BMC Systems Biology

RESEARCH ARTICLE Open Access

Stochastic parameter search for events

Min K Roh* and Philip Eckhoff

Abstract

Background: With recent increase in affordability and accessibility of high-performance computing (HPC), the use of large stochastic models has become increasingly popular for its ability to accurately mimic the behavior of the represented biochemical system. One important application of such models is to predict parameter configurations that yield an event of scientific significance. Due to the high computational requirements of Monte Carlo simulations and dimensionality of parameter space, brute force search is computationally infeasible for most large models.

Results: We have developed a novel parameter estimation algorithm—Stochastic Parameter Search for Events (SParSE)—that automatically computes parameter configurations for propagating the system to produce an event of interest at a user-specified success rate and error tolerance. Our method is highly automated and parallelizable. In addition, computational complexity does not scale linearly with the number of unknown parameters; all reaction rate parameters are updated concurrently at the end of each iteration in SParSE. We apply SParSE to three systems of increasing complexity: birth-death, reversible isomerization, and Susceptible-Infectious-Recovered-Susceptible (SIRS) disease transmission. Our results demonstrate that SParSE substantially accelerates computation of the parametric solution hyperplane compared to uniform random search. We also show that the novel heuristic for handling over-perturbing parameter sets enables SParSE to compute biasing parameters for a class of rare events that is not amenable to current algorithms that are based on importance sampling.

Conclusions: SParSE provides a novel, efficient, event-oriented parameter estimation method for computing parametric configurations that can be readily applied to any stochastic systems obeying chemical master equation (CME). Its usability and utility do not diminish with large systems as the algorithmic complexity for a given system is independent of the number of unknown reaction rate parameters.

Keywords: Stochastic simulation, Parameter estimation, Rare event, Optimization

Background

Stochastic modeling of biochemical and ecological systems has become increasingly popular due to its ability to represent system dynamics correctly at a detailed level, especially when species are present at low population. Deterministic models, on the other hand, are easier to analyze, yet they may fail to capture even the average behavior when the represented system exhibits nonlinearity [1] or is near extinction. Recent advancements in cloud computing platforms [2,3] and GPU computing [4-7] have significantly increased the affordability of computational resources. This enables development and use of stochastic algorithms that would have been deemed computationally infeasible in the past. However, there is still a void in stochastic methods that can answer scientifically interesting questions. One such application is in determining reaction rate configurations that yield an event of interest with a set success probability. Most parameter estimation algorithms in stochastic chemical kinetics setting take time-series data as an input and compute a set of reaction rate parameters that most closely reproduce the data. Methods used to determine these reaction rate parameters include maximum likelihood ratio [8-10], gradient decent [11], and moment closure [12]. While these algorithms are useful in its own right, scientists are often interested in knowing all parameter combinations that yield a specific event of interest. For gene regulatory models, knowledge of all pathways to achieve a specific event, such as bistable transition of *lac* operon in *E. coli* [13-15], may be used to guide laboratory experiments. In epidemiological models, all intervention

*Correspondence: mroh@intven.com
Numerical Methods, Institute for Disease Modeling, 1575 132 Ave. NE, 98005 Bellevue, USA

parameter combinations that achieve eradication can be combined with econometrics in computing the most cost-effective strategy for eradicating a disease [16]. To authors' knowledge, no algorithm has been developed in stochastic chemical kinetics setting that computes such parameter combinations.

In this paper, we present Stochastic Parameter Search for Events (SParSE) that finds a parametric hyperplane of reaction rates conferring a user-specified event with prescribed success rate and error tolerance. Our algorithm is robust in that it accurately computes the solution hyperplane for low probability events as well as high probability events. It is also trivial to parallelize the algorithm; initial parameter sets do not need to communicate with each other to find the direction to the unknown solution hyperplane. Once the algorithm finds a point in the solution hyperplane, the ratio between the initial and final rates can be used as the biasing parameters by the doubly weighted stochastic simulation algorithm (dwSSA) [17] to compute the probability of observing the target event with its success rate under the original system description. This allows calculation of the target event probabilities under the original parameters as a powerful side benefit of the algorithm. Lastly, the SParSE runtime per parameter sample is of the same order as that of the stochastic simulation algorithm (SSA), *i.e.*, the algorithm complexity is independent of the number of unknown parameters for a given system. This is achieved by combining a novel modification of dwSSA [17], Rubinstein's cross-entropy method [18], and exponential interpolation of biasing parameters. This feature provides substantial benefits when searching multi-dimensional parameter space.

Methods

Doubly weighted stochastic simulation

We begin with a brief review of the dwSSA; detailed derivation and applications can be found in Daigle *et al.* [17]. Throughout this paper we assume a well-stirred system at constant temperature with N species $S_1, \ldots, S_N$ and M reactions $R_1, \ldots, R_M$. The state of the system at time t is represented as $\mathbf{X}(t) = [X_1, \ldots, X_N]$, where X_i is the population of species S_i. Using the "direct method" implementation of Gillespie's Stochastic Simulation Algorithm (SSA) [19], the system moves forward in time by sequentially firing one of R_j, $j \in \{1, \ldots, M\}$ reactions, whose propensity at time t is $a_j(\mathbf{X}(t))$ and its sum $a_0 = \sum_{j=1}^{M} a_j(\mathbf{X}(t))$. Here, the next reaction is chosen with a categorical random variable j' and the time to the next reaction with an exponential random variable τ. We also assume that all trajectories are run until the smaller of the final simulation time t_f and the first time $\mathcal{E}$ is observed, where $\mathcal{E}$ is the event of interest. Denoting the stopping time of a trajectory as $\mathcal{T}$, the probability of a complete trajectory $J = \left(t_1, j'_1, \ldots, t_{N_\mathcal{T}}, j'_{N_\mathcal{T}}\right)$ given $\mathbf{X}(0) = \mathbf{x}_0$ under SSA is as follows:

$$\begin{aligned} \mathrm{P}_{SSA}(\mathbf{J}) &= \prod_{i=1}^{N_\mathcal{T}} \left[a_0(\mathbf{X}(t_i)) e^{-a_0(\mathbf{X}(t_i))\tau_i} d\tau_i \times \frac{a_{j'_i}(\mathbf{X}(t_i))}{a_0(\mathbf{X}(t_i))} \right] \\ &= \prod_{i=1}^{N_\mathcal{T}} \left[a_{j'_i}(\mathbf{X}(t_i)) e^{-a_0(\mathbf{X}(t_i))\tau_i} d\tau_i \right], \end{aligned} \tag{1}$$

with $t_i \equiv \sum_{j=1}^{i} \tau_j$ and $N_\mathcal{T}$ the total number of reactions fired in $[0, \mathcal{T}]$.

The dwSSA uses predilection functions to increase the number of trajectories that reach $\mathcal{E}$:

$$b_j(\mathbf{X}(t)) \equiv \gamma_j a_j(\mathbf{X}(t)), \quad b_0(\mathbf{X}(t)) = \sum_{j=1}^{M} b_j(\mathbf{X}(t)), \tag{2}$$

where $\gamma_j \in \mathbb{R}^+$ is a biasing parameter for R_j. The probability of the same trajectory $\mathbf{J}$ under the dwSSA is then given by

$$\begin{aligned} \mathrm{P}_{dwSSA}(\mathbf{J}) &= \prod_{i=1}^{N_\mathcal{T}} \left[b_0(\mathbf{X}(t_i)) e^{-b_0(\mathbf{X}(t_i))\tau_i} d\tau_i \times \frac{b_{j'_i}(\mathbf{X}(t_i))}{b_0(\mathbf{X}(t_i))} \right] \\ &= \prod_{i=1}^{N_\mathcal{T}} \left[b_{j'_i}(\mathbf{X}(t_i)) e^{-b_0(\mathbf{X}(t_i))\tau_i} d\tau_i \right], \end{aligned} \tag{3}$$

and the bias incurred by using the predilection function is corrected with

$$\begin{aligned} W_{dwSSA}(\mathbf{J}) &= \prod_{i=1}^{N_\mathcal{T}} \left[\frac{a_{j'_i}(\mathbf{X}(t_i)) e^{-a_0(\mathbf{X}(t_i))\tau_i}}{b_{j'_i}(\mathbf{X}(t_i)) e^{-b_0(\mathbf{X}(t_i))\tau_i}} \right] \\ &= \prod_{i=1}^{N_\mathcal{T}} \left[\exp\left\{ \left(b_0(\mathbf{X}(t_i)) - a_0(\mathbf{X}(t_i)) \right) \tau_i \right\} \times \left(\gamma_{j'_i}\right)^{-1} \right]. \end{aligned} \tag{4}$$

It is straightforward to confirm that the product of (3) and (4) equals (1).

The Monte Carlo estimator for $p_{dwSSA}(\mathbf{x}_0, \mathcal{E}; t)$, the probability that the system reaches $\mathcal{E}$ by time t given the initial state $\mathbf{x}_0$, is

$$\hat{p}_{dwSSA}(\mathbf{x}_0, \mathcal{E}; t) = \frac{1}{N} \sum_{i=1}^{N} \left[I_{\{\mathbf{J}_i \cap \mathcal{E}\}} W_{dwSSA}(\mathbf{J}_i) \right], \tag{5}$$

where N is the total number of trajectories, $\mathbf{J}_i$ represents the i^{th} simulated dwSSA trajectory, and $I_{\{\mathbf{J}_i \cap \mathcal{E}\}}$ takes a value of 1 if $\mathcal{E}$ is visited by $\mathbf{J}_i$ and 0 otherwise. The quantity in (5) can be interpreted as the weighted average of successful trajectories, *i.e.*, trajectories reaching $\mathcal{E}$, where the weight is computed according to (4). A good set of biasing parameters would yield successful

trajectories with weights close to the true probability and thus reduce variance in the probability estimator. The dwSSA computes low variance biasing parameters by minimizing cross entropy using a modified version of Rubinstein's multilevel cross-entropy method [17,18]. The advantage of minimizing cross entropy over minimizing variance is that the former yields biasing parameters with a closed-form solution for (2) where the latter does not. Having a closed-form solution is of practical necessity, as the alternative would be to solve a large set of nonlinear equations, significantly decreasing the efficiency of the algorithm if not making the simulation infeasible.

Following derivations presented in Daigle *et al.* [17], the dwSSA biasing parameter for R_j is computed as

$$\hat{\gamma}_j^{(l)} = \frac{\sum_i' \left(W_{dwSSA}\left(\mathbf{J}_i^{(l-1)}; \hat{\boldsymbol{\gamma}}^{(l-1)}\right) \times n_{ij} \right)}{\sum_i' \left(W_{dwSSA}\left(\mathbf{J}_i^{(l-1)}; \hat{\boldsymbol{\gamma}}^{(l-1)}\right) \times \sum_{k=1}^{N\tau_k} \left[a_j\left(\mathbf{X}_i^{(l-1)}(t_{ik})\right)\tau_{ik} \right] \right)}, \tag{6}$$

where n_{ij} is the total number of times reaction j fires in the i^{th} trajectory, $\sum_i'$ iterates only over trajectories reaching $\mathcal{E}$, and l is the stage index in multilevel cross-entropy method. Computation for $\hat{\boldsymbol{\gamma}}^{(l)}$ terminates when intermediate rare event reaches $\mathcal{E}$, at which point we set $\hat{\boldsymbol{\gamma}}^{(l)} \equiv \hat{\boldsymbol{\gamma}}^*$.

The objective of the dwSSA necessitates computation of the likelihood ratio (4) as its probability estimator is with respect to the initial reaction rates $\boldsymbol{k}^{(0)}$. On the other hand, the objective of SParSE is to compute a set of reaction rates $\boldsymbol{k}^* \in \mathbb{R}^{M+}$ such that

$$\left| \mathcal{P}_{\mathcal{E}} - \frac{1}{N} \sum_{i=1}^{N} \left[I_{\{f_i(\mathbf{x}(t|\boldsymbol{k}^*))\cap\mathcal{E}\}} \right] \right| \leq \epsilon_{\mathcal{P}_{\mathcal{E}}}, \tag{7}$$

where $\mathcal{P}_{\mathcal{E}}$ is the desired probability of observing $\mathcal{E}$ by time t, $I_{\{f_i(\mathbf{x}(t|\boldsymbol{k}^*))\cap\mathcal{E}\}}$ an indicator function for observing $\mathcal{E}$ during i^{th} trajectory, and $\epsilon_{\mathcal{P}_{\mathcal{E}}}$ a user-specified absolute error tolerance on $\mathcal{P}_{\mathcal{E}}$. Unlike the dwSSA where the biasing parameters are updated each level of the multilevel cross-entropy method according to (6), in SParSE reaction rates are updated instead. We note that it is possible to use the dwSSA Monte Carlo estimator (5) in (7) and update $\boldsymbol{\gamma}^{(l)}$ instead of $\boldsymbol{k}^{(l)}$. However unless $\boldsymbol{k}^{(0)}$ is sufficiently close to $\boldsymbol{k}^*$, the likelihood ratio (4) may become extremely small, *i.e.*, degenerate, and updating reaction rates avoids this problem. We discuss the criteria for updating $\boldsymbol{k}$ in the following section.

Multilevel cross-entropy method

The modification of multilevel cross-entropy method in SParSE is similar to that of Ref [17]. However, there are three major differences between the multilevel cross-entropy method employed by dwSSA and by SParSE: (i) dwSSA only computes a single intermediate event $\xi^{(l)}$ and the corresponding set of biasing parameters $\boldsymbol{\gamma}^{(l)}$ while SParSE may compute multiple such quantities, (ii) SParSE can calculate biasing parameters for initial reaction rates that either over- or under-perturb the system with respect to $\mathcal{P}_{\mathcal{E}}$. For an over-perturbed system, it applies inverse biasing to reaction rates to convert $\mathcal{E}$ from a "sure event" to a "rarer event", and (iii) dwSSA updates biasing parameters $\boldsymbol{\gamma}^{(l)}$ while SParSE updates reaction rates $\boldsymbol{k}^{(l)}$. The following subsection explains the first two differences and highlights how SParSE achieves the same time complexity as dwSSA for computing quantities in (i). The next subsection focuses on (iii) and its effect on simulation details.

Concurrent computation of multiple intermediate events and biasing parameters

In dwSSA, N trajectories in level l of multilevel cross-entropy method are run until either the final simulation time t_{final} or the first occurrence of $\xi^{(l)}$, where $\xi^{(l)}$ is an intermediate rare event chosen by the top $\lceil \rho N \rceil$ trajectories that evolve farthest in the direction of $\mathcal{E}$. Typical value of ρ used in dwSSA is 0.01 for $N = 10^5$, although any value $\rho \in (0, 1)$ can be used in theory [17]. The role of ρ can be thought as a knob that controls the tradeoff between the speed of convergence to $\mathcal{E}$ and accuracy in $\hat{\boldsymbol{\gamma}}^*$. For $\rho' < \rho$, we get $|\mathcal{E} - \xi^{(l')}| \leq |\mathcal{E} - \xi^{(l)}|$, thus smaller values for ρ can potentially drive the system toward $\mathcal{E}$ faster. However the number of trajectories reaching $\xi^{(l')}$ is less than the number of trajectories reaching $\xi^{(l)}$ since $\lceil \rho' N \rceil < \lceil \rho N \rceil$. Having fewer data to compute $\hat{\boldsymbol{\gamma}}^{(l)}$ reduces the confidence on the estimate, therefore it is advised to keep $\lceil \rho N \rceil$ above a set threshold (*e.g.*, 200) in practice. On the other hand, larger value of ρ (*e.g.*, $\rho > 0.3$) implies a less selective intermediate rare event. The resulting biasing parameters may not push the system closer to $\mathcal{E}$, causing a failure in convergence to the target event. In our experience, $\rho < 0.2$ and $\lceil \rho N \rceil > 100$ yield both reliable computation of biasing parameters and acceptable convergence to $\mathcal{E}$.

In order to determine $\xi^{(l)}$, it is necessary to determine the direction of bias in addition to ρ. This is done by grouping the initial state $\mathbf{x}_0$ into two categories according to its distance with respect to the event of interest:

$$\phi_{\text{type}} = \begin{cases} 1 & \text{if } f\ (\mathbf{x}\,(t_0)) \leq \mathcal{E} \\ -1 & \text{otherwise,} \end{cases} \tag{8}$$

where $f\,(\mathbf{x}\,(t))$ is an event function. Two requirements for $f\,(\mathbf{x}\,(t))$ are that it takes $\mathbf{x}(t)$ as an input and can be used to evaluate the distance between the current state and $\mathcal{E}$ (*i.e.*, it can be used to compute extreme values of each

trajectory to determine the next intermediate event). The value of ϕ_{type} indicates *initial* position of $\mathbf{x}(t)$ with respect to $\mathcal{E}$ at $t = 0$. When ϕ_{type} is equal to 1, maximum value of $f(\mathbf{x}(t))$ in each trajectory is recorded, and N such values are sorted in descending order at the end of the simulation. The reasoning for this is that since $f(\mathbf{x}(t_0)) \leq \mathcal{E}$, we need to encourage higher $f(\mathbf{x}(t))$ values to get closer to $\mathcal{E}$. Similarly, minimum $f(\mathbf{x}(t))$ values are recorded and sorted in ascending order when ϕ_{type} is -1. For convenience we refer to the sorted array of extreme $f(\mathbf{x}(t))$ values as $\mathbf{v}_{\boldsymbol{k}}$, where $\boldsymbol{k}$ is the reaction rates used to generate $\mathbf{x}(t)$.

We now define a SParSE probability estimator for $\mathcal{E}$:

$$\hat{p}_{SParSE}(\mathbf{x}_0, \boldsymbol{k}, \mathcal{E}; t) = \frac{1}{N}\sum_{i=1}^{N}\left[I_{\{f_i(\mathbf{x}(t|\boldsymbol{k}))\cap\mathcal{E}\}}\right], \tag{9}$$

where $f_i(\mathbf{x}(t|\boldsymbol{k}))$ are event function evaluated with i^{th} SSA trajectory generated with reaction rates $\boldsymbol{k}$ and $I_{\{f_i(\mathbf{x}(t|\boldsymbol{k}))\cap\mathcal{E}\}}$ takes a value of 1 if $f_i(\mathbf{x}(t|\boldsymbol{k}))$ contains $\mathcal{E}$ and 0 otherwise. Once N trajectories are simulated, we can expect one of the following outcomes: (a) the inequality in (7) is satisfied, (b) $\hat{p}_{SParSE}(\mathbf{x}_0, \boldsymbol{k}, \mathcal{E}; t) < \left(\mathcal{P}_{\mathcal{E}} - \epsilon_{\mathcal{P}_{\mathcal{E}}}\right)$, or (c) $\left(\mathcal{P}_{\mathcal{E}} + \epsilon_{\mathcal{P}_{\mathcal{E}}}\right) < \hat{p}_{SParSE}(\mathbf{x}_0, \boldsymbol{k}, \mathcal{E}; t)$.

In the first case, SParSE exits and returns $\boldsymbol{k}$ as a successful output, i.e., a point in the solution hyperplane ($\boldsymbol{k} \equiv \boldsymbol{k}^*$). In the second case, we need to choose extreme values of $f(\mathbf{x}(t))$ evolving furthest to $\mathcal{E}$, and we can view $\mathcal{E}$ as a "rare event" as in the dwSSA. Thus intermediate events and its respective biasing parameters are computed iteratively, each time taking the system closer to $\mathcal{E}$ with success rate $\mathcal{P}_{\mathcal{E}}$. The last case corresponds to parameter sets that "over-perturb" the system, as $\mathcal{E}$ was reached with probability greater than $\mathcal{P}_{\mathcal{E}}$. The method used to determine an intermediate event in the classical multilevel cross-entropy method cannot be applied here because we do not want trajectories that produce extreme values of $f(\mathbf{x}(t))$. However, the information gathered from such trajectories can be used to quantify the behavior we *do not* want to observe. We achieve this by collecting the extreme values of $f(\mathbf{x}(t))$ as in case (b), except that each SSA simulation is run until the final simulation time without stopping when $\mathcal{E}$ is observed. Once intermediate events are chosen and their corresponding biasing parameters are computed, we update j^{th} reaction rate k_j with $1/\gamma_j$. This inverse biasing discourages over-perturbation with respect to $\mathcal{P}_{\mathcal{E}}$. Algorithms 2 and 3 in Appendix B of Additional file 1 contain pseudocode for (b) and (c), respectively.

Unlike the multilevel cross-entropy method used by dwSSA, where only one intermediate event is computed in each level of multilevel CE method, SParSE may choose multiple intermediate events. While it is not necessary to compute multiple intermediate events to reach the solution hyperplane, doing so greatly improves algorithm efficiency. The caveat here is that the efficiency gain occurs only when the biasing parameters for the multiple intermediate events are computed simultaneously. We start by describing the method SParSE uses to choose multiple intermediate events, all of which can be reached by N trajectories with sufficient frequency. This is attained by choosing multiple values for ρ that is a function of the distance to the desired target event probability $\mathcal{P}_{\mathcal{E}}$. Denoting the distance as $\delta(\boldsymbol{k}) \equiv \mathcal{P}_{\mathcal{E}} - \hat{p}_{SParSE}(\boldsymbol{k})$, we have two different methods for choosing $\boldsymbol{\rho}(\delta)$: one for case (b) and the other for case (c). Handling of the two cases differ, as the result of inverse biasing is not as obvious as the normal biasing for case (b) is. In normal biasing strategy, updating intermediate reaction rates with a particular set of biasing parameters redistributes $\mathbf{v}_{\boldsymbol{k}}$ such that the corresponding intermediate event becomes the mode. However, the inverse biasing operates on the heuristics of discouraging over-perturbation without knowing the exact effect on $\mathbf{v}_{\boldsymbol{k}}$. Thus more conservative values for $\boldsymbol{\rho}$ are used in (11) to compensate for this difference. Lastly, we note that each of these cases can be detected by comparing the sign of δ to the value of ϕ_{type}, where the equality represents case (b).

For $\text{sgn}(\delta(\boldsymbol{k})) == \phi_{\text{type}}$

$$\boldsymbol{\rho}(\delta) = \begin{cases} [0.005\ 0.01] & \text{if } 0.4 < |\delta| \\ [0.01\ 0.05\ 0.1] & \text{if } 0.2 < |\delta| \leq 0.4 \\ [0.05\ 0.1\ 0.2] & \text{otherwise} \end{cases} \tag{10}$$

For $\text{sgn}(\delta(\boldsymbol{k})) \neq \phi_{\text{type}}$

$$\boldsymbol{\rho}(\delta) = \begin{cases} [0.01\ 0.015] & \text{if } 0.4 < |\delta| \\ [0.05\ 0.1\ 0.15] & \text{if } 0.2 < |\delta| \leq 0.4 \\ [0.1\ 0.15\ 0.2] & \text{otherwise} \end{cases} \tag{11}$$

As the distance to the target event decreases, SParSE selects less extreme values for intermediate events and vice versa. This reduces the risk of over- and under-perturbations. We note that the number of elements in $\boldsymbol{\rho}(\delta)$ does not necessarily correspond to the number of intermediate events chosen. For example, elements corresponding to positions $\lceil 0.005 * N \rceil$ and $\lceil 0.01 * N \rceil$ of $\mathbf{v}_{\boldsymbol{k}}$ may be the same. We also note that a custom function can be used to compute $\boldsymbol{\rho}$ to better suit a specific system. However, the above default values work well for all examples presented in this paper. Lastly, N can be chosen as a function of $\min(\boldsymbol{\rho})$ and c, where c is the minimum number of data points desired to reliably compute $\boldsymbol{\gamma}^{(l)}$, *i.e.*, $N \geq c/\min(\boldsymbol{\rho})$.

Once intermediate events are computed, they are sorted in ascending order of its probability, *i.e.*, $Prob(\xi^{(l,1)}) \leq \cdots \leq Prob(\xi^{(l,q)})$, where q is the number of unique intermediate events chosen at level l. We note that this sorting is done automatically if elements of $\boldsymbol{\rho}$ are sorted in ascending order, which (10, 11) are.

Now we describe how biasing parameters for all intermediate events are computed concurrently in a single ensemble of N simulations. In each simulation, we check for $\xi^{(l,q)}$. If $\xi^{(l,q)}$ is observed, the statistics gathered up to the time at which $\xi^{(l,q)}$ was reached are used to compute $\boldsymbol{\gamma}^{(l,q)}$. Then the trajectory continues its course of simulation, this time checking for $\xi^{(l,q-1)}$ while keeping the cumulative statistics. This process repeats until the smaller of t_{final} and the time at which $\xi^{(l,1)}$ is observed (*i.e.*, all intermediate events are observed). When $q == 1$, this method is identical to the one used by dwSSA. Although a single trajectory runtime for $q > 1$ is slightly longer than the runtime for $q == 1$, the additional resources spent on concurrent computation is negligible compared to the savings of $(q-1)\cdot N$ simulations. We note that this process yields biasing parameter sets that are correlated because $\boldsymbol{\gamma}^{(l,i)}$ is computed with a subset of data used to compute $\boldsymbol{\gamma}^{(l,i+1)}$. However, this correlation does not affect the validity or the accuracy of the final output as only one set is selected at each level to update the reaction rates, the process for which we explain in the next section.

Updating intermediate reaction rates

SParSE propagates the system towards the solution hyperplane by iteratively updating reaction rates during the modified multilevel cross-entropy method. The update process requires choosing one set of biasing parameters from possibly many sets, where the set size is determined by the number of unique intermediate events. The current intermediate reaction rates are then multiplied elementwise by the chosen set to produce the next intermediate reaction rates. The criterion SParSE adopts is straightforward; at level l it chooses the biasing parameter set that, when multiplied to the current intermediate reaction rates, takes the system closest to $\mathcal{E}$ while preserving the sign of $\delta(\boldsymbol{k}^{(g)}), g = 0, \cdots, l$.

Without loss of generality, we define $\boldsymbol{k}^{(\mathrm{cur})}$ as the intermediate reaction rates at an arbitrary level l. In order to update the intermediate reaction rates for the next stage, we evaluate how each candidate biasing parameter set $\boldsymbol{\gamma}^{(l,\cdot)}$ performs with respect to the update criterion. We define $\boldsymbol{k}^{(\mathrm{int},i)}$ as

where q is the number of unique intermediate events. We recall that $\mathrm{sgn}\left(\delta(\boldsymbol{k}^{(\mathrm{cur})})\right) \neq \phi_{\mathrm{type}}$ corresponds to the case when $\left(\mathcal{P}_{\mathcal{E}} + \epsilon_{\mathcal{P}_{\mathcal{E}}}\right) < \hat{p}_{SParSE}(\mathbf{x}_0, \boldsymbol{k}^{(\mathrm{cur})}, \mathcal{E}; t)$, which requires inverse biasing to reduce over-perturbation.

Starting with $i = 1$, we compute $\hat{p}_{SParSE}(\mathbf{x}_0, \boldsymbol{k}^{(\mathrm{int},i)}, \mathcal{E}; t)$. If $|\hat{p}_{SParSE}\left(\boldsymbol{k}^{(\mathrm{int},i)}\right) - \mathcal{P}_{\mathcal{E}}| \leq \epsilon_{\mathcal{P}_{\mathcal{E}}}$, then the algorithm exits with $\boldsymbol{k}^* = \boldsymbol{k}^{(\mathrm{int},i)}$. Otherwise, we traverse through available sets of biasing parameters to find $\min_r \left(\hat{p}_{SParSE}(\boldsymbol{k}^{(\mathrm{int},r)}) < \left(\mathcal{P}_{\mathcal{E}} - \epsilon_{\mathcal{P}_{\mathcal{E}}}\right)\right)$ for $\mathrm{sgn}\left(\delta(\boldsymbol{k}^{(\mathrm{cur})})\right) == \phi_{\mathrm{type}}$ and $\max_r \left(\left(\mathcal{P}_{\mathcal{E}} + \epsilon_{\mathcal{P}_{\mathcal{E}}}\right) < \hat{p}_{SParSE}(\boldsymbol{k}^{(\mathrm{int},r)})\right)$ for $\mathrm{sgn}\left(\delta(\boldsymbol{k}^{(\mathrm{cur})})\right) \neq \phi_{\mathrm{type}}$. Since $\boldsymbol{\xi}^{(l,\cdot)}$ are sorted in ascending order of its probability, $\boldsymbol{k}^{(\mathrm{int},i)}$ is expected to produce more extreme $f(\mathbf{x}(t))$ values than $\boldsymbol{k}^{(\mathrm{int},i+1)}$. Thus it is not necessary to evaluate all possible $\hat{p}_{SParSE}\left(\boldsymbol{k}^{(\mathrm{int},\cdot)}\right)$. For the case of under-perturbation we can stop the evaluation at the first occurrence of $\boldsymbol{k}^{(\mathrm{int},i)}$ that satisfies the inequality and set $\boldsymbol{k}^{(\mathrm{l}+1)} \leftarrow \boldsymbol{k}^{(\mathrm{int},i)}$. For the case of over-perturbation, however, we stop the simulation at the first occurrence of $\boldsymbol{k}^{(\mathrm{int},i)}$ that *violates* the inequality and set $\boldsymbol{k}^{(\mathrm{l}+1)} \leftarrow \boldsymbol{k}^{(\mathrm{int},i-1)}$.

It is possible that all candidate biasing parameter sets fail to satisfy the update criterion. The failure indicates $\mathcal{P}_{\mathcal{E}}$ lies between $\hat{p}_{SParSE}\left(\boldsymbol{k}^{(l)}\right)$ and $\hat{p}_{SParSE}\left(\boldsymbol{k}^{(\mathrm{int},\cdot)}\right)$. Furthermore, this failure is a direct result of many-to-one relationship between $\boldsymbol{k} \in \mathbb{R}^M_{>0}$ and $\xi \in \mathbb{R}$. Trajectories simulated with two different sets of reaction rates $\boldsymbol{k}$ and $\boldsymbol{k}' = \boldsymbol{k} + \epsilon$ are likely to differ from each other, resulting in $\mathbf{v}_{\boldsymbol{k}} \neq \mathbf{v}_{\boldsymbol{k}'}$. However, both $\mathbf{v}_{\boldsymbol{k}}$ and $\mathbf{v}_{\boldsymbol{k}'}$ may yield the same intermediate events, since they are determined solely by the value of the sorted array at positions $\lceil \boldsymbol{\rho} N \rceil$. Despite the identical $\boldsymbol{\xi}$, SParSE estimates computed with $\boldsymbol{k}$ and $\boldsymbol{k}'$ will differ if the proportion of occurrences of $\mathcal{E}$ is not the same in the two arrays.

In summary, the modified multilevel cross-entropy method for SParSE comprises of 3 steps. First we determine intermediate events for the current reaction rates using the SSA. We then employ dwSSA simulations to compute biasing parameters for each of the intermediate events. Lastly we follow steps described in this section to choose one set of biasing parameters to update reaction rates for the next iteration. This process repeats until either $\boldsymbol{k}^*$ is found or until intermediate reaction rates cannot be updated any more. For computational efficiency, we can combine the first and the last steps by computing

$$k_j^{(\mathrm{int},i)} = \begin{cases} k_j^{(\mathrm{cur})} \cdot \gamma_j^{(l,i)} & \text{if } \mathrm{sgn}\left(\delta\left(\boldsymbol{k}^{(\mathrm{cur})}\right)\right) == \phi_{\mathrm{type}} \\ k_j^{(\mathrm{cur})} \cdot 1/\gamma_j^{(l,i)} & \text{otherwise} \end{cases}, \; j \in \{1, \cdots, M\}, \; i \in \left\{1, \cdots, q\right\}, \tag{12}$$

$\mathbf{v}_{\boldsymbol{k}}^{(\mathrm{int},i)}$ at the same time as computing $\hat{p}_{SParSE}\left(\boldsymbol{k}^{(\mathrm{int},i)}\right)$. We discard $\mathbf{v}_{\boldsymbol{k}}^{(\mathrm{int},i)}$ if the estimate does not satisfy the required inequality or if $\boldsymbol{k}^{(\mathrm{int},i)}$ is not the best candidate for the next intermediate reaction rates.

Lastly we point out that the original dwSSA formula (6) for computing $\boldsymbol{\gamma}^{(l)}$ requires computation of a trajectory weight, which is the product of the likelihood ratio between the propensity function and the predilection function. We recall that the predilection function in the multilevel cross-entropy method used by dwSSA is characterized with the biasing parameters from level $(l-1)$. However, we do not need to compute the trajectory weight in SParSE because its objective is to find a new set of reaction rates that confer the event specifications regardless of $\boldsymbol{k}^{(0)}$, which is used only as a starting position in the path to find $\boldsymbol{k}^*$. The intermediate reaction rates in level l of SParSE reflect cumulative amount of bias applied to the original system up to stage $l-1$ as quantified in (12). Thus the propensity function at level l does not require additional biasing. This leads to using SSA to determine intermediate events and dwSSA to compute $\boldsymbol{\gamma}^{(l)}$ with $\gamma_j^{(l-1)} = 1$, $j \in \{1, \cdots, M\}$. Therefore the formula (6) for SParSE simplifies to

$$\hat{\gamma}_j^{(l)} = \frac{\sum_i' n_{ij}}{\sum_i' \sum_{k=1}^{N_{\mathcal{T}_k}} \left[a_j \left(\mathbf{X}_i^{(l-1)}(t_{ik}) \right) \tau_{ik} \right]}. \tag{13}$$

Exponential interpolation of biasing parameters

Iteratively updating intermediate reaction rates via the modified multilevel cross-entropy method described in the previous section may not find $\boldsymbol{k}$ that satisfies (7). Possible reasons for the failure include poor choice of $\boldsymbol{\rho}$, insufficient N, and nonexistence of candidate intermediate reaction rates that satisfy the update criterion. The first two aligned can lead to slow convergence to $\mathcal{E}$, especially for systems near a deterministic regime or for simulations that demand high accuracy (*i.e.*, small values of $\epsilon_{\mathcal{P}_\mathcal{E}}$). Setting a limit on the maximum number of iterations for the multilevel cross-entropy method can detect slow or non-converging reaction rates, and increasing N and/or modifying $\boldsymbol{\rho}$ will increase the rate of convergence in most aligned. The last phenomenon occurs when no suitable biasing parameters exist to update the reaction rates. Here we have $\left(\hat{p}(\boldsymbol{k}^{(u)}) + \epsilon_{\mathcal{P}_\mathcal{E}}\right) < \mathcal{P}_\mathcal{E} < \left(\hat{p}(\boldsymbol{k}^{(v)}) - \epsilon_{\mathcal{P}_\mathcal{E}}\right)$, where $\hat{p}(\boldsymbol{k}^{(\mathrm{u})}) = \min\left(\hat{p}_{SParSE}(\boldsymbol{k}^{(\mathrm{cur})}), \hat{p}_{SParSE}(\boldsymbol{k}^{(\mathrm{int},i)})\right)$ and $\hat{p}(\boldsymbol{k}^{(\mathrm{v})}) = \max\left(\hat{p}_{SParSE}(\boldsymbol{k}^{(\mathrm{cur})}), \hat{p}_{SParSE}(\boldsymbol{k}^{(\mathrm{int},i)})\right)$, $i = 1, \cdots, q$. The target probability lies between the two estimates $\hat{p}(\boldsymbol{k}^{(u)})$ and $\hat{p}(\boldsymbol{k}^{(v)})$, and the multilevel cross-entropy method is unable to fine-tune intermediate reaction rates to achieve $\mathcal{P}_\mathcal{E}$ within the specified error tolerance $\epsilon_{\mathcal{P}_\mathcal{E}}$.

It is reasonable to assume that a linear combination of $\boldsymbol{k}^{(u)}$ and $\boldsymbol{k}^{(v)}$ may result in $\boldsymbol{k}^*$. A more sophisticated method for approximating $\boldsymbol{k}^*$ would be to fit an interpolant through past intermediate reaction rates. By making the following two assumptions, SParSE computes candidate biasing parameters such that when multiplied to $\boldsymbol{k}^{(0)}$, they may satisfy (7).

Assumption 1. *$\boldsymbol{k}^*$ exists such that $k_j^{(u)} \le k_j^* \le k_j^{(v)}$ for $1 < k_j^{(v)}$ or $k_j^{(v)} \le k_j^* \le k_j^{(u)}$ for $k_j^{(v)} < 1$, $j \in \{1, \cdots, M\}$.*

Assumption 2. *k_j^* can be computed independently from k_h^* for $j \neq h$.*

We note that a single dwSSA trajectory likelihood ratio at level l of the multilevel cross-entropy method is

$$L_{dwSSA}^{(l)}(\mathbf{J}) = \prod_{i=1}^{N_\mathcal{T}} \left[\exp\left\{ \left(a_0^{(l)}(\mathbf{X}(t_i)) - a_0^{(l-1)}(\mathbf{X}(t_i)) \right) \tau_i \right\} \times \left(\gamma_{j_i'}^{(l-1)} \right)^{-1} \right], \tag{14}$$

where $a_0^{(l-1)}(\mathbf{X}(t))$ and $a_0^{(l)}(\mathbf{X}(t))$ are the propensity sum of the trajectory $\mathbf{J}$ at time t generated with $\boldsymbol{k}^{(l-1)}$ and $\boldsymbol{k}^{(l)}$, respectively, and $\boldsymbol{\gamma}^{(l-1)}$ is the biasing parameter used to update the intermediate reaction rates, *i.e.*, $k_j^{(l)} = k_j^{(l-1)} \times \gamma_j^{(l-1)}$, $j \in \{1, \cdots, M\}$. The quantity inside the exponential term is a function of the system state, which is in turn a function of intermediate reaction rates. In order to compare SParSE estimates generated with different intermediate reaction rates, we rewrite them as a function of the initial reaction rates and normalized intermediate biasing parameters, *i.e.*, $k_j^{(\cdot)} = k_j^{(0)} \times \gamma_j^{(0,\cdot)}$, and $\gamma_j^{(0,0)} = 1$. The purpose here is to quantify the relationship between different values of $\boldsymbol{\gamma}^{(0,\cdot)}$ and its corresponding estimates $\hat{p}_{SParSE}(\mathbf{x}_0, \boldsymbol{k}^{(\cdot)}, \mathcal{E}; t)$. Considering the form of the likelihood ratio in (14), a natural form for the interpolant is

$$g\left(\gamma_j^{(0,\cdot)}\right) = q_j \cdot \exp\left\{ p_j \times \gamma_j^{(0,\cdot)} \right\}, \tag{15}$$

where p_j and q_j are constants and $\gamma_j^{(0,\cdot)}$ are normalized version of the intermediate biasing parameters used to compute past SParSE estimates. Output data used in constructing interpolants are the corresponding SParSE estimates $\hat{p}_{SParSE}(\mathbf{x}_0, \boldsymbol{k}^{(\cdot)}, \mathcal{E}; t)$ multiplied by N, the total number of simulations. This particular form allows for fast solving of p_j and q_j with a first order polynomial curve fitting method. We first transform the data to logarithmic scale, compute for two coefficients in the first order polynomial, and then retransform the output with exponentiation. The reason for scaling the output data with N is to preserve as many significant digits as possible,

as logarithmic y-scale is used to compute the polynomial coefficients. While other forms of interpolant may yield more accurate interpolation, (15) allows for fast computation while satisfying Assumption (1), as an exponential function is monotonic.

The number of past intermediate reaction rates available for interpolation varies by factors such as $\boldsymbol{k}^{(0)}$, $\epsilon_{\mathcal{P}_\mathcal{E}}$, and N. Although all past estimates can be used for interpolation, confining the number of data to X closest estimates of $\mathcal{P}_\mathcal{E}$ (*e.g.* $X = 5$) while having at least one estimate on either side of the target probability is recommended, as the accuracy of interpolation may degrade with estimates that are far from the target probability. Due to the construction of the algorithm, there exists at least one estimate on either side of $\mathcal{P}_\mathcal{E}$ when the algorithm enters the exponential interpolation stage. However, we note that the total number of past intermediate reaction rates can be as few as 2. Once the values of p_j and q_j in (15) are determined for all M interpolants, SParSE executes the following steps to further increase the efficiency of simulation.

Step 1: For each $j \in \{1, \cdots, M\}$, project $g(([-2.0\ -1.0\ -0.5\ 0.0\ 0.5\ 1.0\ 2.0]\,\dot{\epsilon}_{\mathcal{P}_\mathcal{E}} + \mathcal{P}_\mathcal{E}) \times N)$ onto the x axis of the interpolant to compute candidate biasing parameters $\bar{\boldsymbol{\gamma}}_j^{(\cdot,s)}$, where the first element $(\cdot)$ in the superscript is the interpolation iteration index and $s \in \{1, \cdots, 7\}$ is the index of the candidates.

Step 2: Compute candidate intermediate reaction rates $\bar{\boldsymbol{k}}^{(\cdot,s)}$, where $\bar{k}_j^{(\cdot,s)} = k_j^{(0)} \times \bar{\boldsymbol{\gamma}}_j^{(\cdot,s)}$

Step 3: Constrain $\bar{k}_j^{(\cdot,s)}$ to satisfy $k_j^{(u)} \leq \bar{k}_j^{(\cdot,s)} \leq k_j^{(v)}$ for $1 < k_j^{(v)}$ or $k_j^{(v)} \leq \bar{k}_j^{(\cdot,s)} \leq k_j^{(u)}$ for $k_j^{(v)} < 1$, $j \in \{1, \cdots, M\}$, if necessary. Reverse the signs in inequalities for $\text{sgn}(\delta\,(\boldsymbol{k})) \neq \phi_{\text{type}}$.

Starting with $s = 4$, we compute $\hat{p}_{SParSE}(\bar{\boldsymbol{k}}^{(\cdot,s)})$. We note that $\bar{\boldsymbol{k}}^{(\cdot,4)}$ corresponds to the reaction rates computed from projecting the exact target probability to the interpolating function. If executing Step 3 results in duplicate candidates, we eliminate the duplicate set(s) and assign the starting index to s such that $q_j \cdot \exp\left\{p_j \times \bar{\gamma}_j^{(\cdot,s)}\right\} = \mathcal{P}_\mathcal{E} N$.

If $\hat{p}_{SParSE}(\bar{\boldsymbol{k}}^{(\cdot,s)})$ confers the target event probability within $\epsilon_{\mathcal{P}_\mathcal{E}}$, SParSE exits with $\boldsymbol{k}^* = \bar{\boldsymbol{k}}^{(\cdot,s)}$. Otherwise, we compute the next estimate with $\bar{\boldsymbol{k}}^{(\cdot,s+1)}$ for $\hat{p}_{SParSE}(\bar{\boldsymbol{k}}^{(\cdot,s)}) < \left(\mathcal{P}_\mathcal{E} - \epsilon_{\mathcal{P}_\mathcal{E}}\right)$, and $\bar{\boldsymbol{k}}^{(\cdot,s-1)}$ for $\left(\mathcal{P}_\mathcal{E} + \epsilon_{\mathcal{P}_\mathcal{E}}\right) < \hat{p}_{SParSE}(\bar{\boldsymbol{k}}^{(\cdot,s)})$. The interpolation stage continues until either $\boldsymbol{k}^*$ is found or the end of candidate reaction rates is reached, at which point an additional interpolation may be executed with updated data. On a rare occasion, $\boldsymbol{k}^*$ lies between two candidate reaction rates without satisfying the error tolerance. This can lead to infinite loop of incrementing and decrementing s by 1 without converging to $\boldsymbol{k}^*$, but the cycle can easily be detected with a mask vector. SParSE implements one by creating a zero vector whose size is equal to the number of candidate biasing parameter sets. Every time a SParSE estimate is computed with candidate reaction rates at index s, we increment s^{th} position of the mask vector. Once the magnitude of any mask vector position becomes greater than 2, we conclude that $\boldsymbol{k}^*$ lies between two candidate biasing reaction rates. At this point, we have refined an upper and lower bound on $\boldsymbol{k}^*$, as all candidate biasing parameters computed satisfy the inequality in Assumption 1. Once the bounds are sufficiently small, an alternative to an additional interpolation is to take a weighted average of the two candidate reaction rates, where the weight is the distance between the SParSE estimate and $\mathcal{P}_\mathcal{E}$. For the examples presented in the following section, we did not encounter any initial reaction rates that required such treatment.

Results and discussion

We illustrate SParSE performance on the following three examples of increasing complexity: a birth-death process, a reversible isomerization process and a Susceptible-Infectious-Recovered-Susceptible (SIRS) disease transmission model. The first two examples were chosen to demonstrate the algorithm's accuracy against the exact solution, which for these examples can be computed using the master equation or the infinitesimal generator matrix [1]. We then progress to a more complex SIRS model, which has no closed-form analytical solution. For each system, we analyze the SParSE performance on all possible combinations of $\mathcal{P}_\mathcal{E} \in \{0.40,\ 0.60,\ 0.80\}$ and $\epsilon_{\mathcal{P}_\mathcal{E}} \in \{0.01,\ 0.05,\ 0.10\}$, where $\mathcal{P}_\mathcal{E}$ and $\epsilon_{\mathcal{P}_\mathcal{E}}$ denote a desired probability for event $\mathcal{E}$ and its error tolerance, respectively. We then compare the result with that from comparable SSA simulations whose reaction rates are selected using uniform random sampling (URS). SParSE also employs URS but only to generate a number of initial reaction rates as a starting point, here set to 30. The number of simulations, N, used to estimate $\mathcal{P}_\mathcal{E}$ per parameter sample is set to 5×10^4 unless mentioned otherwise. We also test the robustness of SParSE by assessing its performance on a low probability event, $\mathcal{P}_\mathcal{E} = 0.010$ and $\epsilon_{\mathcal{P}_\mathcal{E}} = 0.001$ for the birth-death process, and a high probability event, $\mathcal{P}_\mathcal{E} = 0.95$ and $\epsilon_{\mathcal{P}_\mathcal{E}} = 0.005$ for the reversible isomerization process. The number of samples generated for SSA simulations with URS equals the total number of SParSE ensembles computed for a specific simulation scenario, which is the sum of the following quantities: the number of intermediate event computations, the number of estimates computed for each intermediate event, and the number of estimates computed in the exponential interpolation stage. Since the same number of trajectories is

used for computing an intermediate event and a SParSE estimate, it is straightforward to compare the two strategies with computational fairness. For each simulation scenario, we provide four metrics on performance: the total number of SParSE estimates needed for all 30 initial parameter samples, the number of initial parameters that did not reach the solution hyperplane within 10 iterations of multilevel cross-entropy method or 3 iterations of exponential interpolation, the number of parameter sets that required interpolation in addition to the multilevel cross-entropy method, and the number of successful parameter sets generated by SSA simulations using URS for sampling reaction rates. Lastly, we provide movie files of SParSE ensemble simulations for two test scenarios: birth-death with $\mathcal{P}_{\mathcal{E}} = 0.010$ and $\epsilon_{\mathcal{P}_{\mathcal{E}}} = 0.001$ and SIRS with $\mathcal{P}_{\mathcal{E}} = 0.40$ and $\epsilon_{\mathcal{P}_{\mathcal{E}}} = 0.01$.

All computations were run on a desktop with Intel® Xeon® CPU E5-2680, 2.70 GHz processor with 8 cores and 32 GB of RAM. We utilized Matlab's Parallel Computing Toolbox™ (PCT) and the Coder™. The PCT™ was used to simulate 8 SParSE ensembles in parallel while the Coder™ was used to convert frequently-used custom Matlab functions into low-level C functions for faster computation.

Birth-death process

Our first example is a birth-death process.

$$\emptyset \xrightarrow{k_1} Y, \qquad 1.0 \leq k_1 \leq 1.7$$

$$Y \xrightarrow{k_2} \emptyset, \quad 0.0125 \leq k_2 \leq 0.025$$

with $\mathbf{x}_0 = [40]$ and the target event $\mathcal{E}$ being molecular count of Y reaching 80 before $t = 100$. Table 1 summarizes the results for the 9 standard test aligned, where SParSE achieved 100% success rate in finding $\boldsymbol{k}^*$, a vector of reaction rates $\left[k_1^* \, k_2^*\right]$ that confers desired $\mathcal{P}_{\mathcal{E}}$ and $\epsilon_{\mathcal{P}_{\mathcal{E}}}$, for 8 test aligned. For $\mathcal{P}_{\mathcal{E}} = 0.60$ and $\epsilon_{\mathcal{P}_{\mathcal{E}}} = 0.01$, two of thirty samples, $\boldsymbol{k}_3^{(0)} = [1.606\ 0.0140]$ and $\boldsymbol{k}_{27}^{(0)} = [1.684\ 0.0148]$ (subscript representing the index of initial reaction rates), failed to converge after three rounds of exponential interpolations. We discuss the details of the failure in Appendix C of Additional file 1.

We picked one of 30 initial reaction rates for $\mathcal{P}_{\mathcal{E}} = 0.60$ and $\epsilon_{\mathcal{P}_{\mathcal{E}}} = 0.05$ to illustrate a complete progression of the algorithm. Figure 1 contains a flow chart of a SParSE run with $\boldsymbol{k}_{14}^{(0)} = [1.2414\ 0.02445]$. This particular set required two rounds of multilevel cross-entropy method and one exponential interpolation, which required computing two SParSE estimates (out of seven candidates) to reach the solution hyperplane $\left(\boldsymbol{k}_{(14)}^* = [1.586\ 0.0190]\right)$. Figure 2 shows an illustration of the interpolation results. We see from Table 1 that this particular scenario required 164 SParSE estimates (30 initial, 71 intermediate, and 30 final) in addition to 33 intermediate event computations in order to reach the solution hyperplane. An ensemble result is displayed in Figure 3, where the values of z axis are set to the probability of the target event, which is computed using k_1 and k_2 values defined by the data's x and y coordinates, respectively. Together the figure shows the contour of the event probability surface for different values of k_1 and k_2. Despite a rapid change in the event probability around $\mathcal{P}_{\mathcal{E}} = 0.60$, SParSE was able to find a point in the solution hyperplane for all 30 sets of initial reaction rates.

Table 1 Results of SParSE applied to the birth-death process

$\mathcal{P}_{\mathcal{E}}$	$\epsilon_{\mathcal{P}_{\mathcal{E}}}$	SParSE samples	Interpolations	Failures	Successful URS
0.40	0.01	286 (35)	29 [1, 7, 19, 3]	0	3
	0.05	182 (33)	23 [7, 22, 1, 0]	0	12
	0.10	133 (29)	19 [11, 19, 0, 0]	0	11
0.60	0.01	319 (36)	30 [0, 3, 18, 9]	2	2
	0.05	164 (33)	26 [4, 25, 0, 1]	0	8
	0.10	120 (30)	18 [12, 18, 0, 0]	0	12
0.80	0.01	240 (51)	28 [2, 17, 10, 1]	0	2
	0.05	137 (43)	15 [15, 15, 0, 0]	0	6
	0.10	108 (37)	1 [29, 1, 0, 0]	0	12

The first column denotes the target probability, the second column absolute error tolerance, the third column the total number of SParSE samples computed for the 30 initial parameter sets with the number inside the parenthesis indicating the total number of intermediate event computations, the fourth the number of initial reaction rate parameter sets that required exponential interpolation, the fifth the number of initial sets that did not converge to the solution hyperplane, and the sixth the number of successful parameter sets generated with URS. For the fourth column, four numbers inside the bracket indicate the number of parameter sets that required 0, 1, 2, and 3 interpolations, respectively. $N = 5 \times 10^4$.

Next we illustrate the robustness of SParSE by choosing a very small target probability $\mathcal{P}_{\mathcal{E}} = 0.010$ and $\epsilon_{\mathcal{P}_{\mathcal{E}}} = 0.001$ (animated illustration of SParSE simulations for this scenario is provided as Additional file 2). For this problem, we increased N to 2×10^5 to reduce the relative uncertainty in the estimate [20]. Table 2 summarizes the results. We see that all 30 initial sets of reaction rates successfully converged to the solution hyperplane while SSA-URS yielded only 3 successful samples. Figure 4 displays all 215 SParSE samples (30 initial, 155 intermediate, and 30 final) for this scenario. Figure 5 displays result of the same simulation scenario using SSA-URS, except that it contains 36 additional data to accommodate the total number of intermediate event computations SParSE required. We note that the parameter ranges shown in Figure 4 differ from that in Figure 5, whose data obey parametric constraints specified in the model description (*i.e.*, $1.0 \leq k_1 \leq 1.7$ and $0.0125 \leq k_2 \leq 0.025$). These constraints are shown as white dashed lines in Figure 4. The reason Figure 4

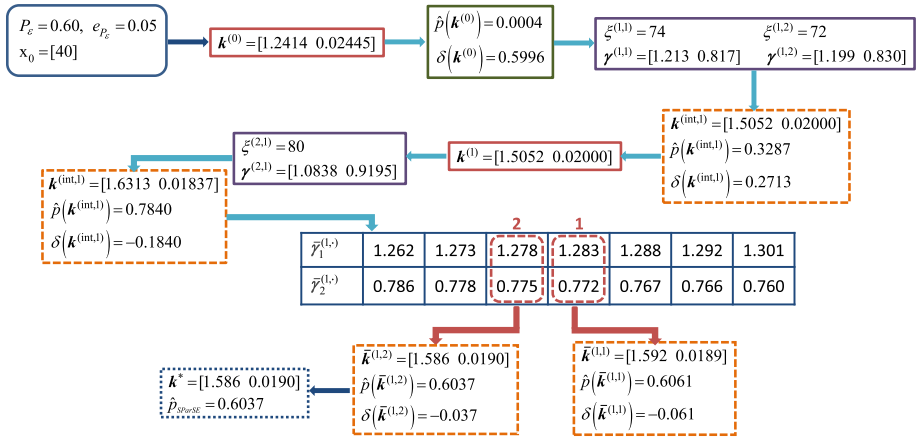

Figure 1 Flow chart of SParSE simulation on the birth-death process with $\mathcal{P}_{\mathcal{E}} = 0.60, \epsilon_{\mathcal{P}_{\mathcal{E}}} = 0.05, k^{(0)} = [1.2414\ 0.02445]$, and $N = 5 \times 10^4$. Arrows represent sequential steps taken by the algorithm.

contains data outside the perimeter of white dashed lines is that our implementation of SParSE does not utilize the parametric constraints other than to generate initial sets of reaction rates. Changing the implementation of SParSE to enforce the parametric constraints throughout the simulation requires the user to provide a parametric region that contains the solution hyperplane. In this alternate implementation, if the solution hyperplane does not exist within the user-specified region, all computations are wasted. The current implementation allows for computation of the solution hyperplane regardless of its location while exploiting the user's knowledge in generating initial reaction rates.

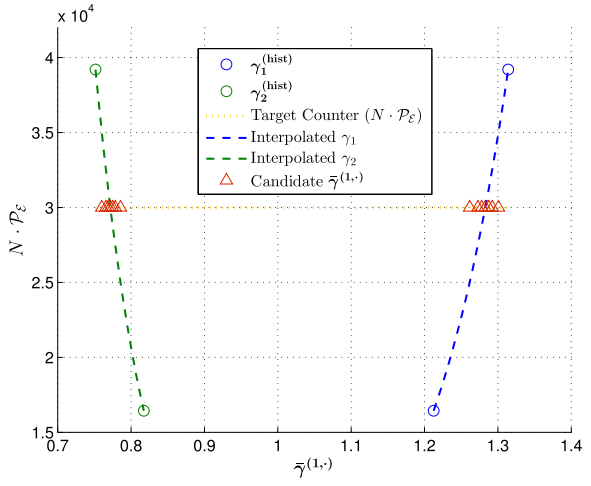

Figure 2 Illustration of exponential interpolation for the birth-death process with $\mathcal{P}_{\mathcal{E}} = 0.60, \epsilon_{\mathcal{P}_{\mathcal{E}}} = 0.05, k^{(0)} = [1.2414\ 0.02445]$, and $N = 5 \times 10^4$. Yellow horizontal dotted line is the desired number of successful trajectories, *i.e.*, $N \times \mathcal{P}_{\mathcal{E}} = 3 \times 10^4$. Blue and green circles denote the past intermediate biasing parameters for R_1 and R_2, respectively, normalized with respect to $k^{(0)}$. Blue and green dashed lines are the interpolants constructed from the past intermediate basing parameters, and the red triangles are candidate biasing parameters computed according to Step 3 in Subsection *Updating intermediate reaction rates*.

Reversible isomerization process

Our next example concerns a reversible isomerization process, where two conformational isomers A and B are interconverted by rotation about single bonds:

$$A \xrightarrow{k_1} B, \quad 0.1 \leq k_1 \leq 0.3$$
$$B \xrightarrow{k_2} A, \quad 0.3 \leq k_2 \leq 1.0,$$

with $\mathbf{x}_0 = [100\,0]$, *i.e.*, all molecules are initially in A form. The target event is set to population of B reaching 30 before $t = 10$. Table 3 summarizes the results from 9 standard test scenarios, all of which attained 100% convergence to the solution hyperplane. We see that the total number of SParSE samples required for $\mathcal{P}_{\mathcal{E}} \in \{0.40, 0.60\}$ is comparable between the birth-death and the reversible isomerization processes. However, the latter required considerably more number of samples for $\mathcal{P}_{\mathcal{E}} = 0.80$. This is due to the difference in the contour of target event probability surface between the two processes. Figure 6 compares ensemble results between the two processes for $\mathcal{P}_{\mathcal{E}} = 0.80$ and $\epsilon_{\mathcal{P}_{\mathcal{E}}} = 0.05$. Figure 6A represents the birth-death process and Figure 6B the reversible isomerization process. We see that the reversible isomerization process contains a significantly larger parametric region that corresponds to $\mathcal{P}_{\mathcal{E}} > 0.80$, and that the probability in this region changes slowly (*i.e.*, plateau-like contour). Only two over-perturbing initial reaction rates (*i.e.*, orange squares above the green dotted line corresponding to $\mathcal{P}_{\mathcal{E}} + \epsilon_{\mathcal{P}_{\mathcal{E}}} = 0.85$) were generated for the birth-death

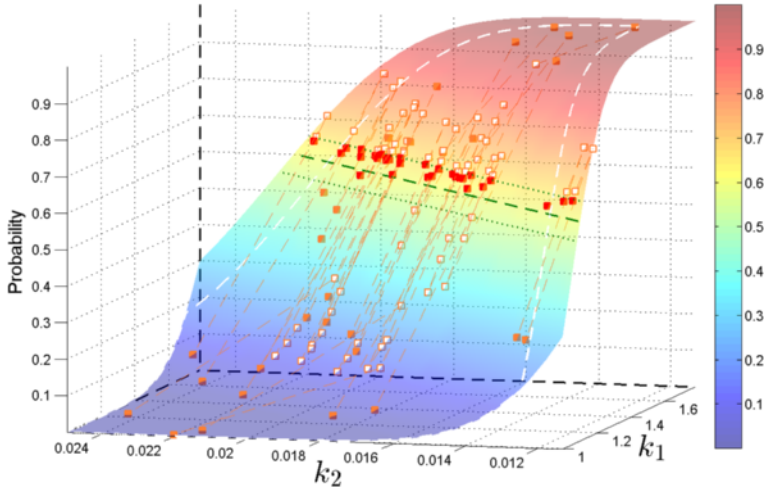

Figure 3 Ensemble result for the birth-death process with $\mathcal{P}_\mathcal{E} = 0.60$, $\epsilon_{\mathcal{P}_\mathcal{E}} = 0.05$, and $N = 5 \times 10^4$. SParSE required a total of 131 samples (30 initial, 71 intermediate, and 30 final). The green dashed line denotes the exact solution for $\mathcal{P}_\mathcal{E} = 0.60$ and the green dotted lines represent ± 0.05 absolute error tolerance band. Initial reaction rates are represented by orange squares, intermediate reaction rates by white squares, and $\boldsymbol{k}^*$s by red squares. Orange dashed lines connect any two subsequent reaction rates originated from the same $\boldsymbol{k}^{(0)}$. White dashed lines represent the parameter ranges specified prior to simulation.

process while eleven such rates were generated for the reversible isomerization process. Intermediate reaction rates (white squares) of these eleven samples are close together due to the slowly changing probability in their vicinity. Lastly we note that none of the data in Figure 6B left the original parameter ranges stated in the model description. This confirms that even for simple systems such as birth-death process and reversible isomerization process, it is nontrivial to predict parameter ranges that form a convex bound.

Next we choose a high probability target of $\mathcal{P}_\mathcal{E} = 0.95$ and $\epsilon_{\mathcal{P}_\mathcal{E}} = 0.005$ with $N = 10^5$. Table 4 summarizes the result. We see that one of 30 samples failed to converge. SParSE was not able to find $\boldsymbol{k}^*$ for $\boldsymbol{k}_{27}^{(0)} = [0.205\ 0.414]$ after 3 rounds of exponential interpolations. The qualitative explanation for the failure is the same as with the birth-death process for $\mathcal{P}_\mathcal{E} = 0.60$ and $\epsilon_{\mathcal{P}_\mathcal{E}} = 0.01$, which is discussed in Appendix C of Additional file 1.

It is worth pointing out that the number of successful parameter sets generated with SSA-URS varies widely from simulation to simulation. The expected number, however, is the volume of the solution hyperplane (which changes with different values of $\epsilon_{\mathcal{P}_\mathcal{E}}$) divided by the total volume, multiplied by the total number of reaction rate samples generated with URS. Here the prescribed parameter ranges are used to compute both volumes (*e.g.*, $0.1 \leq k_1 \leq 0.3$ and $0.3 \leq k_2 \leq 1.0$ for the reversible isomerization process). Since the acceptable solution volume increases with larger $\epsilon_{\mathcal{P}_\mathcal{E}}$, the number of uniform random samples that reside in the solution hyperplane should increase as well. Similarly the expected number of intermediate reaction rates used by SParSE to reach the solution hyperplane decreases because the need for fine-tuning, *i.e.*, exponential interpolation, declines with larger $\epsilon_{\mathcal{P}_\mathcal{E}}$. This trend is confirmed by the simulation results for all three examples presented in this paper (columns 4 and 6 of Tables 1, 3, and 5).

Table 2 Results of SParSE applied to the birth-death process

$\mathcal{P}_\mathcal{E}$	$\epsilon_{\mathcal{P}_\mathcal{E}}$	SParSE samples	Interpolations	Failures	Successful URS
0.010	0.001	251 (36)	27 [3, 17, 9, 1]	0	3

The column identities match those of Table 1. $N = 2 \times 10^5$.

Simple SIRS disease dynamics

The final example is Susceptible-Infectious-Recovered-Susceptible (SIRS) disease transmission model, which consists of the following three reactions:

$$\begin{aligned} S + I &\xrightarrow{\beta} 2I, \quad 0.005 \leq \beta \leq 0.150 \\ I &\xrightarrow{\gamma} R, \quad 0.50 \leq \gamma \leq 4.0 \\ R &\xrightarrow{\omega} S, \quad 0.10 \leq \omega \leq 3.0 \end{aligned}$$

with $\mathbf{x}_0 = [100\ 1\ 0]$, where $\mathbf{x} = [S\ I\ R]$. This model describes a homogenous, fixed population setting where members of S become infected by members of I, who recover from the infection at rate γ. Once recovered, members of R have immunity against the infection. However, the immunity wanes at rate ω, and this transition from recovered to susceptible compartment replenishes

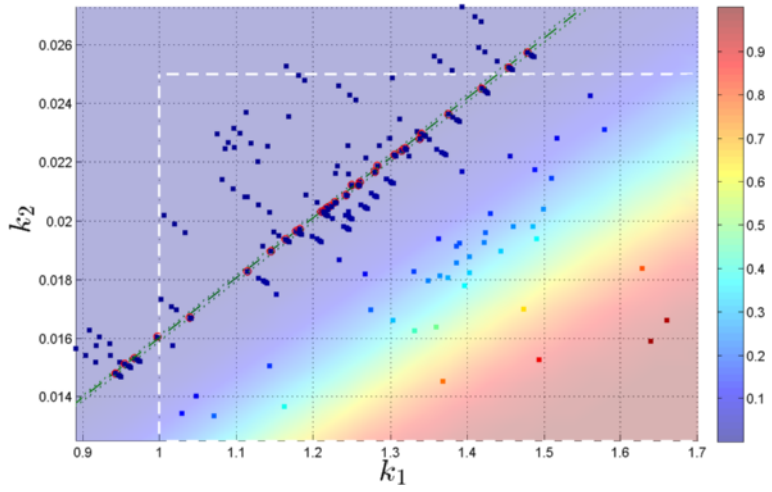

Figure 4 SParSE ensemble result for the birth-death process with $\mathcal{P}_{\mathcal{E}} = 0.010$, $\epsilon_{\mathcal{P}_{\mathcal{E}}} = 0.001$, and $N = 2 \times 10^5$. SParSE required a total of 215 samples (30 initial, 155 intermediate, and 30 final). The green dashed line denotes the exact solution for $\mathcal{P}_{\mathcal{E}} = 0.010$ and the green dotted lines ± 0.001 absolute error tolerance band. Final reaction rates $\boldsymbol{k}^*$ are encircled in red. White dashed lines represent the parameter ranges specified prior to simulation.

the population of S. The target event for this system is set to the population of I reaching 50 before $t = 30$. Unlike the first two examples, there is no closed-form analytical solution for this model. In order to construct the probability voxel for the specified parameter ranges, we divided each parameter region into 30 uniformly-spaced grids and computed each combination with the SSA, where each of $27,000 (30^3)$ ensembles was simulated with $N = 10^5$. We then further refined the resolution of the probability volume to a $70 \times 70 \times 70$ grid using **interp3** function in Matlab, which applied linear interpolation to the 3-dimensional mesh data from SSA simulations. Figure 7 displays the final solution volume, where the color of each point represents the target event probability according to the color bar on the right of the figure.

As with the previous examples, we tested SParSE on all possible combinations of $\mathcal{P}_{\mathcal{E}} \in \{0.40,\ 0.60,\ 0.80\}$ and $\epsilon_{\mathcal{P}_{\mathcal{E}}} \in \{0.01,\ 0.05,\ 0.10\}$ and measured the same quantities as in Table 1. Table 5 summarizes the results. We see that SParSE achieved 100% success rate for all scenarios tested. However, statistics on column 1 demonstrates that the total number of estimates computed for any SIRS scenario is greater than the one for the first two examples with the same target probability and error tolerance.

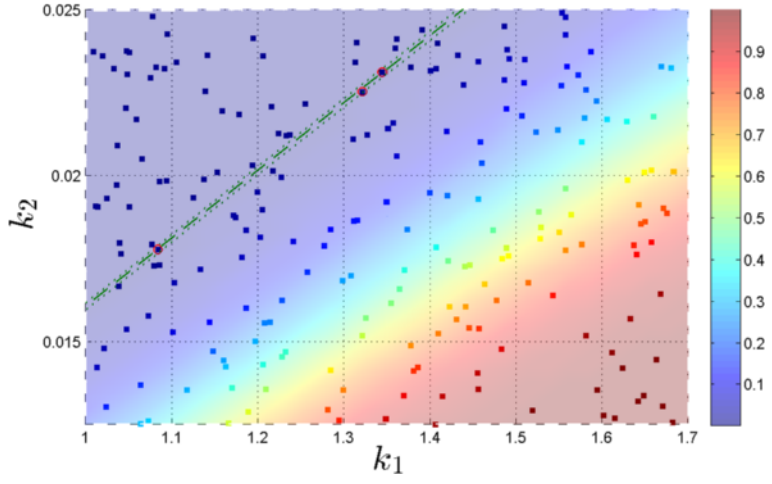

Figure 5 SSA-URS ensemble result for the birth-death process with $\mathcal{P}_{\mathcal{E}} = 0.010$, $\epsilon_{\mathcal{P}_{\mathcal{E}}} = 0.001$, and $N = 2 \times 10^5$. Color of each square represents $\mathcal{P}_{\mathcal{E}}$ given its k_1 and k_2 values according to the color bar given on the right. Legend identities match those of Figure 4.

Table 3 Results of SParSE applied to the reversible isomerization process

$\mathcal{P}_\mathcal{E}$	$\epsilon_{\mathcal{P}_\mathcal{E}}$	SParSE samples	Interpolations	Failures	Successful URS
0.40	0.01	269 (50)	26 [4 9 14 3]	0	2
	0.05	181 (48)	18 [12 17 1 0]	0	6
	0.10	148 (44)	13 [17 13 0 0]	0	8
0.60	0.01	313 (58)	28 [2 6 16 6]	0	4
	0.05	198 (51)	18 [12 14 4 0]	0	8
	0.10	156 (47)	13 [17 13 0 0]	0	11
0.80	0.01	290 (75)	27 [3 17 5 5]	0	1
	0.05	201 (67)	12 [18 11 1 0]	0	6
	0.10	165 (61)	4 [26 4 0 0]	0	18

The column identities match those of Table 1. $N = 5 \times 10^4$.

SIRS ensembles required up to 198 more samples, except for $\mathcal{P}_\mathcal{E} = 60$ and $\epsilon_{\mathcal{P}_\mathcal{E}} = 0.01$, which required one fewer sample than the birth-death process. If we ignore the intermediate event computations, the number of samples required by all three examples are comparable to each other (mean difference of 17.7 samples). In addition, quantities in column 4 of Tables 1, 3 and 5 indicate that SParSE required fewer interpolations on the SIRS model than it did on the other two examples. These results imply that the multilevel cross-entropy method applied to the SIRS model made conservative moves to reach the solution hyperplane; the algorithm required many intermediate event computations to approach the vicinity of $\mathcal{P}_\mathcal{E}$ but fewer fine-tuning steps (*i.e.*, exponential interpolations). Although the same $\boldsymbol{\rho}(\delta)$ values were used for all three examples, we see that its effect differs depending on the underlying system.

Two expected trends emerge from Table 5; the total number of SParSE samples and the total number of exponential interpolations required to reach the solution hyperplane decrease with increasing $\epsilon_{\mathcal{P}_\mathcal{E}}$. Although numbers in columns 3 and 4 differ among Tables 1, 3, and 5, qualitative algorithmic behavior as a function of $\epsilon_{\mathcal{P}_\mathcal{E}}$ remain the same for all three examples. As for its performance, SParSE outperformed SSA-URS (by a factor of 1.15 to 10) on all scenarios except one. For $\mathcal{P}_\mathcal{E} = .80$ and $\epsilon_{\mathcal{P}_\mathcal{E}} = 0.10$, SSA with URS yielded 41 successful sets, while SParSE yielded 30. We note that the maximum number of successful sets for SParSE cannot exceed the number of initial parameter sets, which is 30 for all examples presented in this paper. Also, the parameter ranges we chose for the SIRS model result in an uneven distribution of the target probability. From Figure 7, we see that a significant portion of the probability volume belongs to high (> 0.8) or low (< 0.2) probability region. Since the SSA-URS success probability is determined solely by the ratio between the volume of the solution hyperplane and the total volume defined by the specified parameter ranges, this particular scenario is biased to be more favorable toward SSA-URS. For general applications involving a target event, however, we cannot expect the solution hyperplane to lie within the user-specified parameter ranges, to which SSA-URS samples are confined. If this region does not contain the solution hyperplane, SSA-URS is unable to produce $\boldsymbol{k}^*$ regardless of the number of samples generated. The current implementation of SParSE, on the other hand, is highly likely to find the closest point (cross-entropy metric) in the solution hyperplane through multilevel cross-entropy method and exponential interpolation stages, both of which are not limited by the user-specified parameter ranges. In practical situations, it is likely that the user does not have enough systematic

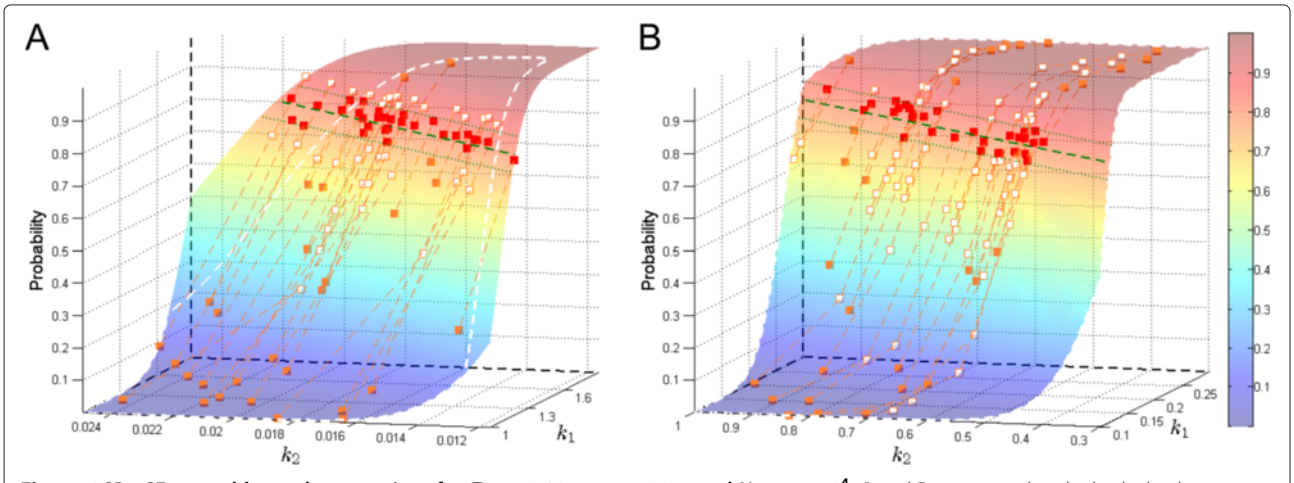

Figure 6 SParSE ensemble result comparison for $\mathcal{P}_\mathcal{E} = 0.80$, $\epsilon_{\mathcal{P}_\mathcal{E}} = 0.05$, and $N = 5 \times 10^4$. A and **B** correspond to the birth-death process and the reversible isomerization process, respectively. Legend identities match those of Figure 4.

Table 4 Results of SParSE applied to the reversible isomerization process

$\mathcal{P}_\mathcal{E}$	$\epsilon_{\mathcal{P}_\mathcal{E}}$	SParSE samples	Interpolations	Failures	Successful URS
0.95	0.005	302 (109)	10 [20, 3, 4, 3]	1	2

The column identities match those of Table 1. $N = 1 \times 10^5$.

insight to identify a region that contains the solution hyperplane for a particular target event. We expect SParSE to be more efficient than SSA-URS by orders of magnitude in such aligned, as the performance of SParSE is much less sensitive to the dimensionality of the search space and the volume within $\epsilon_{\mathcal{P}_\mathcal{E}}$ of the solution hyperplane than the performance of SSA-URS is.

We picked one scenario, $\mathcal{P}_\mathcal{E} = 0.40$ and $\epsilon_{\mathcal{P}_\mathcal{E}} = 0.05$, for visual comparison between SParSE and SSA-URS outcomes outcomes (animated illustration of SParSE simulations for this scenario is provided in Additional file 3). Figure 8 display the ensemble result for each method. The solution hyperplane for $\mathcal{P}_\mathcal{E} = 0.40$ is represented by a cyan-colored surface, which was obtained by applying **isosurface** function in Matlab to the probability volume. We have omitted displaying the region corresponding to $\mathcal{P}_\mathcal{E} \pm \epsilon_{\mathcal{P}_\mathcal{E}}$ for clear visualization of data. We see that the volume of the solution hyperplane for this particular scenario is small relative to the volume of the entire voxel. Thus we expect poor performance from SSA-URS, which is confirmed by statistics in Table 5. SSA-URS generated only 5 successful parameter combinations out of 467 samples, while SParSE generated 30. Since one point in the solution hyperplane corresponds to one set of initial reaction rates in SParSE, this indicates 100% convergence. The number of data in Figure 8A and 8B are 305 and 467, respectively. Figure B contains 162 more data to compensate for the total number of intermediate event computations required by SParSE. Despite having fewer data, SParSE

Table 5 Results of SParSE applied to the SIRS model

$\mathcal{P}_\mathcal{E}$	$\epsilon_{\mathcal{P}_\mathcal{E}}$	SParSE samples	Interpolations	Failures	Successful URS
0.4	0.01	467 (162)	26 [4, 15, 8, 3]	0	5
	0.05	282 (149)	8 [22, 6, 1, 1]	0	6
	0.10	246 (142)	4 [26, 4, 0, 0]	0	14
0.6	0.01	318 (63)	28 [2, 8, 18, 2]	0	3
	0.05	206 (59)	20 [10, 19, 0, 1]	0	8
	0.10	166 (57)	10 [20, 9, 0, 1]	0	8
0.8	0.01	328 (113)	8 [22, 4, 3, 1]	0	4
	0.05	224 (90)	1 [29, 0, 0, 1]	0	26
	0.10	177 (73)	0 [30, 0, 0, 0]	0	41

The column identities match those of Table 1. $N = 5 \times 10^4$.

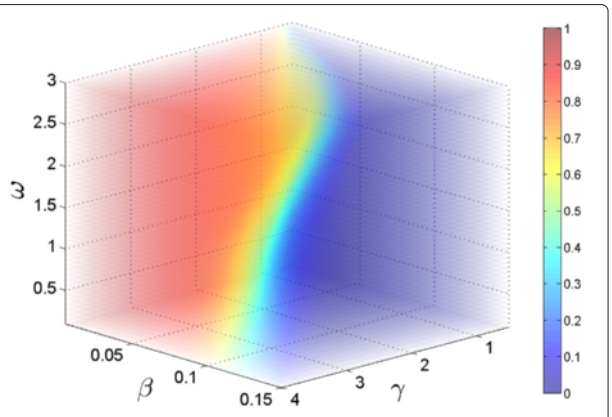

Figure 7 Probability volume (70 × 70 × 70 grid) for the SIRS model after applying Matlab's interp3 function to the 30 × 30 × 30 data simulated with SSA. Color of each voxel represents $\mathcal{P}_\mathcal{E}$ given its β, γ, and ω values according to the color bar given on the right.

produced not only 6 times the number of $\boldsymbol{k}^*$ but also data that are closely scattered around the solution hyperplane. The latter fact offers couple advantages. First, having good resolution near $\mathcal{P}_\mathcal{E} \pm \epsilon_{\mathcal{P}_\mathcal{E}}$ enables more accurate mapping of the solution hyperplane, as it is unknown or computationally infeasible to be computed even for moderately sized systems (*e.g.*, 5–20 parameters). In addition if we were to run another set of simulations on the identical scenario, we can generate initial reaction rates that are near the solution hyperplane using the past simulation results.

Lastly, we chose one of 30 initial reaction rates to illustrate a complete progression of SParSE on the SIRS model. Unlike the set of initial reaction rates chosen for the birth-death process ($\boldsymbol{k}_{14}^{(0)} = [1.2414\ 0.02445]$ with $\mathcal{P}_\mathcal{E} = 0.60$ and $\epsilon_{\mathcal{P}_\mathcal{E}} = 0.01$) in Figure 1, which under-perturbs the system, the set of initial reaction rates chosen here, $\boldsymbol{k}_{26}^{(0)} = [0.0942\ 1.7150\ 0.6196]$, over-perturbs the system. Figure 9 displays the flow chart of SParSE simulations for this scenario, and Figure 10 illustrates the results from the first and second exponential interpolations, respectively. The interpolants for all three reactions exhibit a good fit with respect to the past biasing parameters, and the quality of fit improves in the second iteration with updated past estimates closer to the target probability. According to the flow chart and Figure 10A, SParSE entered the first iteration of exponential interpolation with four past estimates, three of which under-perturbed the system (from multilevel cross-entropy method). After exhausting the candidate biasing parameters from the first iteration, all of which produced estimates greater than 0.61, SParSE entered a second iteration of exponential interpolation. At this point, the top five closest estimates to $\mathcal{P}_\mathcal{E} = 0.60$ all came from over-perturbing biasing parameter sets. SParSE then removed the most over-perturbing set and inserted the least under-perturbing set in attempt to

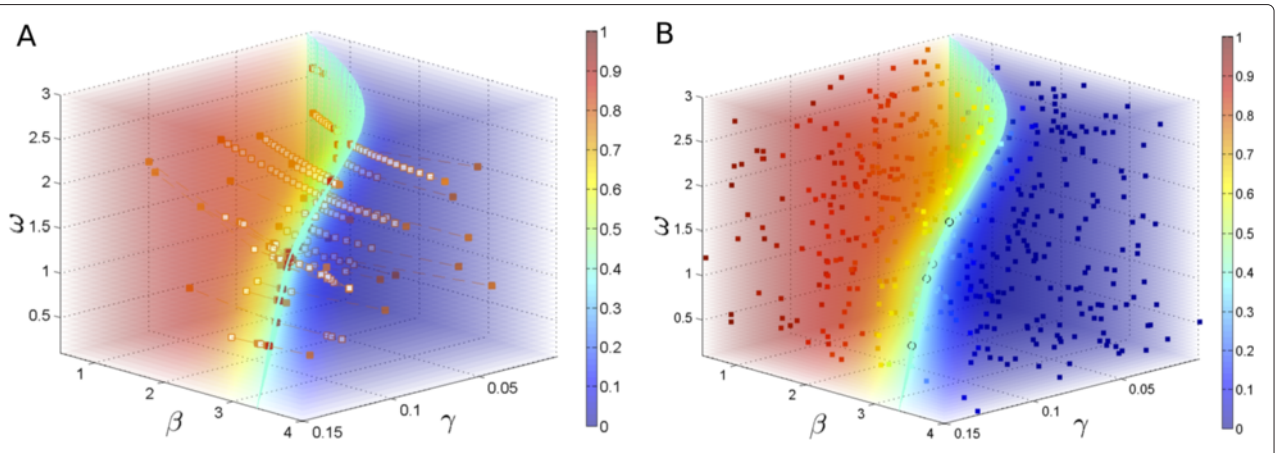

Figure 8 SIRS ensemble result comparison between SParSE and SSA-URS for $\mathcal{P}_{\mathcal{E}} = 0.40$, $\epsilon_{\mathcal{P}_{\mathcal{E}}} = 0.01$, and $N = 5 \times 10^4$. Cyan-colored surface represents the solution hyperplane for the target event. **A** represents the result of SParSE applied to the SIRS model. Orange squares represent the initial reaction rates, white squares intermediate reaction rates, and red squares the final reaction rates on the solution hyperplane. Orange dashed lines connect any two subsequent reaction rates originated from the same initial reaction rates. **B** represents the result of SSA-URS applied to the SIRS model. Color of each rectangle represents the target event probability from the represented parameter combination according to the color bar given on the right. Out of 467 samples, only 5 lie on the solution hyperplane. These 5 successful parameter sets are encircled in black. Due to the 3-dimensional nature of this figure, there is no single angle where both the solution hyperplane and the 5 sets are easily visible.

improve the quality of the interpolant. Figure 10B reflects these modifications. The last candidate from the second interpolation produced an estimate within $\epsilon_{\mathcal{P}_{\mathcal{E}}} = 0.01$, at which point the algorithm exited with the final reaction rates ($\boldsymbol{k}^*_{(26)} = [0.0917\ 1.899\ 0.587]$). We note that the slope of the interpolants in Figure 10 are opposite from the ones in Figure 2. This is because inverse biasing technique is used for over-perturbing reaction rates, as described by Equation 12.

Conclusions

In this paper, we presented SParSE–a novel stochastic parameter estimation algorithm for events. SParSE contains two main research contributions. First, it presents a novel modification of the multilevel cross-entropy method that (1) concurrently computes multiple intermediate events as well as their corresponding biasing parameters, and (2) handles over-perturbing initial reaction rates as well as under-perturbing ones. Second, it uses information from past simulations to automatically find a path to the parametric hyperplane corresponding to the target event with user-specified probability and absolute error tolerance.

By introducing a novel heuristic for handling reaction rates that over-perturb the system, SParSE can handle target events whose probability does not need to be rare with respect to the initial reaction rates $\boldsymbol{k}^{(0)}$. If the user wishes to compute the probability of observing $\mathcal{P}_{\mathcal{E}}$ with respect to $\boldsymbol{k}^{(0)}$, however, it can be done by simply running the dwSSA with biasing parameters that are the ratio between the final reaction rates $\boldsymbol{k}^*$ from SParSE and $\boldsymbol{k}^{(0)}$. No additional multilevel cross-entropy simulations are required by the dwSSA to determine biasing parameters since the final set of reaction rates computed by SParSE contains this information. For this reason, SParSE improves upon the dwSSA in that it can handle an additional type of rare event. The only class of rare events whose probability dwSSA can estimate is the one that is seldom reached by the system using the original reaction rates. SParSE, on the other hand, can also compute the probability of events that are reached too often with respect to the target probability using the original reaction rates. Average frequency of observing such target event with $\boldsymbol{k}^{(0)}$ would be much higher than the desired frequency (*i.e.*, $(\mathcal{P}_{\mathcal{E}} \pm \epsilon_{\mathcal{P}_{\mathcal{E}}}) \times N$), and therefore the probability of observing $\mathcal{E}$ with success rate $\mathcal{P}_{\mathcal{E}} \pm \epsilon_{\mathcal{P}_{\mathcal{E}}}$ and reaction rates $\boldsymbol{k}^{(0)}$ would be very small, yet its biasing parameters are uncomputable with the dwSSA, but are computable with SParSE.

It is important to note that the computational complexity of SParSE is *independent* of the number of parameters to be estimated. Like the dwSSA [17], SParSE utilizes information-theoretic concept of cross-entropy to concurrently compute biasing parameters for all reactions in the system. Moreover, SParSE avoids serial computation of biasing parameters for multiple intermediate events at any given stage of multilevel cross-entropy method by introducing a clever ordering of intermediate events and data management. Figures 4, 5 and 8 illustrate that SParSE not only is more efficient than SSA-URS in finding $\boldsymbol{k}^*$ but also gives a better resolution of the area near the solution hyperplane. This is because intermediate reaction rates computed by SParSE are guaranteed to be closer

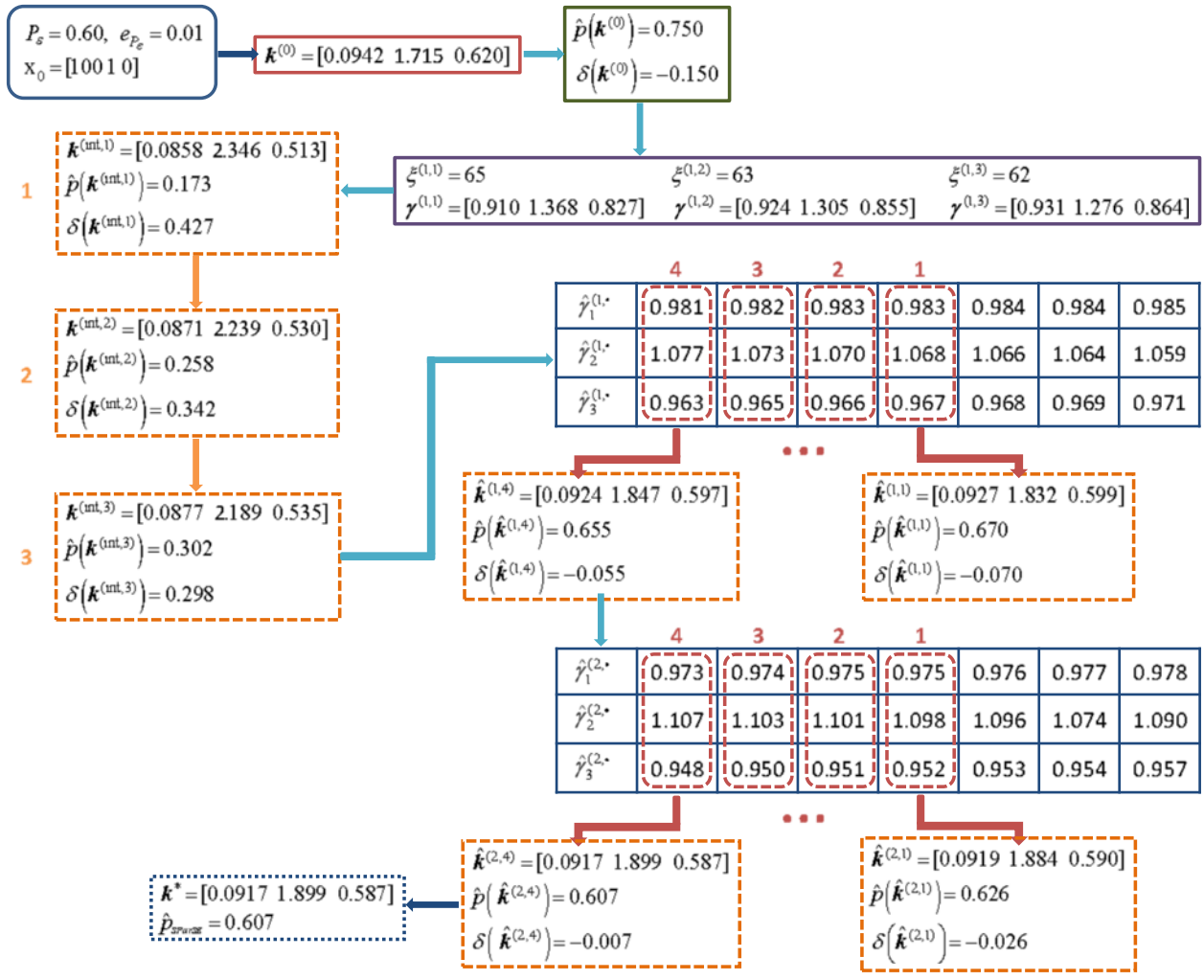

Figure 9 Flow chart of SParSE simulation on the SIRS model with $\mathcal{P}_\mathcal{E} = 0.60$, $\epsilon_{\mathcal{P}_\mathcal{E}} = 0.01$, $k^{(0)} = [0.0942\ 1.7150\ 0.6196]$, and $N = 5 \times 10^4$. This particular simulation required three stages of multilevel cross-entropy method and two rounds of exponential interpolation. Arrows represent sequential steps taken by the algorithm.

to k^* than $k^{(0)}$ is. Thus intermediate reaction rates near k^* can be used to improve the quality of interpolation in constructing the solution hyperplane. Another computational asset of SParSE is that it is highly parallelizable. In large scale application, multiple sets of initial reaction rates can be dispatched separately since each set finds its way to the solution hyperplane independently from each other. In smaller scale, SParSE estimate computation or an ensemble of multilevel cross-entropy method simulations also can be parallelized. In simulating examples presented in this paper, we have chosen the latter method; each set of N simulations was distributed among 8 cores using the Parallel Computing Toolbox™ in Matlab. Lastly, a single SParSE trajectory from the multilevel cross-entropy method without any biasing (*i.e.*,$\boldsymbol{\gamma} = \vec{\mathbf{1}}$) generates the same number of uniform random numbers as the SSA does. The only difference is that SParSE requires additional data management for recording biasing parameter information (two floating point numbers for each reaction [17]), which is used in the next round of multilevel cross-entropy method. It is difficult to compare the exact computational cost between the two methods when SParSE utilizes $\boldsymbol{\gamma} \neq \vec{\mathbf{1}}$; depending on the amount of bias applied per reaction, the number of random numbers generated per trajectory will differ between the two methods even if the same reaction rates were used. For the exponential interpolation stage in SParSE, SSA is used to compute $\hat{p}_{SParSE}$, thus the computational cost of SParSE and SSA trajectory are identical for a given set of reaction rates.

One of the inputs required by SParSE is a range of values each parameter can take. There is no theoretical limit on the parameter range SParSE can manage; however, it is required for the following practical reasons. First, the

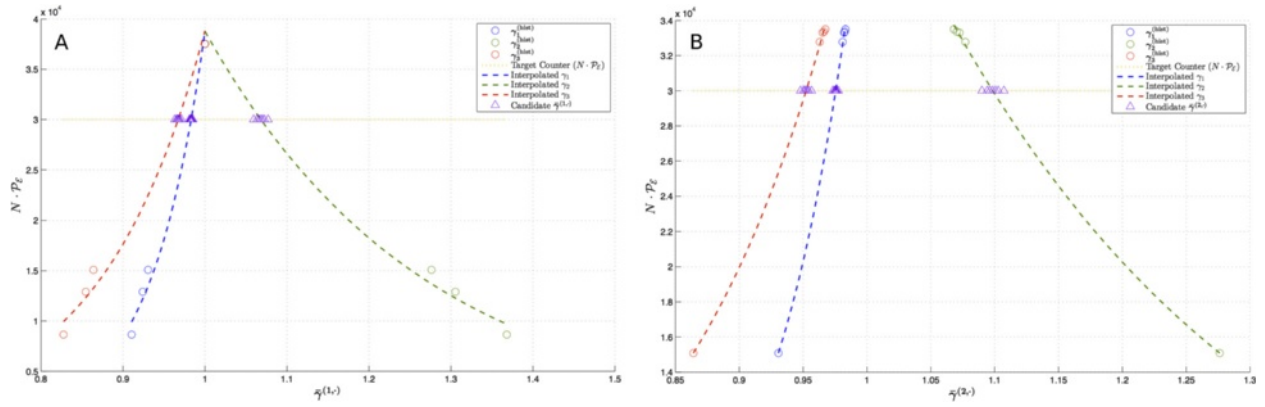

Figure 10 Illustrations of first exponential interpolation (A) and second exponential interpolation (B) for the SIRS model with $\mathcal{P}_\mathcal{E} = 0.60$, $\epsilon_{\mathcal{P}_\mathcal{E}} = 0.01$, $k^{(0)} = [0.0942\ 1.7150\ 0.6196]$, and $N = 5 \times 10^4$. Yellow horizontal dotted line is the desired number of successful trajectories, *i.e.*, $N \times \mathcal{P}_\mathcal{E} = 3 \times 10^4$. Blue, green, and red circles denote the past intermediate biasing parameters for R_1, R_2, and R_3, respectively, normalized with respect to $k^{(0)}$. Blue, green, and red dashed lines are the interpolants constructed from the past intermediate basing parameters, and the purple triangles are candidate biasing parameters computed according to Step 3 in Subsection *Updating intermediate reaction rates*.

volume of the solution hyperplane could be infinite if we do not confine parameter ranges. For the reversible isomerization process presented in the previous section, all solution hyperplanes from the 9 standard test scenarios are defined by the ratio between the two reaction rate parameters; infinitely many pairs exist that keep this ratio conserved. In addition, a range is required to sample initial reaction rates. If a user wishes to use a distribution other than the uniform distribution to generate initial reaction rates, different statistics (mean, standard deviation, *etc.*) may be needed.

We remind our readers that although parameter ranges are used to constrain the position of initial reaction rates, the same ranges are *not* enforced on the final reaction rates on the solution hyperplane. The main reason for this is that there is no guarantee the solution hyperplane intersects with the volume defined by the user-specified parameter ranges. By not limiting the final reaction rates to reside within the user-specified region, SParSE is able to find a set of reaction rates that lie on the solution hyperplane that are close to the user-specified parameter ranges. For example, in Figure 3, white dashed lines represent parameter ranges specified prior to the simulation. We see that 3 of 30 initial sets reached the solution hyperplane but are outside this region. We also see that some intermediate reaction rates (white squares) escape the region but return to it by the time $\boldsymbol{k}^*$ (red square) is found. For most practical applications, we know neither the curvature of the solution hyperplane nor the existence of it within the prescribed parameter ranges. The parameter ranges for all examples in this paper were chosen such that all possible values in (0 1) are captured while the volume of a solution hyperplane for any particular $\mathcal{P}_\mathcal{E}$ is well-defined within this region. Therefore we expect the computational gain from employing SParSE over SSA-URS to be much higher for an arbitrary problem where the user is unable to provide informative parameter ranges for the target event of interest and its desired probability.

Future work will focus on two main areas whose improvement will substantially benefit the algorithm. First, the multilevel cross-entropy method for SParSE can improve from employing an adaptive $\boldsymbol{\rho}(\delta)$ function, whose values for determining intermediate events would change as the simulation progresses. While SParSE proved to be computationally efficient for all three examples presented in this paper, their results demonstrated that the same $\boldsymbol{\rho}(\delta)$ function can produce qualitatively different behavior on how the system approaches the solution hyperplane. We can use past values of $\boldsymbol{\rho}(\delta)$ and its effect on $\hat{p}_{SParSE}$ to estimate the speed of convergence toward the solution hyperplane. This can potentially reduce the number of multilevel cross-entropy method iterations, where reduction of each iteration saves $2 \times N$ simulations. The second area of future research will be on efficient sampling of initial reaction rates. Once SParSE finishes simulating first sets of $\boldsymbol{k}^{(0)}$, positions of resulting $\boldsymbol{k}^*$ may be far away from each other and thus insufficient to construct an accurate picture of the solution hyperplane. Instead of randomly sampling the next set of initial reaction rates, we can utilize information from the prior ensemble of SParSE simulations to improve the positioning of the next set of $\boldsymbol{k}^{(0)}$. For example, we can construct a rough interpolation (*e.g.*, linear interpolation) of the solution hyperplane using $\boldsymbol{k}^*$s from the first ensemble, and sample the next set from the estimated solution hyperplane, which could be constrained by the user-specified parameter ranges if necessary. A more sophisticated method would be required

for high-dimensional systems or for target events with discontinuity in the solution hyperplane.

Additional files

Additional file 1: Appendix. Appendix is composed of three parts: tables, pseudocode, and discussion of failure in convergence. Table I contains the definition of variables used in Methods Section, and Table II provides a list of input parameters for SParSE and its default value when applicable. Table III contains the definition of variables exclusively used in pseudocode. In the second part, SParSE pseudocode is provided in a format of five separate algorithms for easy of reading and reproducibility. Lastly, we discuss in detail the three failures in Results Section.

Additional file 2: This file contains an animated illustration of SParSE simulations for Birth-death process with $\mathcal{P}_{\mathcal{E}} = 0.010$ and $\epsilon_{\mathcal{P}_{\mathcal{E}}} = 0.001$.

Additional file 3: This file contains an animated illustration of SParSE simulations for SIRS disease dynamics model with $\mathcal{P}_{\mathcal{E}} = 0.40$ and $\epsilon_{\mathcal{P}_{\mathcal{E}}} = 0.01$.

Competing interests
The authors declare that they have no competing interests.

Authors' contributions
MR conceived of the method, coded the algorithm to carry out numerical experiments, and prepared figures and tables. PE participated in the design of the numerical experiments and revising the manuscript. Both authors read and approved the final manuscript.

Acknowledgements
The authors would like to thank Bill and Melinda Gates for their active support of this work and their sponsorship through the Global Good Fund. Writing assistance from Christopher Lorton and productive discussions with colleagues at the Institute for Disease Modeling are likewise greatly appreciated.

Received: 15 April 2014 Accepted: 22 August 2014
Published online: 08 November 2014

References
1. van Kampen NG: *Stochastic Processes in Physics and Chemistry*. 3rd: Elsevier; 2007. [http://store.elsevier.com/Stochastic-Processes-in-Physics-and-Chemistry/NG-Van-Kampen/isbn-9780080475363]
2. Aldinucci M, Torquati M, Spampinato C, Drocco M, Misale C, Calcagno C, Coppo M: **Parallel stochastic systems biology in the cloud.** *Brief Bioinform* 2013, **15**(5):798–813.
3. Bunch C, Chohan N, Krintz C, Shams K: **Neptune: a domain specific language for deploying hpc software on cloud platforms.** In *Proceedings of the 2nd International Workshop on Scientific Cloud Computing*. New York: ACM; 2011:59–68.
4. Klingbeil G, Erban R, Giles M, Maini PK: **Fat versus thin threading approach on gpus: Application to stochastic simulation of chemical reactions.** *IEEE Trans Parallel Distr Syst* 2012, **23**(2):280–287.
5. Dematté L, Prandi D: **Gpu computing for systems biology.** *Brief Bioinform* 2010, **11**(3):323–333.
6. Li H, Petzold LR: **Efficient parallelization of the stochastic simulation algorithm for chemically reacting systems on the graphics processing unit.** *Int J High Perform Comput Appl* 2010, **24**(2):107–116.
7. Komarov I, D'Souza RM: **Accelerating the gillespie exact stochastic simulation algorithm using hybrid parallel execution on graphics processing units.** *PLoS ONE* 2012, **7**(11):46693.
8. Daigle B, Roh M, Petzold L, Niemi J: **Accelerated maximum likelihood parameter estimation for stochastic biochemical systems.** *BMC Bioinformatics* 2012, **13**(1):68.
9. Poovathingal S, Gunawan R: **Global parameter estimation methods for stochastic biochemical systems.** *BMC Bioinformatics* 2010, **11:**414.
10. Horvath A, Manini D: **Parameter estimation of kinetic rates in stochastic reaction networks by the em method.** *BMEI* 2008, **1**(1):713–717.
11. Wang Y, Christley S, Mjolsness E, Xie X: **Parameter inference for discretely observed stochastic kinetic models using stochastic gradient descent.** *BMC Syst Biol* 2010, **4:**99.
12. Hasenauer J, Wolf V, Kazeroonian A, Theis FJ: **Method of conditional moments (mcm) for the chemical master equation: a unified framework for the method of moments and hybrid stochastic-deterministic models.** *J Math Biol* 2013, **69**(3):687–735. doi:10–10070028501307115.
13. Yildirim N, Mackey MC: **Feedback regulation in the lactose operon: a mathematical modeling study and comparison with experimental data.** *Biophys J* 2003, **84**(5):2841–2851.
14. Griffith JS: **Mathematics of cellular control processes ii. positive feedback to one gene.** *J Theor Biol* 1968, **20**(2):209–216.
15. Vilar JMG, Guet CC, Leibler S: **Modeling network dynamics: the lac operon, a case study.** *J Cell Biol* 2003, **161**(3):471–476.
16. Klein DJ, Baym M, Eckhoff P: **The separatrix algorithm for synthesis and analysis of stochastic simulations with applications in disease modeling.** *PLoS ONE* 2014, **9**(7):103467.
17. Bernie J., Daigle J, Roh MK, Gillespie DT, Petzold LR: **Automated estimation of rare event probabilities in biochemical systems.** *J Chem Phys* 2011, **134**(4):044110.
18. Rubinstein RY: **Optimization of computer simulation models with rare events.** *Eur J Oper Res* 1997, **99**(1):89–112.
19. Gillespie DT: **Exact stochastic simulation of coupled chemical reactions.** *J Phys Chem* 1977, **81**(25):2340–2361.
20. Gillespie DT, Roh M, Petzold LR: **Refining the weighted stochastic simulation algorithm.** *J Chem Phys* 2009, **130**(17):174103.

doi:10.1186/s12918-014-0126-y
Cite this article as: Roh and Eckhoff: Stochastic parameter search for events. *BMC Systems Biology* 2014 **8**:126.