S-Leaping: An Adaptive, Accelerated Stochastic Simulation Algorithm, Bridging τ-Leaping and R-Leaping

Bulletin of Mathematical Biology (2019) 81:3074–3096
https://doi.org/10.1007/s11538-018-0464-9

Society for Mathematical Biology

SPECIAL ISSUE: GILLESPIE AND HIS ALGORITHMS

S-Leaping: An Adaptive, Accelerated Stochastic Simulation Algorithm, Bridging τ-Leaping and *R*-Leaping

Jana Lipková[1] · Georgios Arampatzis[3] · Philippe Chatelain[2] · Bjoern Menze[1] · Petros Koumoutsakos[3]

Received: 29 January 2018 / Accepted: 29 June 2018 / Published online: 10 July 2018

Abstract
We propose the *S*-leaping algorithm for the acceleration of Gillespie's stochastic simulation algorithm that combines the advantages of the two main accelerated methods; the τ-leaping and *R*-leaping algorithms. These algorithms are known to be efficient under different conditions; the τ-leaping is efficient for non-stiff systems or systems with partial equilibrium, while the *R*-leaping performs better in stiff system thanks to an efficient sampling procedure. However, even a small change in a system's set up can critically affect the nature of the simulated system and thus reduce the efficiency of an accelerated algorithm. The proposed algorithm combines the efficient time step selection from the τ-leaping with the effective sampling procedure from the *R*-leaping algorithm. The *S*-leaping is shown to maintain its efficiency under different conditions and in the case of large and stiff systems or systems with fast dynamics, the *S*-leaping outperforms both methods. We demonstrate the performance and the accuracy of the *S*-leaping in comparison with the τ-leaping and *R*-leaping on a number of benchmark systems involving biological reaction networks.

Keywords Stochastic simulation algorithms · Stiff systems · Accelerated simulation

1 Introduction

The celebrated Gillespie's stochastic simulation algorithm (SSA) (Gillespie 1976, 1977) simulates continuous-time Markov chains systems. An example of such system is a well-stirred chemically reacting system with small population of reactants (Ander-

✉ Petros Koumoutsakos
petros@ethz.ch

1 Department of Informatics, Technical University of Munich, 85748 Munich, Germany

2 Institute of Mechanics, Materials and Civil Engineering, Université catholique de Louvain, 1348 Louvain-la-Neuve, Belgium

3 Computational Science and Engineering Laboratory, ETH Zurich, 8092 Zurich, Switzerland

son and Kurtz 2011). The SSA is an exact numerical algorithm. However, since SSA allows to simulate only one reaction event per time step, it becomes computationally costly for large systems and long timescales. Over the years, several algorithms were proposed to accelerate the SSA at the expense of sacrificing its accuracy. The most prominent are the τ-leaping (Gillespie 2001) with its further enhancements (Cao et al. 2005, 2006, 2005, 2007; Rathinam et al. 2003; Tian and Burrage 2004) and the R-leaping algorithm (Auger et al. 2006; Mjolsness et al. 2009). Other accelerated algorithms involve the FLAVOR-SSA, where flow averaging is used to accelerate the simulation (Bayati et al. 2010), coupling of multi-scale frameworks with any stochastic simulation algorithm (Koumoutsakos and Feigelman 2013) and an adaptive mesh refinement algorithm for reaction–diffusion systems (Bayati et al. 2011). One can finally mention a special class of algorithms which achieve both exact, SSA-like, sampling of the reaction events and computational acceleration, as initiated by the Exact R-leaping (Mjolsness et al. 2009); the acceleration offered by such techniques is, however, weaker than in the τ-leaping and R-leaping algorithms.

The τ-leaping algorithm (Gillespie 2001) accelerates the SSA by advancing the state of the system by a larger time step τ, allowing multiple reaction events to occur within the preselected time step. The number of firings of each reaction channel at each time step is a random variable that follows Poisson distribution. On the other hand, the R-leaping algorithm preselects the total number of reaction firings L (Auger et al. 2006). The time step needed for those L reactions events to occur follows a Gamma distribution and the number of firings of each reaction follows a multinomial distribution, which can be efficiently sampled through correlated binomial distributions. Both approximate algorithms are valid under the *leap condition* which states that the propensities must remain approximately constant during each simulation step.

Each of these algorithms is efficient under different conditions. In non-stiff systems, the τ-leaping is more effective than the R-leaping algorithm. In addition, the implicit extension of the τ-leaping for stiff systems where some reaction channels appear in partial equilibrium (Cao et al. 2007), allows to advance the system with bigger time steps, which yields to significant speed-up over the explicit R-leaping method. However, the sampling procedure in the τ-leaping method requires to draw one random number for each reaction channel. This is especially inefficient in big and stiff systems, where only few reaction channels are fired per time step. On the other hand, since the samples in the R-leaping are drawn from a correlated probability distribution, the amount of drawn random numbers can be reduced by reordering the reaction indices in a way that the most probable reaction channels are sampled first. This yields appreciable computational savings in big and stiff systems.

In this paper, we present the S-leaping algorithm as an efficient coupling of both methods. Our algorithm uses the efficient time step selection procedure present in the τ-leaping. This feature allows the S-leaping to exploit the advantage of implicit formulation for stiff systems with partial equilibrium. In addition, the S-leaping estimates the total number of firings within a preselected time interval as a sample from Poisson distribution. The knowledge of the total number of reaction channels allows to draw individual firings from the correlated binomial distributions, with further optimization through reordering of reaction channels in big and stiff systems. Thus, the S-leaping algorithm provides an effective fusion of both methods. The name of the method was

chosen so that it represents the position of the S-leaping between the R-leaping and τ-leaping method.

The paper is organized as follows. In Sect. 2, we provide a brief specification of the SSA, τ-leaping and R-leaping algorithms. The S-leaping algorithm is introduced in Sect. 3. In Sect. 4, the S-leaping method is tested on four benchmark cases, a non-stiff, a stiff, a fast dynamics and a large reaction network. We conclude with a summary in Sect. 5.

2 Background

We consider a well-stirred system that contains N molecular species $\{S_1, \ldots, S_N\}$ that can react through M chemical reactions channels $\{R_1, \ldots, R_M\}$. In what follows, the letter $i \in \{1, \ldots, N\}$ will be used for chemical species, e.g., S_i, and the letter $j \in \{1, \ldots, M\}$ for chemical reactions, e.g., R_j. The state of the system is characterized by the *state vector* $\mathbf{X}(t) = (X_1(t), \ldots, X_N(t))$, where $X_i(t)$ denotes the number of molecules S_i at time t. The dynamics of each reaction channel R_j are being characterized by a *propensity function* a_j and a *state change vector* $\boldsymbol{\nu}_j = (\nu_{1j}, \ldots, \nu_{Nj})$. Given the state vector $\mathbf{x} = (x_1, \ldots, x_N) = \mathbf{X}(t)$, the quantity $a_j(\mathbf{x})dt$ gives the probability that the reaction R_j will occur in the next infinitesimal time interval $[t, t + dt)$. The state change vector $\boldsymbol{\nu}_j$ gives the change in the molecular population caused by one reaction R_j. Finally, we define a_0 as the sum of all propensity functions $a_0(\mathbf{x}) = \sum_{j=1}^{M} a_j(\mathbf{x})$.

For the rest of the paper, we will use the notation $\mathcal{B}$, Γ, $\mathcal{E}$, $\mathcal{M}$, and $\mathcal{P}$ to denote the probability distribution function of the binomial, the gamma, the exponential, the multinomial and the Poisson distribution, respectively. The same notation will be used to denote the function that produces pseudo-random numbers from the respective distribution. With $\lfloor x \rceil$, we will denote the closest integer to x.

2.1 The Stochastic Simulation Algorithm

The SSA (Gillespie 1977) is an exact algorithm for simulating the time evolution of well-stirred chemically reacting systems. It is an exact algorithm in the sense that the generated sample paths are distributed according to the solution of the corresponding chemical master equation (Gillespie 1977). However, since SSA simulates only one reaction event per time step, it becomes inefficient for most realistic systems. The SSA algorithm is summarized in Algorithm 1.

2.2 Approximate Accelerated Stochastic Simulation Algorithms

Several approximate stochastic simulation algorithms (Auger et al. 2006; Cao et al. 2006, 2005; Gillespie 2001) have been introduced to accelerate the SSA by advancing the system with larger time steps, allowing to fire more reactions per time step. The accurate advancement of the system is limited by the so-called *leap condition*, which

Algorithm 1 Stochastic Simulation Algorithm (SSA)

1: Initialise: $T_{\text{end}}, \mathbf{x} \leftarrow \mathbf{X}(0), t \leftarrow 0$
2: **while** $t < T_{\text{end}}$ **do**
3: Compute $a_j(\mathbf{x})$ for $j = 1, \ldots, M$ and $a_0(\mathbf{x})$
4: $\tau \leftarrow \mathcal{E}(1/a_0(\mathbf{x}))$
5: Choose the j-th reaction with probability $a_j(\mathbf{x})/a_0(\mathbf{x})$
6: $\mathbf{x} \leftarrow \mathbf{x} + \boldsymbol{\nu}_j$
7: $t \leftarrow t + \tau$
8: **end while**

states that propensities $a_j(\mathbf{x})$ should remain approximately constant over the time interval $[t, t+\tau)$,

$$|a_j(\mathbf{X}(t+\tau)) - a_j(\mathbf{X}(t))| \leq \varepsilon\, a_0(\mathbf{x}), \quad j = 1, \ldots, M\ , \tag{1}$$

where $0 < \varepsilon \ll 1$ is a user-defined parameter that controls the models accuracy.

2.2.1 Nonnegative τ-Leaping

The τ-leaping algorithm (Gillespie 2001) preselects a deterministic time step τ, much bigger than the mean stochastic time step of SSA. Then, the number of times $k_j^{\mathcal{P}}$ the reaction R_j will be fired during the time interval $[t, t+\tau)$ is sampled from a Poisson distribution with parameter $a_j(\mathbf{x})\tau$. Since the Poisson random variables $k_j^{\mathcal{P}}$ are unbounded, the algorithm might result in negative populations. To overcome this problem a nonnegative version of the τ-leaping algorithm was proposed in Cao et al. (2005). The algorithm identifies the critical reactions, those which are N_c firings from exhausting one of its reactants. No more than one critical reaction can occur within the time leap τ, while multiple non-critical reactions are allowed. The critical reaction is handled by the SSA, while the non-critical reactions are modeled by the τ-leaping method. Several methods (Cao et al. 2006; Gillespie 2001; Gillespie and Petzold 2003) were introduced for the computation of the leap length τ. The most efficient one (Cao et al. 2006) selects τ by

$$\tau = \min_{i \in I_{\text{rs}}} \left\{ \frac{\max\left\{\frac{\varepsilon x_i}{g_i(\mathbf{x})}, 1\right\}}{|\,\mu_i(\mathbf{x})\,|}, \frac{\max\left\{\frac{\varepsilon x_i}{g_i(\mathbf{x})}, 1\right\}^2}{|\,\sigma_i^2(\mathbf{x})\,|} \right\}\ , \tag{2}$$

for $\mathbf{x} = \mathbf{X}(t)$ and I_{rs} the set of indices of all reactant species. The factor g_i takes into account the highest order of reaction, denoted as h_i, in which species S_i appears as a reactant,

$$g_i(\mathbf{x}) = h_i + \frac{h_i}{n_i} \sum_{j=1}^{n_i - 1} \frac{j}{x_i - j}\ , \tag{3}$$

where n_i denotes the maximum number of S_i molecules required by any of the highest order reactions (Sandmann 2009). Finally, the terms μ_i and σ_i^2 are given by

$$\mu_i(\mathbf{x}) = \sum_{j \in J_{\text{ncr}}} \nu_{ij}\, a_j(\mathbf{x}), \quad \forall i \in I_{\text{rs}}\ , \tag{4}$$

$$\sigma_i^2(\mathbf{x}) = \sum_{j \in J_{\text{ncr}}} \nu_{ij}^2\, a_j(\mathbf{x}), \quad \forall i \in I_{\text{rs}},\ , \tag{5}$$

where J_{ncr} is the set of all non-critical reactions. The nonnegative τ-leaping algorithm is outlined in Algorithm 2.

Algorithm 2 Nonnegative τ-leaping

1: Initialise: $T_{\text{end}}, \mathbf{x} \leftarrow \mathbf{X}(0), t \leftarrow 0, N_{\text{c}} \leftarrow 10$.
2: **while** $t < T_{\text{end}}$ **do**
3: Compute $a_j(\mathbf{x})$ for $j = 1, \ldots, M$ and $a_0(\mathbf{x})$
4: Compute the list of critical reactions J_{crit}. The reaction R_j is critical if:

$$a_j(\mathbf{x}) > 0 \quad \text{and} \quad \min_i \left\lfloor \frac{x_i}{|\nu_{ij}|} \right\rfloor \leq N_c$$

5: Compute time the step τ_1 by Eq. (2)
6: **if** $\tau_1 < 10\frac{1}{a_0(\mathbf{x})}$ **then**
7: Execute 100 steps of the SSA
8: **else**
9: $a_0^{\text{c}}(\mathbf{x}) = \sum_{j \in J_{\text{crit}}} a_j(\mathbf{x})$ and $\tau_2 \leftarrow \mathcal{E}(1/a_0^{\text{c}}(\mathbf{x}))$ {time of critical reaction}
10: **if** $\tau_1 \leq \tau_2$ **then**
11: $\tau \leftarrow \tau_1$
12: $k_j \leftarrow \mathcal{P}(a_j(\mathbf{x})\tau), \quad j \notin J_{\text{crit}}$
13: $k_j = 0, \quad j \in J_{\text{crit}}$
14: **else**
15: $\tau \leftarrow \tau_2$
16: Choose j_{c} with probability $a_{j_{\text{c}}}(\mathbf{x})/a_0^{\text{c}}(\mathbf{x})$ and $j_c \in J_{\text{crit}}$
17: $k_{j_c} \leftarrow 1$
18: $k_j \leftarrow 0$ for $j \in J_{\text{crit}}$ and $j \neq j_{\text{c}}$
19: $k_j \leftarrow \mathcal{P}(a_j(\mathbf{x})\tau)$ for $j \notin J_{\text{crit}}$
20: **end if**
21: **end if**
22: **if** there is a negative component in $\mathbf{x} + \sum_{j=1}^{M} k_j \boldsymbol{\nu}_j$ **then**
23: $\tau_1 \leftarrow \tau/2$ and go to 6.
24: **else**
25: $\mathbf{x} \leftarrow \mathbf{x} + \sum_{j=1}^{M} k_j \boldsymbol{\nu}_j$
26: $t \leftarrow t + \tau$.
27: **end if**
28: **end while**

2.2.2 Adaptive τ-Leaping

An adaptive version of the τ-leaping algorithm was introduced in Cao et al. (2007). It automatically alternates between the explicit (Algorithm 2) and implicit τ-leaping (Rathinam et al. 2003) algorithm. The implicit τ-leaping algorithm is inspired by the implicit Euler method for differential equations. Ideally, we would like to compute the state $\mathbf{X}(t+\tau)$ as

$$\mathbf{X}(t+\tau) = \mathbf{X}(t) + \sum_{j=1}^{M} \boldsymbol{\nu}_j \, k_j^{\mathcal{P}}(\mathbf{X}(t+\tau)) \; . \tag{6}$$

However, this would require the generation of random samples from a Poisson distribution with unknown parameter,

$$k_j^{\mathcal{P}}(\mathbf{X}(t+\tau)) \sim \mathcal{P}(a_j \, (\mathbf{X}(t+\tau) \, \tau) \; .$$

To avoid this difficulty, a partial implicit approach was introduced in Rathinam et al. (2003). If $k_j^{\mathcal{P}}$ is a random variable that follows a Poisson distribution with mean $a_j\tau$, then $k_j^{\mathcal{P}}$ can be expressed as a sum of a random variable with mean $a_j\tau$ and zero mean random variable $k_j^{\mathcal{P}} - a_j\tau$,

$$k_j^{\mathcal{P}} = a_j\tau + k_j^{\mathcal{P}} - a_j\tau \; . \tag{7}$$

The partial implicit approach evaluates the variable $a_j\tau$ at the state $\mathbf{X}(t+\tau)$ and the zero mean variable $k_j^{\mathcal{P}} - a_j\tau$ at the state $\mathbf{X}(t)$. Applying this approach to the firings $k_j^{\mathcal{P}}$ in Eq. (6) leads to the following implicit system of equations,

$$\mathbf{x}' = \mathbf{x} + \sum_{j=1}^{M} \boldsymbol{\nu}_j a_j(\mathbf{x}')\tau + \sum_{j=1}^{M} \boldsymbol{\nu}_j \left(k_j^{\mathcal{P}}(\mathbf{x}) - a_j(\mathbf{x})\tau \right) \; . \tag{8}$$

for $\mathbf{x} = \mathbf{X}(t)$ and $\mathbf{x}' = \mathbf{X}(t+\tau)$. If we denote by $\mathbf{X}^\star$ the solution of the above implicit system, which can be obtained with Newton-Raphson method, the implicit state update in Eq. (6) is given by,

$$\mathbf{X}(t+\tau) = \mathbf{X}(t) + \sum_{j=1}^{M} \boldsymbol{\nu}_j k_j^{\mathcal{P}\star}, \tag{9}$$

where

$$k_j^{\mathcal{P}\star} = \left\lfloor a_j(\mathbf{X}^\star)\tau + k_j^{\mathcal{P}}(\mathbf{X}(t)) - a_j(\mathbf{X}(t))\tau \right\rceil \; . \tag{10}$$

The rounding in Eq. (10) ensures that the updated population will remain integer.

Implicit numerical methods provide an efficient way for solving stiff systems since they advance the system with bigger time steps than explicit methods. While implicit methods for differential equations are unconditionally stable, the time step in the

implicit leaping methods is bounded by the leap condition of Eq. (1). The computation of the implicit leap step τ under the condition of partial equilibrium was introduced in Cao et al. (2007). The assumption is that if some reaction channels are in equilibrium or close to a partial equilibrium, then the net change of their propensities would be small. Thus the dynamics of the system would be driven by the reactions outside the equilibrium and the implicit time step can be computed as

$$\tau^{(\mathrm{im})} = \min_{i \in I_{\mathrm{rs}}} \left\{ \frac{\max\left\{\frac{\varepsilon x_i}{g_i(\mathbf{x})}, 1\right\}}{|\mu_i^{(\mathrm{im})}(\mathbf{x})|}, \frac{\max\left\{\frac{\varepsilon x_i}{g_i(\mathbf{x})}, 1\right\}^2}{\sigma_i^{(\mathrm{im})}(\mathbf{x})^2} \right\} , \tag{11}$$

where g_i is given by Eq. (3) and $\mu_i^{(\mathrm{im})}$, $\sigma_i^{(\mathrm{im})}$ are given by

$$\mu_i^{(\mathrm{im})}(\mathbf{x}) = \sum_{j \in J_{\mathrm{necr}}} \nu_{ij}\, a_j(\mathbf{x}), \quad \forall i \in I_{\mathrm{rs}} , \tag{12}$$

$$\sigma_i^{(\mathrm{im})}(\mathbf{x})^2 = \sum_{j \in J_{\mathrm{necr}}} \nu_{ij}^2\, a_j(\mathbf{x}), \quad \forall i \in I_{\mathrm{rs}} , \tag{13}$$

for $\mathbf{x} = \mathbf{X}(t)$. Here, J_{necr} denotes the set of indices of the reaction channels that are neither critical nor in partial equilibrium.

In general, it is difficult to detect which reaction channels are currently in partial equilibrium; however, it can be easily detected for reversible reactions (Cao et al. 2007). Let R_+ and R_- denote a pair of reversible reactions, with the corresponding propensity functions a_+ and a_-. If the reaction R_+ and R_- are in partial equilibrium, their propensities must be similar,

$$|a_+(\mathbf{x}) - a_-(\mathbf{x})| \leq \delta \min\{a_+(\mathbf{x}), a_-(\mathbf{x})\} , \tag{14}$$

where δ is a small positive number, usually chosen around 0.05 (Cao et al. 2007). The adaptive τ-leaping algorithm is outlined in Algorithm 3.

2.2.3 R-Leaping

The R-leaping algorithm (Auger et al. 2006), instead of prescribing the time step, it imposes the total number of reactions L that can be fired during the next time interval. Under the leap condition of Eq. (1), the number of firings is computed as Auger et al. (2006),

$$L = a_0(\mathbf{x}) \min_{i \in I_{\mathrm{rs}}} \left\{ \frac{\max\left\{\frac{\varepsilon x_i}{g_i(\mathbf{x})}, 1\right\}}{|\,\mu_i(\mathbf{x})\,|}, \frac{\max\left\{\frac{\varepsilon x_i}{g_i(\mathbf{x})}, 1\right\}^2}{|\,\sigma_i^2(\mathbf{x})\,| - |\mu_i^2(\mathbf{x})/a_0(\mathbf{x})|} \right\} , \tag{15}$$

for $\mathbf{x} = \mathbf{X}(t)$, I_{rs} the set of indices of all reactant species and the terms g_i, $\mu_i(\mathbf{x})$ and $\sigma_i^2(\mathbf{x})$ given by Eqs. (3), (4) and (5), respectively. The time span τ_L for the L reactions

Algorithm 3 Adaptive τ-leaping

1: Initialise: $T_{\text{end}}, \mathbf{x} \leftarrow \mathbf{X}(0), t \leftarrow 0, N_{\text{c}} \leftarrow 10$.
2: **while** $t < T_{\text{end}}$ **do**
3: Compute $a_j(\mathbf{x})$ for $j = 1, \ldots, M$ and $a_0(\mathbf{x})$
4: Compute the list of critical reactions J_{crit}. The reaction R_j is critical if:

$$a_j(\mathbf{x}) > 0 \quad \text{and} \quad \min_i \left\lfloor \frac{x_i}{|\nu_{ij}|} \right\rfloor \leq N_c$$

5: Compute $\tau^{(\text{ex})}$ using Eq. (2) and $\tau^{(\text{im})}$ using Eq. (11)
6: **if** $\tau^{(\text{im})} > 100\, \tau^{(\text{ex})}$ **then**
7: The system is stiff and $\tau_1 \leftarrow \tau^{(\text{im})}$
8: **else**
9: The system is non-stiff and $\tau_1 \leftarrow \tau^{(\text{ex})}$
10: **end if**
11: **if** $\tau_1 \leq 10 \frac{1}{a_0(\mathbf{x})}$ **then**
12: Execute 100 steps of the SSA.
13: **else**
14: $a_0^c(\mathbf{x}) = \sum_{j \in J_{\text{crit}}} a_j(\mathbf{x})$ and $\tau_2 \leftarrow \mathcal{E}(1/a_0^c(\mathbf{x}))$ {time of critical reaction}
15: **if** $\tau_2 > \tau_1$ **then**
16: $\tau \leftarrow \tau_1$
17: **if** the system is currently stiff **then**
18: Compute k_j using Eq. (10) for $j \notin J_{\text{crit}}$
19: **else**
20: $k_j \leftarrow \mathcal{P}(a_j(\mathbf{x}))$ for $j \notin J_{\text{crit}}$
21: **end if**
22: **else**
23: $\tau \leftarrow \tau_2$
24: Choose j_{c} with probability $a_{j_{\text{c}}}(\mathbf{x})/a_0^c(\mathbf{x})$ and $j_c \in J_{\text{crit}}$
25: $k_{j_c} \leftarrow 1$
26: $k_j \leftarrow 0$ for $j \in J_{\text{crit}}$ and $j \neq j_{\text{c}}$
27: **if** $\tau_2 < \tau^{ex}$ **or** the system is non-stiff **then**
28: $k_j \leftarrow \mathcal{P}(a_j(\mathbf{x})\tau)$ for $j \notin J_{\text{crit}}$
29: **else**
30: Compute k_j using Eq. (10) for $j \notin J_{\text{crit}}$
31: **end if**
32: **end if**
33: **if** there is a negative component in $\mathbf{x} + \sum_{j=1}^{M} k_j \boldsymbol{\nu}_j$ **then**
34: $\tau_1 \leftarrow \tau/2$ and go to 11
35: **else**
36: $\mathbf{x} \leftarrow \mathbf{x} + \sum_{j=1}^{M} k_j \boldsymbol{\nu}_j$
37: $t \leftarrow t + \tau$
38: **end if**
39: **end if**
40: **end while**

follows the gamma distribution, $\tau_L \sim \Gamma(L, 1/a_0(\mathbf{x}))$. The number of firings $k_j^{\mathcal{B}}$ for the reaction channel R_j, fired within the time span τ_L, is sampled from a sequence of correlated binomial distributions,

$$k_j^{\mathcal{B}} \sim \mathcal{B}\left(L - \sum_{m=1}^{j-1} k_m^{\mathcal{B}}, \frac{a_j(\mathbf{x})}{a_0(\mathbf{x}) - \sum_{m=1}^{j-1} a_m(\mathbf{x})}\right) . \tag{16}$$

This approach requires at most $M-1$ drawings of random numbers since $k_M^{\mathcal{B}} = L - \sum_{j=1}^{M-1} k_j^{\mathcal{B}}$. Furthermore, it can be shown that the sampling procedure is invariant under the permutation of reaction channels indices (Auger et al. 2006). This fact can be exploited to reduce the number of samples drawn per time step by reordering the reactions indices in a way that the most probable reactions channels are sampled first. The R-leaping algorithm is summarized in Algorithm 4.

The sampling of reaction channels from the bounded binomial distribution reduces the appearance of negative species, compared to sampling from the unbounded Poisson distribution. However, in systems involving species with population close to zero taking place in very fast reactions, the R-leaping algorithm might also introduce negative population. To control the appearance of negative population, an additional bounding condition for L was proposed (Auger et al. 2006). In systems with high rejection rates of the proposed state update, the total number of firings is computed as $L = \min(L', L'')$, where L' is given by Eq. (15) and

$$L'' = \min_{j=1,\ldots,M} \left(1 - \theta\left(1 - \frac{a_0(\mathbf{x})}{a_j(\mathbf{x})}\right)\right) L_j , \tag{17}$$

where

$$L_j = \min_{\substack{i=1,\ldots,N \\ \nu_{ij}<0}} \left\lfloor \frac{x_i}{|\nu_{ij}|} \right\rceil . \tag{18}$$

The parameter θ controls appearance of negative species. Smaller values of θ lead to better control of negative species but also lead to lower performance.

3 S-Leaping

Here, we propose the S-leaping, an algorithm which combines the advantages of the τ-leaping and R-leaping algorithms. The S-leaping couples the efficient time step selection of the τ-leaping with the effective binomial sampling of the R-leaping algorithm. The coupling of the algorithms is achieved in the following way. First, the time step τ is selected according to Eq. (2). Then the total number of firings L that will take place in the time interval $[t, t+\tau)$ is estimated. Since in the τ-leaping each reaction channel is independently sampled as $k_j \sim \mathcal{P}(a_j(\mathbf{x})\tau)$, the total amount of all firings L follows the Poisson distribution with parameter $a_0(\mathbf{x})\tau$, i.e.,

$$L(t) \sim \mathcal{P}(a_0(\mathbf{x})\tau) , \tag{19}$$

Algorithm 4 R-Leaping

Initialise: T_{end}, $\mathbf{x} \leftarrow \mathbf{X}(0)$, $t \leftarrow 0$, $steps \leftarrow 0$, $p \leftarrow$ frequency of reordering.
while $t < T_{\text{end}}$ **do**
  Compute $a_j(\mathbf{x})$ for $j = 1, \ldots, M$ and $a_0(\mathbf{x})$
  if mod $(steps, p) = 0$ **then**
    Reorder the reactions such that $a_1(\mathbf{x}) \geq a_2(\mathbf{x}) \geq \ldots \geq a_M(\mathbf{x})$
  end if
  Compute L by Eq. (15), then set $L \leftarrow \max(L, 1)$
  Sample k_j by Eq. (16)
  if there is a negative component in $\mathbf{x} + \sum_{j=1}^{M} k_j \boldsymbol{\nu}_j$ **then**
    $L \leftarrow L/2$ and go to 8.
  else
    $\tau \leftarrow \Gamma(L, 1/a_0(\mathbf{x}))$
    $\mathbf{x} \leftarrow \mathbf{x} + \sum_{j=1}^{M} k_j \boldsymbol{\nu}_j$
    $t \leftarrow t + \tau$
    $steps = steps + 1$
  end if
end while

for $\mathbf{x} = \mathbf{X}(t)$. Knowing the number of reactions that will take place in $[t, t + \tau)$, the firings of each channel k_j can be sampled from the binomial distribution given by Eq. (16). If the sampled L is zero, it means the system will advance to the time $t = t + \tau$ without any changes since no reaction will be fired. In this case the system can be further advanced by setting $L = 1$ and $\tau \sim \Gamma(1, 1/a_0(\mathbf{x}))$ and proceeding with the S-leaping algorithm. Notice that this is just one step of SSA since the Γ distribution with parameters 1 and $1/a_0(\mathbf{x})$ is equal to exponential distribution with parameter $1/a_0(\mathbf{x})$. The S-leaping algorithm is summarized in Algorithm 5.

To control the appearance of the negative species, the S-leaping algorithm can inherit the control mechanism from the τ-leaping or R-leaping. Here we bound L similarly as in the R-leaping method. In systems with high rejections rates, the total amount of firings is computed as $L = \min(L', L'')$, where L' is given by Eq. (19) and L'' by Eq. (17). If L'' was chosen, then the time step τ should be recomputed as $\tau \sim \Gamma(L, 1/a_0(\mathbf{x}))$.

Thanks to the coupling of the two algorithms, the S-leaping performs always as well as the τ-leaping or R-leaping algorithm. In the non-stiff systems, the S-leaping benefits from the efficient time step selection and might outperform the R-leaping method. On the other hand, in the case of big and stiff systems, the S-leaping surpasses the τ-leaping due to the effective sampling of the reaction channels. Moreover, since the behavior of the system might change over time, the S-leaping can outperform both the R-leaping and τ-leaping. Finally, since the S-leaping uses the same time step selection as the τ-leaping, the algorithm can easily be extended to an adaptive explicit-implicit version.

Algorithm 5 S-Leaping

1: Initialise: T_{end}, $\mathbf{x} \leftarrow \mathbf{X}(0)$, $t \leftarrow 0$, $steps \leftarrow 0$, $p \leftarrow$ frequency of reordering.
2: **while** $t < T_{\text{end}}$ **do**
3: Compute $a_j(\mathbf{x})$ for $j = 1, \ldots, M$ and $a_0(\mathbf{x})$
4: **if** mod $(steps, p) = 0$ **then**
5: Reorder the reactions such that $a_1(\mathbf{x}) \geq a_2(\mathbf{x}) \geq \ldots \geq a_M(\mathbf{x})$
6: **end if**
7: Compute τ by Eq. (2)
8: Sample L by Eq. (19)
9: **if** L=0 **then**
10: $t \leftarrow t + \tau$
11: $L \leftarrow 1$ and $\tau \leftarrow \Gamma(1, 1/a_0(\mathbf{x}))$
12: **end if**
13: Sample k_j by Eq. (16)
14: **if** there is a negative component in $\mathbf{x} + \sum_{j=1}^{M} k_j \boldsymbol{\nu}_j$ **then**
15: $\tau \leftarrow \tau/2$ and go to 8
16: **else**
17: Update: $\mathbf{x} \leftarrow \mathbf{x} + \sum_{j=1}^{M} k_j \boldsymbol{\nu}_j$
18: $t \leftarrow t + \tau$
19: $steps = steps + 1$.
20: **end if**
21: **end while**

3.1 Adaptive S-Leaping

The adaptive leap methods switch between explicit (Algorithm 5) and implicit method depending on the stiffness of the system. The implicit S-leaping method updates the system state as,

$$\mathbf{X}(t+\tau) = \mathbf{X}(t) + \sum_{j=1}^{M} \boldsymbol{\nu}_j \, k_j^{\mathcal{B}} \left(\mathbf{X}(t+\tau)\right) \ . \tag{20}$$

This requires sampling random numbers $k_j^{\mathcal{B}}(\mathbf{X}(t+\tau))$ from the binomial distribution $\mathcal{B}(\alpha(\mathbf{x}'), \beta(\mathbf{x}'))$ with mean and variance evaluated at the unknown state $\mathbf{x}' = \mathbf{X}(t+\tau)$ given by,

$$\alpha(\mathbf{x}') = L(\mathbf{x}') - \sum_{m=1}^{j-1} k_m^{\mathcal{B}}(\mathbf{x}') \ ,$$

$$\beta(\mathbf{x}') = \frac{a_j(\mathbf{x}')}{a_0(\mathbf{x}') - \sum_{m=1}^{j-1} a_m(\mathbf{x}')} \ .$$

In the implicit τ-leaping algorithm, each firing $k_j^{\mathcal{P}}$ is independently approximated by the partially implicit formulation given by Eq. (7). This can not be directly applied in

the S-leaping, since each sample $k_j^{\mathcal{B}}$ depends on all previously drawn samples $k_\ell^{\mathcal{B}}$, $\ell = 1, 2, \ldots, j-1$. The partially implicit treatment for the S-leaping can be obtained by rather considering the distribution of the whole vector of all firings $(k_1, \ldots, k_M)$, i.e., the multinomial distribution with parameter $(\frac{a_1(\mathbf{x})}{a_0(\mathbf{x})}, \ldots, \frac{a_M(\mathbf{x})}{a_0(\mathbf{x})})$ and L the number of trials. If $k_j^{\mathcal{M}}$ is the j-th entry of a random vector that follows the multinomial distribution with parameters $(\frac{a_1(\mathbf{x})}{a_0(\mathbf{x})}, \ldots, \frac{a_M(\mathbf{x})}{a_0(\mathbf{x})})$ and L, then $k_j^{\mathcal{M}}$ can be expressed as the sum of a random variable with mean $\frac{a_j(\mathbf{x})}{a_0(\mathbf{x})}L$ and the zero mean variable $k_j^{\mathcal{M}} - \frac{a_j(\mathbf{x})}{a_0(\mathbf{x})}L$, i.e.,

$$k_j^{\mathcal{M}} = \frac{a_j(\mathbf{x})}{a_0(\mathbf{x})}L + k_j^{\mathcal{M}} - \frac{a_j(\mathbf{x})}{a_0(\mathbf{x})}L \ . \tag{21}$$

The variable $\frac{a_j}{a_0}L$ is evaluated at the unknown state $\mathbf{X}(t+\tau)$, while the variable $k_j^{\mathcal{M}} - \frac{a_j}{a_0}L$ is evaluated at the known state $\mathbf{X}(t)$. The partial implicit approximation to the variables $k_j^{\mathcal{M}}$ leads to the following system of implicit equations,

$$\mathbf{x}' = \mathbf{x} + \sum_{j=1}^{M} \boldsymbol{\nu}_j \frac{a_j(\mathbf{x}')}{a_0(\mathbf{x}')}L(t+\tau) + \sum_{j=1}^{M} \boldsymbol{\nu}_j \left(k_j^{\mathcal{M}}(\mathbf{x}) - \frac{a_j(\mathbf{x})}{a_0(\mathbf{x})}L(t) \right) \ , \tag{22}$$

for $\mathbf{x} = \mathbf{X}(t)$ and $\mathbf{x}' = \mathbf{X}(t+\tau)$. Since the multinomial random vectors $k_j^{\mathcal{M}}$ in Eq. (22) depend on the known state $\mathbf{x}$ and since the j-th element of the multinomial distribution follows binomial distribution, $k_j^{\mathcal{M}}(\mathbf{x})$ are computed by Eq. (16). However, $L(t+\tau)$ is also a random variable from Poisson distribution with the parameter evaluated at the unknown state $\mathbf{X}(t+\tau)$,

$$L(t+\tau) \sim \mathcal{P}(a_0(\mathbf{X}(t+\tau))\tau) \ . \tag{23}$$

The term $L(t+\tau)$ could be expressed in the partial implicit manner following Eq. (7). However, a simpler formulation can be obtain by a mean approximation,

$$L(t+\tau) \approx a_0(\mathbf{X}(t+\tau))\tau \ , \tag{24}$$

which corresponds to the computation of the L in the R-leaping method. The advantage of the mean approximation in Eq. (24) is that it significantly reduces numerical complexity of the implicit system in Eq. (22), while the partial implicit approximation provided by Eq. (7) would increase the complexity even more. Since we are dealing with stiff system, increased complexity could reduce accuracy of the numerical methods used for solving the implicit system of equations. If $\mathbf{X}^\star$ is the solution of the implicit system of Eq. Eq. (22), then the implicit update is obtained as

$$\mathbf{X}(t+\tau) = \mathbf{X}(t) + \sum_{j=1}^{M} \boldsymbol{\nu}_j k_j^{\mathcal{M}\star} \ , \tag{25}$$

where

$$k_j^{\mathcal{M}\star} = \left\lfloor a_j(\mathbf{X}^\star)\tau + k_j^{\mathcal{M}}(\mathbf{X}(t)) - \frac{a_j(\mathbf{X}(t))}{a_0(\mathbf{X}(t))} L(t) \right\rceil . \tag{26}$$

This means that both, the implicit τ-leaping and implicit S-leaping algorithm solve the implicit system with the same computational complexity. However, since the implicit S-leaping can exploit reordering of reaction channels, it might result in less random number generations (at most M samples) than the implicit τ-leaping (always M samples). This might allow the implicit S-leaping to outperform the implicit τ-leaping, especially in large stiff systems where only few reaction channels are fired per time step. The adaptive S-leaping method is summarized in Algorithm 6.

Algorithm 6 Adaptive S-Leaping

Initialise: T_{end}, $\mathbf{x} \leftarrow \mathbf{X}(0)$, $t \leftarrow 0$, $steps \leftarrow 0$, $p \leftarrow$ frequency of reordering.
while $t < T_{\text{end}}$ **do**
  Compute $a_j(\mathbf{x})$ for $j = 1, \ldots, M$ and $a_0(\mathbf{x})$
  if mod $(steps, p) = 0$ **then**
    Reorder the reactions such that $a_1(\mathbf{x}) \geq a_2(\mathbf{x}) \geq \ldots \geq a_M(\mathbf{x})$
  end if
  Compute $\tau^{(\text{ex})}$ by Eq. (2) and $\tau^{(\text{im})}$ by Eq. (11)
  if $\tau^{(\text{im})} > 100\ \tau^{(\text{ex})}$ **then**
    System is stiff and $\tau \leftarrow \tau^{(\text{im})}$
  else
    System is non-stiff and $\tau \leftarrow \tau^{(\text{ex})}$
  end if
  if the system is currently non stiff **then**
    Compute L by Eq. (19)
    if L=0 **then**
      $t \leftarrow t + \tau$
      $L \leftarrow 1$ and $\tau \leftarrow \Gamma(1, 1/a_0(\mathbf{x}))$
    end if
    Sample k_j by Eq. (16)
  else
    Compute k_j by Eq. (26), where $L(t)$ is given by Eq. (19)
  end if
  if there is a negative component in $\mathbf{x} + \sum_{j=1}^{M} k_j \boldsymbol{\nu}_j$ **then**
    $\tau \leftarrow \tau/2$ and go to 13
  else
    $\mathbf{x} \leftarrow \mathbf{x} + \sum_{j=1}^{M} k_j \boldsymbol{\nu}_j$
    $t \leftarrow t + \tau$
    $steps = steps + 1$.
  end if
end while

4 Numerical Simulations

To demonstrate the efficiency of the S-leaping algorithm, it is studied in comparison with the τ-leaping and R-leaping methods on four reaction networks. The first one is a non-stiff system simulating decaying dimerization. The second system is a stiff decaying dimerization with reversible reaction channels in partial equilibrium. The third one is a system with very fast dynamics simulating the evolution of *Bacillus subtilis*. The last one is a LacZ/LacY system, which consists of a relatively large amount of reactions and which stiffness change over time. For each reaction network and each algorithm, we measure two quantities: the error and the execution time of the algorithm.

The error is measured as follows. For 25 equally distributed time points and all species we estimate the distance between the distributions of the tested algorithm and the SSA (Cao et al. 2006). Since the methods do not advance the system with a fixed time step, the population at a given time is approximated by the population at the closest time where the algorithm has landed. The distance d between two distributions P and Q is approximated by the estimated histogram as

$$d = \Delta \sum_k |\tilde{P}(k) - \tilde{Q}(k)| \,, \tag{27}$$

where Δ is the bin size and $\tilde{P}(k)$, $\tilde{Q}(k)$ are the values of the histogram for P and Q at the k-th bin. The histogram is computed using $N_s = 10^4$ independent trajectories and $K = 10$ number of bins. Finally, the average error over all time points and all species is reported. This definition of the error can be interpreted as a global error, since it accounts for temporal and interspecies error of the algorithm. Averaging the error over many time points takes into account the error not only at equilibrium but at transient regimes as well.

Cao and Petzold (2006) introduced the histogram self-distance as a measure of accuracy of the histogram distance estimation. Any estimate below the value of self-distance should be considered inaccurate. A bound for the self-distance was derived in Cao and Petzold (2006) and is given by $\sqrt{4K/(\pi N_s)}$. In all the histogram error plots, we show the self-distance estimate as a constant blue line. Although the errors close or below this line should not be considered accurate we present them for completeness.

The execution time is averaged over 10 independent runs. The ratio between the execution time of SSA and the execution time of each algorithm is reported as a speed-up. Note that here the SSA is used only as a reference in order to compare the relative speed-up of the three approximate algorithm, the τ-, the R- and S-leaping. The execution times of the three methods are compared using as a reference the execution time of SSA. Hence changing the base implementation of SSA will not affect these comparisons results. Moreover, since we report the speed-up over the baseline SSA rather than CPU time, the presented results do not depend on the type of the used computer.

All reaction networks discussed in this section follow the law of mass action which states that the rate of a reaction is proportional to the product of the concentrations of the reactants. For example, for the reaction $2S_1 + S_2 \to S_3$ with reaction rate c, the propensity is defined as,

Table 1 The reaction network for the Dimerization system studied in Sects. 4.1 and 4.2

	Reaction	Reaction rate (non-stiff)	Reaction rate (stiff)
R_1	$S_1 \longrightarrow \emptyset$	1	1
R_2	$S_1 + S_1 \longrightarrow S_2$	0.002	10
R_3	$S_2 \longrightarrow S_1 + S_1$	0.5	1000
R_4	$S_2 \longrightarrow S_3$	0.04	0.1

$$a(\mathbf{x}) = c\, x_1\, (x_1 - 1)\, x_2\ . \tag{28}$$

For the general formula of the propensity function under the law of mass action we refer to Anderson and Kurtz (2011), Erban et al. (2007).

All the tested methods are implemented in the C++ language, using the C++11 random number generator library and the code is publicly available.[1]

4.1 Non-stiff Decaying Dimerization

Following the same test as in Auger et al. (2006), Gillespie (2001), Gillespie and Petzold (2003) we consider the non-stiff decaying dimerization system summarized in Table 1. The initial populations are $\mathbf{X}(0) = (4150, 39565, 3445)$ and the system is evolved until $T_{\text{end}} = 10$ using $\varepsilon = \{0.01, 0.03, 0.05\}$. In Fig. 1, we show the convergence of the histograms of the approximate algorithms to that of the SSA for the second species S_2 at time $t = 10$. All the approximate methods converge to the SSA solution as the accuracy parameter ε decreases. Figure 2 shows the accuracy (left) and the performance (right) for all leap methods with different accuracy parameter ε. In this system, all the leaping methods have comparable accuracy and performance. No additional speed-up was obtained by reordering of the reaction channels in the R-leaping and S-leaping, since in each step of the simulation all reaction channels are fired.

4.2 Stiff Decaying Dimerization

To study the efficiency of the adaptive S-leaping method we consider the stiff decaying dimerization system studied in Cao et al. (2007), Rathinam et al. (2003). The system is defined by the same set of reactions and initial conditions as in Sect. 4.1, see Table 1. The stiffness arises from the reaction rates that vary by a few orders of magnitude. The behavior of this system changes over time starting with a non-stiff phase. However, once the reversible reactions R_2 and R_3 approach the equilibrium, the system becomes stiff. Under this set up S_1 and S_2 are the fast variables, while S_3 is the slow variable. The system is evolved until the final time $T_{\text{end}} = 10$ for $\varepsilon = \{0.01, 0.03, 0.05\}$.

In Fig. 3, we present the accuracy and the performance of the adaptive τ-leaping and adaptive S-leaping as well as the explicit R-leaping, τ-leaping and S-leaping. All explicit methods reach comparable accuracy and performance. The adaptive methods

[1] https://github.com/JanaLipkova/SSM

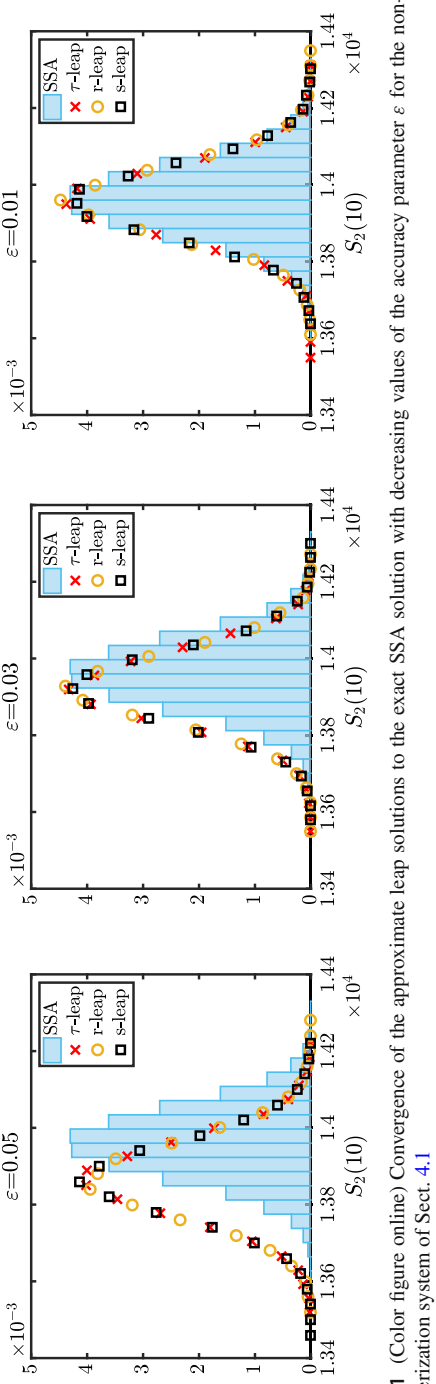

Fig. 1 (Color figure online) Convergence of the approximate leap solutions to the exact SSA solution with decreasing values of the accuracy parameter ε for the non-stiff dimerization system of Sect. 4.1

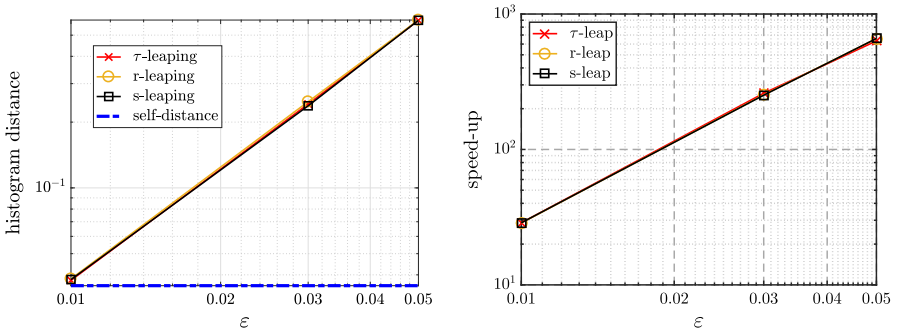

Fig. 2 (Color figure online) Errors and efficiency for non-stiff dimerization system of Sect. 4.1

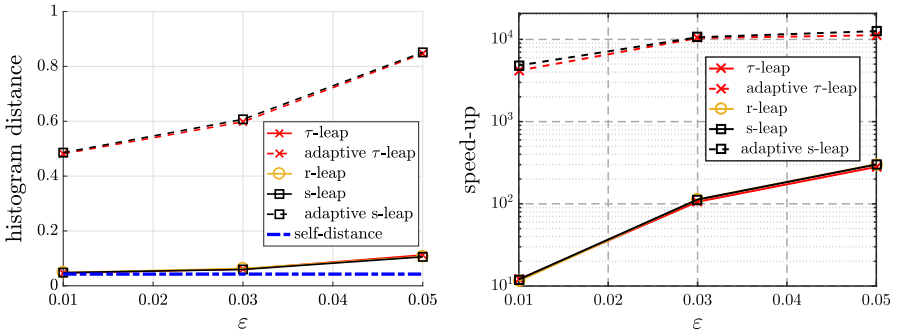

Fig. 3 (Color figure online) Errors and efficiency for stiff dimerization system discussed in Sect. 4.2

provide significant speed-up over their explicit counterparts. The reduced accuracy of the adaptive methods arises from the dumping effect of the implicit methods on the fast variables. As reported in Rathinam et al. (2003), the implicit schemes capture the distribution of the slow variable S_3 correctly. However, for the fast variables S_1 and S_2, the mean is computed correctly but the histogram distribution around the mean is too narrow. Rathinam et al. (2003) proposed a downshifting strategy to restore the natural fluctuations in the fast variables by simulating the final time steps of the adaptive method with the explicit method. As shown in Cao et al. (2005), Rathinam et al. (2003), the downshifting leads to correct histogram distributions for all variables at the final time. Since the downshifting procedure corrects the dumping effect only in the final time, the global error of the adaptive method will not be reduced to the level of the explicit methods. Since we report the global error, the downshifting strategy was not applied here. However, the downshifting procedure can be used to increase the accuracy of the adaptive τ-leaping and adaptive S-leaping method at the fixed time point.

4.3 *Bacillus subtilis*

This system describes the cellular differentiation dynamics of the *Bacillus subtilis* which exhibits stochastic behavior at the single-cell level (Maamar et al. 2007; Süel et al. 2006). The differentiation dynamics depends on the expression of the transcrip-

Table 2 The reaction network for the *Bacillus subtilis* system studied in Sect. 4.3

	Reaction	Reaction rate
R_1	$\emptyset \longrightarrow S_1 + 3S_3$	1.51×10^{-1}
R_2	$S_1 + S_2 \longrightarrow 4S_3$	3.1×10^{-4}
R_3	$S_2 \longrightarrow 4S_3$	3.4×10^{-3}
R_4	$S_3 \longrightarrow S_1 + S_2$	2.0×10^{-2}
R_5	$S_1 + 2S_2 \longrightarrow \emptyset$	6.2×10^{-5}
R_6	$2S_1 \longrightarrow S_1 + S_2$	4.9×10^{-4}

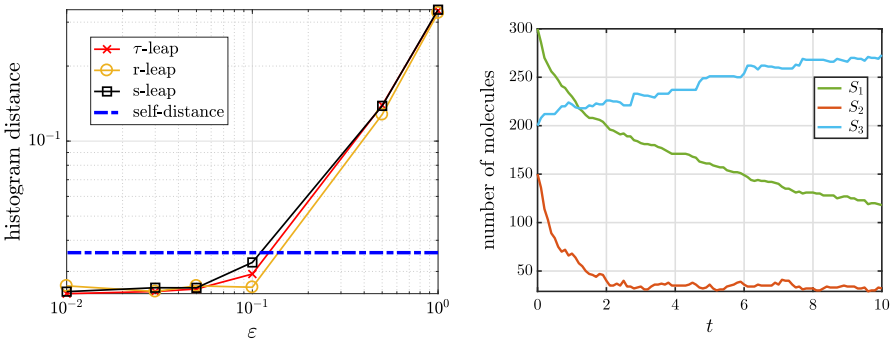

Fig. 4 (Color figure online) Error of the leap methods (left) and a single trajectory of the *Bacillus subtilis* computed with SSA (right)

Table 3 The averaged number of simulation steps and the execution time for the *Bacillus subtilis* system with the $\varepsilon = 0.05$ presented in Sect. 4.3

Method	Average number of steps	Average CPU time (s)
SSA	266.6	0.028
τ-leap	423.4	0.086
R-leap	263.2	0.053
S-leap	220.8	0.045

tional genes S_1=Spo0A, S_2=ComG and S_3=sinI and the reaction network is presented in Table 2 (Chattopadhyay et al. 2013). The system is evolved until the final time $T_{\text{end}} = 10$ with initial population $\mathbf{X}(0) = (300, 150, 200)$. Figure 4 (right) shows a single realization of the *Bacillus subtilis* system computed with SSA. This system exhibits diverse reaction rates and very fast dynamics. As a consequence, the leap methods are strongly restricted by the leap condition and do not provide additional speed-up over SSA. Table 3 shows the average number of steps executed by each method and the averaged CPU time for $\varepsilon = 0.05$. The *R*-leaping algorithm advances the system only with one reaction per time step, emulating the SSA. Moreover, the τ-leaping executes two times more steps than SSA. In this system, the *S*-leaping is the only method which requires less steps than SSA. Since in this case the leap methods do not provide additional speed-up, the SSA alone would be the best choice. This example, however, shows that the *S*-leaping maintains its performance even in

Table 4 The reaction network for the LacZ/LacY system discussed in Sect. 4.4

	Reaction	Reaction rate
R_1	PLac + RNAP $\longrightarrow$ PLacRNAP	0.17
R_2	PLacRNAP $\longrightarrow$ PLac + RNAP	10
R_3	PLacRNAP $\longrightarrow$ TrLacZ1	1
R_4	TrLacZ1 $\longrightarrow$ RbsLacZ + PLac + TrLacZ2	1
R_5	TrLacZ2 $\longrightarrow$ TrLacY1	0.015
R_6	TrLacY1 $\longrightarrow$ RbsLacY + TrLacY2	1
R_7	TrLacY2 $\longrightarrow$ RNAP	0.36
R_8	Ribosome + RbsLacZ $\longrightarrow$ RbsribosomeLacZ	0.17
R_9	RbsribosomeLacZ $\longrightarrow$ Ribosome + RbsLacZ	0.45
R_{10}	Ribosome + RbsLacY $\longrightarrow$ RbsribosomeLacY	0.17
R_{11}	RbsribosomeLacY $\longrightarrow$ Ribosome + RbsLacY	0.45
R_{12}	RbsribosomeLacZ $\longrightarrow$ TrRbsLacZ + RbsLacZ	0.4
R_{13}	RbsribosomeLacY $\longrightarrow$ TrRbsLacY + RbsLacY	0.4
R_{14}	TrRbsLacZ $\longrightarrow$ LacZ	0.015
R_{15}	TrRbsLacY $\longrightarrow$ LacY	0.036
R_{16}	LacZ $\longrightarrow$ dgrLacZ	6.42×10^{-5}
R_{17}	LacY $\longrightarrow$ dgrLacY	6.42×10^{-5}
R_{18}	RbsLacZ $\longrightarrow$ dgrRbsLacZ	0.3
R_{19}	RbsLacY $\longrightarrow$ dgrRbsLacY	0.3
R_{20}	LacZ + lactose $\longrightarrow$ LacZlactose	9.52×10^{-5}
R_{21}	LacZlactose $\longrightarrow$ product + LacZ	431
R_{22}	LacY $\longrightarrow$ lactose + LacY	14

fast dynamical systems and outperforms the other leap methods. All methods reach comparable accuracy as shown in Fig. 4 (left).

4.4 LacZ/LacY

In this section, we consider the LacZ/LacY model which describes the expression of the LacZ and LacY genes and the activity of LacZ and LacY proteins in *Escherichia Coli* (Kierzek 2002). The reaction network consists of 22 reactions and 23 species. We present the reaction network, along with the reaction rate of each reaction in Table 4. The propensity functions of this system vary by a few orders of magnitude making the system stiff. Moreover, the reaction system is considered inside a growing cell, with generation time $T_{\text{gen}} = 2100$. The growing volume changes the stiffness of the system over time since the propensities of the second and higher order reactions have to be rescaled by the volume. We consider two different initial conditions. In the first case we assume a small initial population where all species are initially 0 except for PLac=1. In the second case, we consider bigger initial populations with all species initialized at 50 and PLac=100. In addition, the number of the species RNAP and ribosome

are sampled every time step from a normal distribution $\mathcal{N}(35(1 + t/T_{\text{gen}}), 3.5^2)$ and $\mathcal{N}(350(1 + t/T_{\text{gen}}), 35^2)$, respectively, for each case. The role of the system with small initial population is to investigate the behavior of all methods in the presence of negative population, while the behavior without the appearance of negative populations is studied in the system with the bigger initial population.

The system with small initial population is simulated until $T_{\text{end}} = 2100$. Since none of the reversible reactions approached partial equilibrium during this time interval, only explicit methods are reported. To control the appearance of negative species, the τ-leaping algorithm is used with control parameter $N_c = 10$ (Cao et al. 2005), while in the *S*-leaping and *R*-leaping we used $\theta = 0.1$ as suggested in Auger et al. (2006). For comparison purposes, all three methods are also considered without the control mechanism. The frequency of reordering in the *R*-leaping and *S*-leaping is set to $p = 10000$ as proposed in Auger et al. (2006). Figure 5 (right) shows the speed-up for the leap methods over SSA for $T_{\text{end}} = 2100$. A single evaluation of the SSA for time $T_{\text{end}} = 2100$ takes around 45 min, making the evaluation of the models accuracy at this time point computationally expensive. Instead, Fig. 5 (left) reports the error for all methods over the time interval [0, 100]. For this system, the error is averaged over the species TrLacZ2, TrRbsLacZ, and RbsribsomeLacY.

The τ-leaping algorithm, as presented in Algorithm 2, executes mainly SSA steps and provides almost no speed-up over SSA. Therefore, we turned off the SSA execution in the reported τ-leaping algorithms. The leap methods without the control mechanism provide better speed-up over SSA; however, their accuracy is reduced due to the high rejection rate. The sampling of reaction channels from the correlated binomial distribution in the *R*-leaping and *S*-leaping leads to lower rejection rate in comparison with the τ-leaping, which is also reflected by the lower accuracy of the τ-leaping method.

The control mechanisms in all leap methods result in high accuracy, at the cost of slightly reduced performance. The error reported in Fig. 5 (left) is relatively constant and do not scale with ε, since the accuracy of these leap methods is mainly restricted by the mechanism preventing appearance of the negative species. The *S*-leaping reached comparable accuracy with the *R*-leaping, since they both use similar control mechanisms. On the other hand, the τ-leaping considers most reactions critical and thus advance them with SSA, which lead to higher accuracy. The *R*-leaping and *S*-leaping algorithms benefit from the reordering of reaction channels and outperform the τ-leaping. Moreover, since the stiffness of the system changes over time, the *S*-leaping outperforms both methods.

The system with big initial population is evolved until time $T_{\text{end}} = 100$. As before, the τ-leaping with the SSA steps performs mostly SSA and therefore the SSA step was disabled. Since all species appear in relatively large populations, the leap methods are considered without the control of negative population. The performance and accuracy of all methods is shown in Fig. 6. The *S*-leaping algorithm again outperforms both the τ-leaping and *R*-leaping method due to the combined advantages inherited from the both methods.

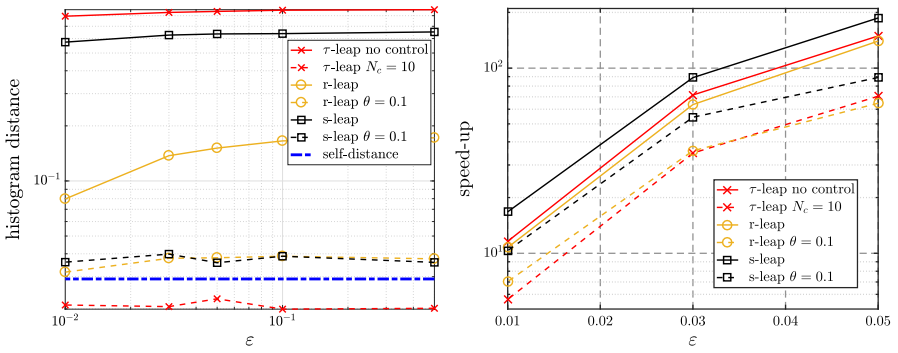

Fig. 5 (Color figure online) Errors and efficiency for the LacZ/LacY system discussed in Sect. 4.4, with small initial population

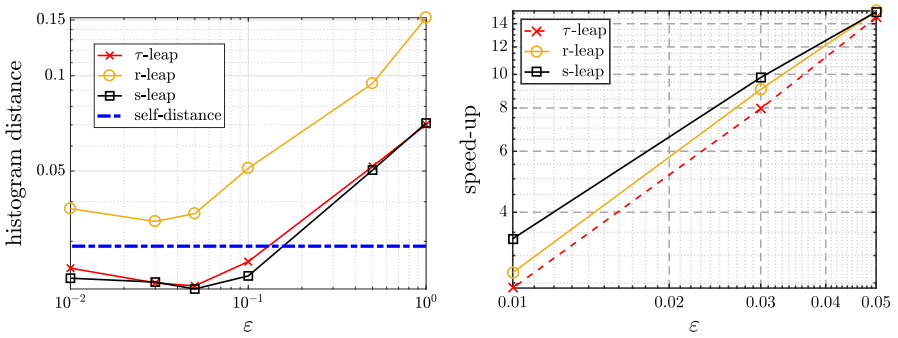

Fig. 6 (Color figure online) Errors and efficiency for the LacZ/LacY system discussed in Sect. 4.4, with large initial population

5 Conclusion

In this paper, we have introduced the S-leaping, an approximate algorithm for accelerating the SSA. The algorithm combines the advantages of two main approximate algorithms, the τ-leaping and R-leaping.

The S-leaping method uses a time step selection, intrinsic to the τ-leaping, which enables the extension of the algorithm to an implicit version. Furthermore, the S-leaping exploits the efficient sampling procedure from the R-leaping which reduce appearance of negative species. Moreover, the reordering of reaction channels inherited from the R-leaping, leads to a better performance of the S-leaping, compared to the τ-leaping, in big and stiff systems. On the other hand, if a stiff system involves reversible reactions appearing close to equilibrium, then the implicit approach derived from the τ-leaping accelerates the S-leaping by a few orders of magnitude in comparison with the explicit methods.

The performance of the proposed algorithm was tested on several examples, including a stiff, a non-stiff and a system involving slow and fast reactions with some species appearing in populations close to zero. In all test cases, accuracy of the S-leaping is similar to accuracy of the other accelerated methods. The performance of the S-leaping

is comparable with the fastest method or even outperform both, the τ-leaping and R-leaping methods. The S-leaping can be thus consider as optimal adaptive coupling of the R-leaping and τ-leaping method.

Future work directions involve the extension of the S-leaping algorithm to systems with spatial component by using compartment-based approach (Erban and Chapman 2009) or Brownian dynamics models (Lipkova et al. 2011) to extend the simulation framework for reaction-diffusion processes that arise in many biological systems.

Acknowledgements PK and GA gratefully acknowledge support from the European Research Council (ERC) Advanced Investigator Award (No. 341117). The authors thank both anonymous reviewers for their insightful comments that have helped us to improve the content of the paper.

References

Anderson DF, Kurtz TG (2011) Continuous time Markov chain models for chemical reaction networks. Springer, New York, pp 3–42

Auger A, Chatelain P, Koumoutsakos P (2006) *R*-leaping: accelerating the stochastic simulation algorithm by reaction leaps. J Chem Phys 125(8):084103

Bayati B, Chatelain P, Koumoutsakos P (2011) Adaptive mesh refinement for stochastic reaction–diffusion processes. J Comput Phys 230(1):13–26

Bayati B, Owhadi H, Koumoutsakos P (2010) A cutoff phenomenon in accelerated stochastic simulations of chemical kinetics via flow averaging (FLAVOR-SSA). J Chem Phys 133(24):1–7

Cao Y, Gillespie D, Petzold L (2005) Multiscale stochastic simulation algorithm with stochastic partial equilibrium assumption for chemically reacting systems. J Comput Phys 206(2):395–411

Cao Y, Gillespie D, Petzold L (2006) Efficient step size selection for the tau-leaping simulation method. J Chem Phys 124(4):044109

Cao Y, Gillespie DT, Petzold LR (2005) Avoiding negative populations in explicit poisson tau-leaping. J Chem Phys 123(5):054104

Cao Y, Petzold LR (2006) Accuracy limitations and the measurement of errors in the stochastic simulation of chemically reacting systems. J Comput Phys 212(1):6–24

Cao Y, Gillespie DT, Petzold LR (2007) Adaptive explicit–implicit tau-leaping method with automatic tau selection. J Chem Phys 126(22):224101

Chattopadhyay I, Kuchina A, Süel GM, Lipson H (2013) Inverse gillespie for inferring stochastic reaction mechanisms from intermittent samples. Proc Natl Acad Sci 110(32):12990–12995

Erban R, Chapman J, Maini P (2007) A practical guide to stochastic simulations of reaction-diffusion processes. arXiv:0704.1908

Erban R, Chapman SJ (2009) Stochastic modelling of reaction–diffusion processes: algorithms for bimolecular reactions. Phys Biology 6(4):046001

Gillespie DT (1976) A general method for numerically simulating the stochastic time evolution of coupled chemical reactions. J Comput Phys 22(4):403–434

Gillespie DT (1977) Exact stochastic simulation of coupled chemical reactions. J Phys Chem 81(25):2340–2361

Gillespie DT (2001) Approximate accelerated stochastic simulation of chemically reacting systems. J Chem Phys 115:1716

Gillespie DT, Petzold LR (2003) Improved leap-size selection for accelerated stochastic simulation. J Chem Phys 119:8229

Kierzek AM (2002) STOCKS: STOChastic Kinetic Simulations of biochemical systems with Gillespie algorithm. Bioinformatics (Oxford, England) 18(3):470–481

Koumoutsakos P, Feigelman J (2013) Multiscale stochastic simulations of chemical reactions with regulated scale separation. J Comput Phys 244:290–297

Lipkova J, Zygalakis KC, Chapman SJ, Erban R (2011) Analysis of Brownian dynamics simulations of reversible bimolecular reactions. SIAM J Appl Math 71(3):714–730

Maamar H, Raj A, Dubnau D (2007) Noise in gene expression determines cell fate in *Bacillus subtilis*. Science 317(5837):526–529

Mjolsness E, Orendorff D, Chatelain P, Koumoutsakos P (2009) An exact accelerated stochastic simulation algorithm. J Chem Phys 130:144110

Rathinam M, Petzold LR, Cao Y, Gillespie DT (2003) Stiffness in stochastic chemically reacting systems: the implicit tau-leaping method. J Chem Phys 119:12784

Sandmann W (2009) Exposition and streamlined formulation of adaptive explicitimplicit tau-leaping. Technical report, Citeseer

Süel GM, Garcia-Ojalvo J, Liberman LM, Elowitz MB (2006) An excitable gene regulatory circuit induces transient cellular differentiation. Nature 440(7083):545–550

Tian T, Burrage K (2004) Binomial leap methods for simulating stochastic chemical kinetics. J Chem Phys 121:10356