICE algorithm's performance are presented in Fig. 3. Note that the Schlögl system is sensitive to the initial conditions, so even slight perturbations of its parameters can cause the system to fail in producing bimodality.

Fig. 3. Schlögl System parameter estimates: box plots comparing the parameter estimates across 100 runs of COPASI's methods and SPICE (all simulated using tau-leaping, $\varepsilon = 0.1$). Again, SPICE shows the smallest variance, with mean estimates quite close to the real values of θ_1 and θ_3. For θ_2 and θ_4, all the best mean estimates have variance much larger than SPICE estimates.

Toggle Switch Model The genetic toggle switch is a well studied bistable system, with particular importance toward synthetic biology. The toggle switch

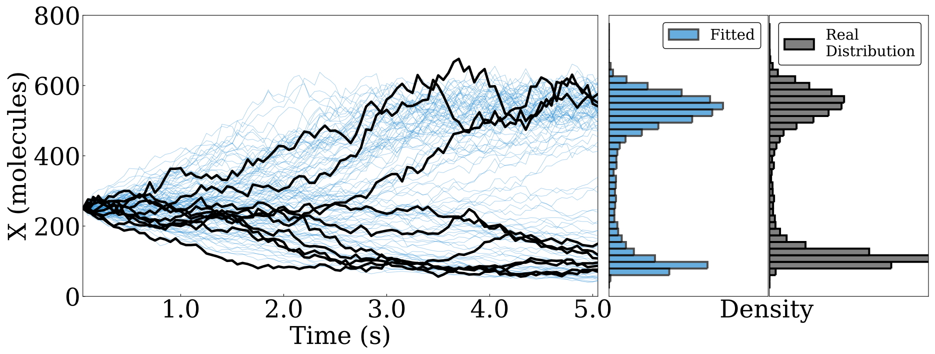


Fig. 4. Schlögl: From the left: Solid black lines: the 10 datasets generated using the SSA direct method and the real parameters, and used as input for SPICE. Blue lines: 100 model runs with estimated parameters sampled by the final parameter distributions obtained by SPICE with the direct method (means = $(2.14\text{e}-7, 7.63\text{e}-5, 4.54\text{e}-4, 2.18)$); variances = $(7.81\text{e}-16, 2.81\text{e}-10, 4.05\text{e}-8, 0.13)$). Fitted: empirical distribution of 1,000 model simulations with sampled parameters from SPICE output. Real distribution: empirical distribution of 1,000 model simulations with the real parameters.

is comprised of two repressors, and two promoters, often mediated in practice through IPTG[2] and aTc[3] induction. We perform parameter inference based on real high-throughput data (see Fig. 5), implemented upon a simple model (see below) based on [12]. For our model, we define the following reaction propensities:

$$h_1 = \theta_1 \times \text{GFP} \qquad h_3 = \theta_3 \times \text{MCHERRY}$$

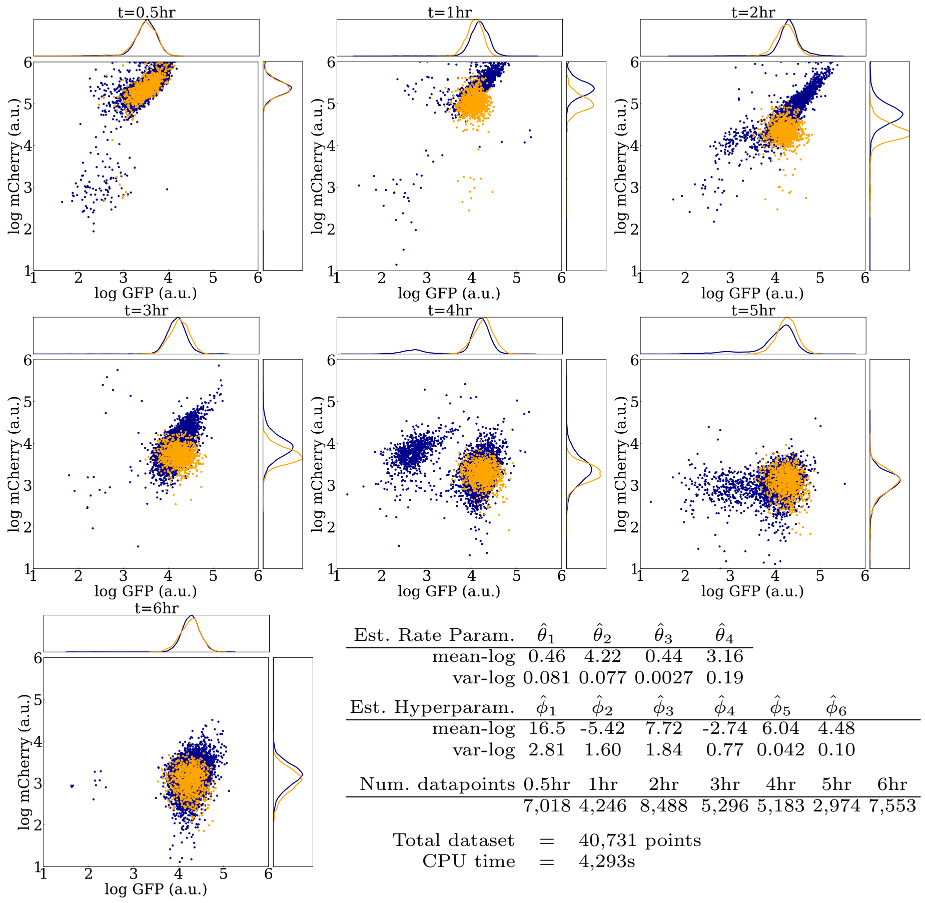
$$h_2 = \frac{\theta_2 \times \phi_1}{1 + \phi_1 + \phi_2 \times \text{MCHERRY}^2} \qquad h_4 = \frac{\theta_4 \times \phi_3}{1 + \phi_3 + \phi_4 \times \text{GFP}^2}$$

where GFP and mCherry are the two model species (reporter molecules), and the stochastic rate parameters are $(\theta_1, \ldots, \theta_4)$. The data used for parameter inference was obtained through fluorescent flow cytometry in [21], via the GFP and mCherry reporters, and consists 40,731 measurements across 7 timepoints over 6 hours. We look specifically at the case where the switch starts in the low-GFP (high mCherry) state, and switches to the high-GFP (low-mCherry) state over the time course after aTc induction to the cells. The inclusion of real, noisy data requires a degree of additional care as the data needs to be rescaled from arbitrary units (a.u.) to discrete molecular counts. We assume a linear (multiplicative) scale, *e.g.*, such that GFP (a.u.) = $\phi_5 \times$GFP *molecules*. Furthermore, we can no longer assume all the cells begin at the same state, and we must assume the initial state belongs to a distribution. This introduces extra so-called 'hyperparameters', specifically the GFP molecule count to fluorescent (a.u.) scale factor ϕ_5, and the respective mCherry scale factor ϕ_6. In addition,

[2] Isopropyl β-D-1-thiogalactopyranoside
[3] anhydrotetracycline

the model now contains 4 additional parameters, $\phi_1, \ldots, \phi_4$, which in turn are required to be estimated. Each hyperparameter is initially sampled as before using the low-discrepancy Sobol sequence, and updated using the means and variances of the generated elite samples as per the CE method.

The placed bounds on the initial kinetic parameter search space, based upon reported half-lives for the variants of GFP [2] and mCherry [31], were $\theta_{1,3} \in [1\text{e}{-3}, 1]$, and $\theta_{2,4} \in [1, 50]$. The respective bounds on the search space for the hyperparameters were $\phi_{1,2,3,4} \in [1\text{e}{-3}, 10]$, and $\phi_{5,6} \in [50, 500]$. To generate the parameter estimates, we used SPICE with tau-leaping ($\varepsilon = 0.1$, CPU time = 4,293s). The estimated parameters and the resulting fit against the data for the model can be seen in Fig. 5.



Est. Rate Param.	$\hat{\theta}_1$	$\hat{\theta}_2$	$\hat{\theta}_3$	$\hat{\theta}_4$
mean-log	0.46	4.22	0.44	3.16
var-log	0.081	0.077	0.0027	0.19

Est. Hyperparam.	$\hat{\phi}_1$	$\hat{\phi}_2$	$\hat{\phi}_3$	$\hat{\phi}_4$	$\hat{\phi}_5$	$\hat{\phi}_6$
mean-log	16.5	-5.42	7.72	-2.74	6.04	4.48
var-log	2.81	1.60	1.84	0.77	0.042	0.10

Num. datapoints	0.5hr	1hr	2hr	3hr	4hr	5hr	6hr
	7,018	4,246	8,488	5,296	5,183	2,974	7,553

Total dataset	=	40,731 points
CPU time	=	4,293s

Fig. 5. Toggle Switch Model: Blue circles: the experimental data with the $\log_{10}$(GFP) fluorescence plotted against the $\log_{10}$(mCherry) fluorescence, across all timepoints up to 6hr. Orange circles: 1,000 model simulations using the direct method, with parameters sampled from the final distribution obtained by SPICE using tau-leaping ($\varepsilon = 0.1$).

Table 1. The relative errors for each stochastic rate parameter averaged across 100 runs using COPASI's Evolutionary Programming (EP), Evolution Strategy (ES), Genetic Algorithm (GA), and Particle Swarm (PS) algorithms, and our SPICE algorithm are shown. The minimum, maximum, and average mean relative error (MRE) for all parameter estimates across all runs are also given alongside the averaged CPU time.

Lotka-Volterra Model							
Alg.	θ_1	θ_2 (%$_{\text{ERR}}$)	θ_3	Min. MRE (%$_{\text{ERR}}$)	Av. MRE (%$_{\text{ERR}}$)	Max. MRE (%$_{\text{ERR}}$)	Av. CPU (s)
EP	38.4	3.5	29.6	0.4	23.8	156.5	1200
ES	3.8	0.6	4.4	0.3	3.0	9.0	5763
GA	5.2	0.8	5.7	0.8	3.9	15.2	3640
PS	25.6	2.2	18.6	**0.1**	15.5	126.6	2689
SPICE	**3.6**	**0.4**	**0.4**	1.0	**1.5**	**2.1**	**1025**
Lotka-Volterra Latent-Species Model							
SPICE	9.4	0.41	6.4	4.1	5.4	6.8	1589

Yeast Polarization Model											
Alg.	θ_1	θ_2	θ_3	θ_4 (%$_{\text{ERR}}$)	θ_5	θ_6	θ_7	θ_8	Min. MRE (%$_{\text{ERR}}$)	Av. MRE (%$_{\text{ERR}}$)	Max. MRE (%$_{\text{ERR}}$)	Av. CPU (s)
EP	662.9	138.4	1.7	235.4	1.7	25.3	3.4	357.0	56.2	178.2	316.9	**405**
ES	**109.8**	**18.5**	**1.2**	35.4	1.3	3.3	1.5	**27.9**	**3.6**	**24.9**	62.8	1650
GA	564.0	120.2	1.3	275.3	1.6	6.5	2.6	312.4	38.8	160.5	299.4	2696
PS	156.4	29.0	1.4	52.6	**0.9**	3.7	1.6	48.6	7.3	36.8	173.6	1755
SPICE	221.2	21.7	2.5	**34.9**	**0.9**	**1.7**	**1.1**	62.7	27.6	43.3	**54.4**	1116

Schlögl System Model								
Alg.	θ_1	θ_2 (%$_{\text{ERR}}$)	θ_3	θ_4	Min. MRE (%$_{\text{ERR}}$)	Av. MRE (%$_{\text{ERR}}$)	Max. MRE (%$_{\text{ERR}}$)	Av. CPU (s)
EP	12.2	9.7	15.1	142.9	24.4	45.0	60.5	**307**
ES	**3.3**	15.5	19.0	**40.3**	**11.5**	**19.3**	31.7	1505
GA	13.7	11.0	14.0	159.7	32.2	49.6	66.3	987
PS	12.0	**8.5**	11.4	141.4	18.7	43.3	60.0	1095
SPICE	4.6	14.6	**6.3**	73.0	18.5	24.6	**30.9**	1054

5 Discussion

We can see from the presented results that our SPICE algorithm performs well on the models studied. For the Lotka-Volterra model the quality of the estimates is always good — there is no relative error larger than 2.1% in Table 1 for SPICE. The CPU times are reasonable in absolute terms (about 20 minutes, single core), and much smaller than those of the methods implemented in COPASI, and with smaller errors. Also, having one unobserved species (X_2) in the data does not seem to impact the results very much. In particular, from Table 1 we see that the latent model indeed has higher error than the fully observable model. However, the error is always smaller than 10%, which is acceptable.

The Yeast Polarization model is a more difficult system: we can indeed see from Table 1 that a number of parameter estimates have large relative errors. These are the same 'hard' parameters estimated by MCEM^2[8] with similar errors. However, in CPU time terms, our SPICE algorithm does much better than MCEM^2: SPICE can return a quite good estimate (in line with MCEM^2's) on average in about 18 minutes using the direct method, while MCEM^2 would need about 30 days [8] — a speed-up of 2,400 times. Furthermore, for this model one could use tau-leaping instead of the direct method, gaining a 3x speedup in performance while giving up little on accuracy (the Min., Av., and Max. MRE $\%_{\mathrm{ERR}}$ were 31.2, 41.5, and 56.3, respectively; Av. CPU time was 303s).

The Schlögl system is another challenging case study, as clearly showed by results of Table 1, which were obtained by utilising tau-leaping (as a matter of fact, for the Schlögl model the average accuracy of SPICE increases with the use of tau-leaping). Our choice was motivated by the large CPU time of the direct method due to the fact that the upper steady state for X in the model has a large molecule number (about 600), which negatively impacts the running time of the direct method samples. The results of Table 1 show that there is no clear winner: the Evolutionary Programming method in COPASI has the smallest runtime, but twice the error achieved by SPICE, which has the best accuracy. As noted before, running the COPASI implementations with larger populations and more iterations did not significantly improve accuracy for the increased cost.

Lastly, the genetic Toggle Switch presents an interesting real-world case study with high-throughput data. The model now comprises four hyperparameters, each of which must be estimated alongside the four kinetic rate constants. In addition, the non-discrete (and noisy) data is no longer known to be generated from a convenient mathematical model. In other terms, there is no guarantee that the model reflects the true underlying biochemical reaction network. Despite these challenges, our SPICE algorithm does a very good job (in *little more than an hour* of CPU time) in computing parameter estimates for which the model quite closely matches the experimental data — we see in fact from Fig. 5 that the model simulations fall inside the data, with very few exceptions, and the empirical and simulated distributions closely match.

Related Work Techniques for stochastic rate parameter estimation fall into four categories. Early efforts included methods based on MLE: simulated maximum likelihood utilises Monte Carlo simulation and a genetic algorithm to maximise an approximated likelihood [34]. Efforts have been made to incorporate the Expectation-Maximisation (EM) algorithm with the SSA [18]. The stochastic gradient descent explores a Markov Chain Monte Carlo sampler with a Metropolis-Hastings update step [39]. In [25] a hidden Markov model is used for the system state, which is then solved by (approximate) likelihood maximisation. Lastly, a recent work [8] has combined an ascent-based EM algorithm with a modified cross-entropy method. Another category of methodologies include Bayesian inference. In particular, approximate Bayesian computation (ABC) gains an advantage by becoming 'likelihood free', and recent advances in sequential Monte Carlo (SMC) samplers have further improved these methods

[32, 35]. We note the similarities between ABC(-SMC) approaches and SPICE. Both methods can utilize 'elite' samples to produce better parameter estimates. A key difference is that ABC(-SMC) uses accepted simulation parameters to construct a posterior distribution, while SPICE utilizes complete trajectory information to compute optimal updates of an underlying parameter distribution. The Bayesian approach presented in [5] can handle partially observed systems, including notions of experimental error. Linear noise approximation techniques have been used alongside Bayesian analysis [19]. A very recent work [36] combines Bayesian analysis with statistical emulation in an attempt at reducing the cost due to the SSA simulations. A third class of methodologies center around the numerical solution of the chemical master equation (CME), which is often intractable for all but the simplest of systems. One approach is to use dynamic state space truncation [3] or finite state projection methods [9] that truncate the CME state space by ignoring the smallest probability states. Another variation is to use a method of moments approximation [10, 16] to construct ordinary differential equations (ODEs) describing the time evolution for the mean, variance, *etc.*, of the underlying distribution. Other CME approximations are system size expansion using van Kampen's expansion [11], and solutions of the Fokker-Planck equation [22] using a form of linear noise approximation. Finally, another method [42] treats intervals between time measurements piecewise, and within each interval an ODE approximation is used for the objective function. This method has been recently extended using linear noise approximation [43]. A recent work [1], tailored for high-throughput data, proposes a stochastic parameter inference approach based on the comparison of distributions.

6 Conclusions

In this paper we have introduced the SPICE algorithm for rate parameter inference in stochastic reaction networks. Our algorithm is based on the cross-entropy method and Gillespie's algorithm, with a number of significant improvements. Key strengths of our algorithm are its ability to use multiple, possibly incomplete datasets (including distribution data), and its (theoretically justified) use of tau-leaping methods for model simulation. We have shown that SPICE works well in practice, in terms of both computational cost and estimate accuracy (which was often the best in the models tested), even on challenging case studies involving bistable systems and real high-throughput data. On a non-trivial case study, SPICE can be orders of magnitude faster than other approaches, while offering comparable accuracy in the estimates.

Acknowledgements This work has been supported by a BBSRC DTP PhD studentship and the EPSRC Portabolomics project (EP/N031962/1).

References

1. L. U. Aguilera, C. Zimmer, and U. Kummer. A new efficient approach to fit stochastic models on the basis of high-throughput experimental data using a model of IRF7 gene expression as case study. *BMC Systems Biology*, 11(1):26, 2017.

2. J. B. Andersen, C. Sternberg, L. K. Poulsen, S. P. Bjørn, M. Givskov, and S. Molin. New unstable variants of green fluorescent protein for studies of transient gene expression in bacteria. *Appl Environ Microbiol*, 64(6):2240–2246, Jun 1998.
3. A. Andreychenko, L. Mikeev, D. Spieler, and V. Wolf. Approximate maximum likelihood estimation for stochastic chemical kinetics. *EURASIP Journal on Bioinformatics and Systems Biology*, 2012(1):9, 2012.
4. W. J. Blake, M. KAErn, C. R. Cantor, and J. J. Collins. Noise in eukaryotic gene expression. *Nature*, 422(6932):633–637, 2003.
5. R. Boys, D. Wilkinson, and T. Kirkwood. Bayesian inference for a discretely observed stochastic kinetic model. *Statistics and Computing*, 18, 2008.
6. A. Costa, O. D. Jones, and D. Kroese. Convergence properties of the cross-entropy method for discrete optimization. *Operations Res. Lett.*, 35(5):573 – 580, 2007.
7. B. J. Daigle, M. K. Roh, D. T. Gillespie, and L. R. Petzold. Automated estimation of rare event probabilities in biochemical systems. *J. of Chem. Phys.*, 134(4), 2011.
8. B. J. Daigle, M. K. Roh, L. R. Petzold, and J. Niemi. Accelerated maximum likelihood parameter estimation for stochastic biochemical systems. *BMC Bioinformatics*, 13(1):68, 2012.
9. S. H. Dandach and M. Khammash. Analysis of stochastic strategies in bacterial competence: A master equation approach. *PLoS Comp. Bio.*, 6(11):1–11, 2010.
10. S. Engblom. Computing the moments of high dimensional solutions of the master equation. *Applied Mathematics and Computation*, 180(2):498 – 515, 2006.
11. F. Fröhlich, P. Thomas, A. Kazeroonian, F. J. Theis, R. Grima, and J. Hasenauer. Inference for stochastic chemical kinetics using moment equations and system size expansion. *PLoS Comp. Bio.*, 12(7):1–28, 2016.
12. T. S. Gardner, C. R. Cantor, and J. J. Collins. Construction of a genetic toggle switch in *Escherichia coli*. *Nature*, 403(6767):339–342, 2000.
13. D. T. Gillespie. A general method for numerically simulating the stochastic time evolution of coupled chemical reactions. *J. of Comp. Phys.*, 22(4):403–434, 1976.
14. D. T. Gillespie. Exact stochastic simulation of coupled chemical reactions. *J. of Phys. Chem.*, 81(25):2340–2361, 1977.
15. D. T. Gillespie. Approximate accelerated stochastic simulation of chemically reacting systems. *Journal of Chemical Physics*, 115(4):1716–1733, 2001.
16. J. Hasenauer, V. Wolf, A. Kazeroonian, and F. J. Theis. Method of conditional moments (MCM) for the Chemical Master Equation. *Journal of Mathematical Biology*, 69(3):687–735, 2014.
17. S. Hoops, S. Sahle, R. Gauges, C. Lee, J. Pahle, N. Simus, M. Singhal, L. Xu, P. Mendes, and U. Kummer. COPASI - A Complex PAthway SImulator. *Bioinformatics*, 22(24):3067–3074, 2006.
18. A. Horváth and D. Martini. Parameter estimation of kinetic rates in stochastic reaction networks by the EM method. In *BMEI*, pages 713–717. IEEE, 2008.
19. M. Komorowski, B. Finkenstädt, C. V. Harper, and D. A. Rand. Bayesian inference of biochemical kinetic parameters using the linear noise approximation. *BMC Bioinformatics*, 10(1):343, 2009.
20. S. Kullback and R. A. Leibler. On information and sufficiency. *Ann. Math. Statist.*, 22(1):79–86, 1951.
21. M. Leon. *Computational design and characterisation of synthetic genetic switches*. PhD thesis, University College London, UK, 2017. Available at `http://discovery.ucl.ac.uk/1546318/1/Leon_Miriam_thesis_final.pdf`.
22. S. Liao, T. Vejchodský, and R. Erban. Tensor methods for parameter estimation and bifurcation analysis of stochastic reaction networks. *Interface, Journal of the Royal Society*, 12(108):20150233, 2015.

23. H. H. McAdams and A. Arkin. Stochastic mechanisms in gene expression. *PNAS*, 94(3):814–819, 1997.
24. J. R. Pirone and T. C. Elston. Fluctuations in transcription factor binding can explain the graded and binary responses observed in inducible gene expression. *Journal of Theoretical Biology*, 226(1):111–112, 2004.
25. S. Reinker, R. M. Altman, and J. Timmer. Parameter estimation in stochastic biochemical reactions. *IEE Proceedings - Systems Biology*, 153(4):168–178, 2006.
26. C. P. Robert and G. Casella. *Monte Carlo Statistical Methods.* Springer, 2004.
27. R. Y. Rubinstein. Optimization of computer simulation models with rare events. *European Journal of Operational Research*, 99(1):89–112, 1997.
28. R. Y. Rubinstein. The cross-entropy method for combinatorial and continuous optimization. *Methodol. and Comp. in Appl. Prob.*, 1(2):127–190, 1999.
29. R. Y. Rubinstein and D. P. Kroese. *The Cross-Entropy Method.* Springer, 2004.
30. F. Schlögl. Chemical reaction models for non-equilibrium phase transitions. *Zeitschrift für physik*, 253(2):147–161, 1972.
31. N. C. Shaner, R. E. Campbell, P. A. Steinbach, B. N. G. Giepmans, A. E. Palmer, and R. Y. Tsien. Improved monomeric red, orange and yellow fluorescent proteins derived from *Discosoma* sp. red fluorescent protein. *Nature Biotechnology*, 22:1567-1572, 2004.
32. S. A. Sisson, Y. Fan, and M. M. Tanaka. Sequential Monte Carlo without likelihoods. *PNAS*, 104(6):1760–5, 2007.
33. I. M. Sobol'. On the distribution of points in a cube and the approximate evaluation of integrals. *USSR Comput. Math. and Math. Phys.*, 7(4):86 – 112, 1967.
34. T. Tian, S. Xu, J. Gao, and K. Burrage. Simulated maximum likelihood method for estimating kinetic rates in gene expression. *Bioinformatics*, 23(1):84–91, 2007.
35. T. Toni, D. Welch, N. Strelkowa, A. Ipsen, and M. P. H. Stumpf. Approximate Bayesian computation scheme for parameter inference and model selection in dynamical systems. *Interface, Journal of the Royal Society*, 6(31):187–202, 2009.
36. I. Vernon, J. Liu, M. Goldstein, J. Rowe, J. Topping, and K. Lindsey. Bayesian uncertainty analysis for complex systems biology models: emulation, global parameter searches and evaluation of gene functions. *BMC Sys. Biol.*, 12(1):1, 2018.
37. A. F. Villaverde and J. R. Banga. Reverse engineering and identification in systems biology: strategies, perspectives and challenges. *Interface, Journal of The Royal Society*, 11(91):20130505, 2013.
38. E. O. Voit. The best models of metabolism. *Wiley Interdisciplinary Reviews: Systems Biology and Medicine*, 9(6):e1391, 2017.
39. Y. Wang, S. Christley, E. Mjolsness, and X. Xie. Parameter inference for discretely observed stochastic kinetic models using stochastic gradient descent. *BMC Systems Biology*, 4(1):99, 2010.
40. D. J. Wilkinson. Stochastic modelling for quantitative description of heterogeneous biological systems. *Nature Reviews Genetics*, 10(2):122–133, 2009.
41. D. J. Wilkinson. *Stochastic Modelling for Systems Biology.* CRC Press, 2012.
42. C. Zimmer and S. Sahle. Parameter estimation for stochastic models of biochemical reactions. *Journal of Computer Science & Systems Biology*, 6(1):11–21, 2012.
43. C. Zimmer and S. Sahle. Deterministic inference for stochastic systems using multiple shooting and a linear noise approximation for the transition probabilities. *IET Systems Biology*, 9:181–192, 2015.