

## ADAPTIVE FINITE ELEMENT METHOD ASSISTED BY STOCHASTIC SIMULATION OF CHEMICAL SYSTEMS\*

SIMON L. COTTER<sup>†</sup>, TOMÁŠ VEJCHODSKÝ<sup>‡</sup>, AND RADEK ERBAN<sup>§</sup>

**Abstract.** Stochastic models of chemical systems are often analyzed by solving the corresponding Fokker–Planck equation, which is a drift-diffusion partial differential equation for the probability distribution function. Efficient numerical solution of the Fokker–Planck equation requires adaptive mesh refinements. In this paper, we present a mesh refinement approach which makes use of a stochastic simulation of the underlying chemical system. By observing the stochastic trajectory for a relatively short amount of time, the areas of the state space with nonnegligible probability density are identified. By refining the finite element mesh in these areas, and coarsening elsewhere, a suitable mesh is constructed and used for the computation of the stationary probability density. Numerical examples demonstrate that the presented method is competitive with existing a posteriori methods.

**Key words.** finite element methods, chemical Fokker–Planck, adaptive meshes, stochastic simulation algorithm

**AMS subject classifications.** 65N50, 65N30, 80A30, 82C31, 92C40

**DOI.** 10.1137/120877374

**1. Introduction.** Stochastic simulation algorithms (SSAs) have been successfully used in recent years to aid in the understanding of a number of biochemical models [3, 32, 43]. However, a systematic analysis of these models is challenging because of the computational intensity of SSAs. In some cases, analysis of stochastic chemical models is possible by considering suitable Fokker–Planck equations which are partial differential equations (PDEs) written for probability distribution functions. Examples include the chemical Fokker–Planck equation [18, 26] and the effective Fokker–Planck equations, which describe small (interesting) components of large chemical systems [13, 20]. In this paper, we will focus on solving the stationary chemical Fokker–Planck equation, but the presented computational approach is also applicable to solving effective Fokker–Planck equations.

Consider a well-mixed system of  $N$  chemical species and denote by  $\mathbf{x} = (x_1, \dots, x_N)$  a vector of concentrations of these species. The stationary chemical Fokker–Planck equation is a drift-diffusion PDE for an  $N$ -dimensional probability distribution func-

\*Submitted to the journal's Computational Methods in Science and Engineering section May 15, 2012; accepted for publication (in revised form) December 3, 2012; published electronically January 10, 2013. This work was supported by the European Research Council under the European Community's Seventh Framework Programme (FP7/2007-2013)/ERC grant agreement 239870 and was based on work supported in part by award KUK-C1-013-04, made by King Abdullah University of Science and Technology (KAUST).

<http://www.siam.org/journals/sisc/35-1/87737.html>

<sup>†</sup>School of Mathematics, University of Manchester, Oxford Road, Manchester, M13 9PL, United Kingdom (simon.cotter@manchester.ac.uk). This author's work was partially supported by a Junior Research Fellowship of St Cross College, University of Oxford.

<sup>‡</sup>Institute of Mathematics, Czech Academy of Sciences, Žitná 25, 115 67 Praha 1, Czech Republic (vejchod@math.cas.cz). This author's work was supported by the Grant Agency of the Academy of Sciences (project IAA100190803) and RVO 67985840.

<sup>§</sup>Mathematical Institute, University of Oxford, 24-29 St. Giles', Oxford, OX1 3LB, United Kingdom (erban@maths.ox.ac.uk). This author's work was supported by Somerville College, University of Oxford, by a Fulford Junior Research Fellowship; Brasenose College, University of Oxford, by a Nicholas Kurti Junior Fellowship; the Royal Society for a University Research Fellowship; and the Leverhulme Trust for a Philip Leverhulme Prize. This prize money was used to support research visits of Tomáš Vejchodský in Oxford.

tion  $p \equiv p(\mathbf{x})$ , where  $\mathbf{x} \in \Omega \subset \mathbb{R}^N$ , which can be written in the following form:

$$(1.1) \quad \operatorname{div} [D(\mathbf{x})\nabla p(\mathbf{x}) - \mathbf{v}(\mathbf{x})p(\mathbf{x})] = 0,$$

where  $D \equiv D(\mathbf{x}) : \Omega \rightarrow \mathbb{R}^{N \times N}$  is the diffusion matrix,  $\mathbf{v} \equiv \mathbf{v}(\mathbf{x}) : \Omega \rightarrow \mathbb{R}^N$  is the drift term, and  $\operatorname{div}$  is the divergence operator.

There have been several methods developed in the literature to solve (1.1) for moderately large  $N$ . They include adaptive finite element methods (FEMs), which are commonly used for  $N \leq 3$  [4], and sparse grid approaches, which are applicable for larger values of  $N$  [44]. In [40], a finite volume method was implemented in order to solve the chemical Fokker–Planck equation for dimension  $N \leq 4$ . A uniform mesh was used, and it was shown that for low dimensional problems, the Fokker–Planck approach can be more efficient than exhaustive stochastic simulations. On the other hand, SSAs were more efficient than solution of the Fokker–Planck equations for examples in three and four dimensions, because uniform meshes become inefficient and computationally intractable for large values of  $N$ . Adaptive FEMs [4] can be used to identify a suitable mesh which is refined in crucial regions and not in others. Although these approaches can be used for the chemical Fokker–Planck equation, they do not exploit the fact that the solution of (1.1) is a probability distribution of a stochastic process.

In this paper, we present an adaptive mesh construction which is suitable for the FEM solution of (1.1) if this equation arises from modeling of stochastic chemical systems. The main idea is to exploit the fact that stochastic trajectories spend a significant amount of time in parts of the state space where the mesh refinement is needed. Since adaptive FEMs are mostly applicable for systems up to  $N = 3$ , we will focus on systems of three chemical species. However, the presented methodology can be modified for larger (multiscale) chemical systems provided that they have up to  $n \leq 3$  important (slow) variables. In [13, 20], a method for estimation of coefficients of an effective Fokker–Planck equation is presented. This effective equation is of the same form as (1.1) but it is written in the dimension  $n$  which is smaller than a total number  $N$  of chemical species. If one has a suitable SSA for simulating the low dimensional slow dynamics [10, 11, 17], the presented mesh refinement can be applied. However, since the main aim of this paper is to present the numerical methodology for solving (1.1), we will not consider any dimensional reduction and restrict the study to systems which are directly written in terms of  $N = 3$  chemical species.

When using the Fokker–Planck approach to approximate the stationary probability distribution function, one must also identify a suitable domain  $\Omega$  on which to solve (1.1). Too small a domain, and the solution will be inaccurate, even missing areas with high probability with respect to the stationary probability distribution function. Too large a domain, and the method becomes inefficient. Automated methods have been developed to identify appropriate domains for finite state projections of the chemical master equation [37], but to the best of our knowledge, automated approaches were not used when solving the chemical Fokker–Planck equation. The method presented in [37] uses an iterative approach for finding  $\Omega$ , which increases the number of states included in the approximation, until error tolerances have been achieved.

The presented method is useful in situations where the coefficients of the Fokker–Planck equation are expensive to calculate, as in the scenario where they must be estimated using a suitable multiscale algorithm [13, 20]. In this instance, an a priori mesh generation method is preferable, since a posteriori methods would require more evaluations of the coefficients at different sets of quadrature points. In this setting,

the complexity is dominated by these estimations, and therefore any algorithm which minimizes the number of evaluations is advantageous.

The paper is organized as follows. In section 2, we introduce our notation, the Gillespie SSA, and the chemical Fokker–Planck equation. In section 3, we introduce the standard finite element framework that we will use throughout the paper. We also formulate the problem in its weak form in order to define the problem to be solved. In section 4, we introduce the stochastic simulation assisted adaptive finite element method (saFEM). In section 5, we offer some insight into how the algorithmic parameters of the saFEM may be chosen in practice. In section 6, implementational issues related to the algorithm are addressed. Numerical results concerning convergence of the method and comparison with an a posteriori method are presented in section 7.

**2. Chemical Fokker–Planck equation.** Let us consider a well-mixed system of  $N$  chemical species  $X_i$ ,  $i = 1, 2, \dots, N$ , which are subject to  $M$  chemical reactions  $R_k$ ,  $k = 1, 2, \dots, M$ . The state of the system is described by the state vector  $\mathbf{X}(t) = [X_1(t), X_2(t), \dots, X_N(t)]$ , where  $X_i(t)$  denotes the number of molecules of the corresponding chemical species. Each chemical reaction is described by its propensity function and the stoichiometric vector [19, 24]. The propensity function  $\alpha_k(\mathbf{x})$  is defined in such a way that  $\alpha_k(\mathbf{x})dt$  is the probability that the  $k$ th reaction occurs in the infinitesimally small interval  $[t, t+dt)$  provided that  $\mathbf{X}(t) = \mathbf{x}$ . The stoichiometric vector is  $\boldsymbol{\nu}_k = [\nu_{k1}, \nu_{k2}, \dots, \nu_{kN}]$ , where  $\nu_{ki}$  is the change in  $X_i$  during reaction  $R_k$ ,  $k = 1, 2, \dots, M$ . Stoichiometric vectors form the corresponding stoichiometric matrix  $\boldsymbol{\nu} = (\nu_{ki})_{k,i=1}^{M,N}$ .

The time evolution of the state vector  $\mathbf{X}(t)$  is often simulated by the Gillespie SSA [24], which is described in Table 2.1. Given the values of the propensity functions (step [1]), the waiting time to the next reaction is given by

$$(2.1) \quad \tau = -\frac{\log(u)}{\alpha_0(\mathbf{X}(t))}, \quad \text{where} \quad \alpha_0(\mathbf{X}(t)) = \sum_{k=1}^M \alpha_k(\mathbf{X}(t)),$$

and  $u$  is a uniformly distributed random number in  $(0, 1)$ . The reaction  $R_j$  is chosen in step [3] using another uniformly distributed random number.

The Gillespie SSA samples directly from the probability distribution  $p^d \equiv p^d(\mathbf{x}^d, t) : \mathbb{N}_0^N \times [0, \infty) \rightarrow [0, \infty)$ , where  $\mathbb{N}_0 = \{0, 1, 2, 3, 4, \dots\}$ . The evolution of  $p^d$  is governed by the chemical master equation

$$(2.2) \quad \frac{\partial p^d}{\partial t}(\mathbf{x}^d, t) = \sum_{k=1}^M \alpha_k(\mathbf{x}^d - \boldsymbol{\nu}_k) p^d(\mathbf{x}^d - \boldsymbol{\nu}_k, t) - \sum_{k=1}^M \alpha_k(\mathbf{x}^d) p^d(\mathbf{x}^d, t),$$

where  $p^d(\mathbf{x}^d, t)$  is the probability that  $\mathbf{X}(t) = \mathbf{x}^d$ . Here, the index  $d$  highlights that the chemical master equation (2.2) is a large system of ordinary differential equations

TABLE 2.1  
The pseudo code for the Gillespie SSA.

- [1] Calculate propensity functions  $\alpha_k(\mathbf{X}(t))$ ,  $k = 1, 2, \dots, M$ .
- [2] Waiting time  $\tau$  till next reaction is given by (2.1).
- [3] Choose one  $j \in \{1, 2, \dots, M\}$  with probability  $\alpha_j(\mathbf{X}(t))/\alpha_0(\mathbf{X}(t))$ , and perform reaction  $R_j$ , by adding  $\nu_{ji}$  to each  $X_i(t)$  for all  $i = 1, 2, \dots, N$ .
- [4] Continue with step [1] with time  $t = t + \tau$ .

(ODEs) for probabilities of discrete states  $\mathbf{x}^d \in \mathbb{N}_0^N$ . The stationary probability distribution corresponding to the chemical system can be approximated by solving the chemical Fokker–Planck equation [26], which is a continuous approximation of the chemical master equation (2.2) and in fact describes the evolution of probability densities of the chemical Langevin equation [26, 28]. The chemical Fokker–Planck equation can be written in the form (1.1), where the diffusion and drift coefficients are

$$(2.3) \quad d_{ij}(\mathbf{x}) = \frac{1}{2} \sum_{k=1}^M \nu_{ki} \nu_{kj} \alpha_k(\mathbf{x}), \quad i, j = 1, 2, \dots, N,$$

$$(2.4) \quad v_i(\mathbf{x}) = \sum_{k=1}^M \nu_{ki} \alpha_k(\mathbf{x}) - \sum_{j=1}^N \frac{\partial d_{ij}}{\partial x_j}(\mathbf{x}), \quad i = 1, 2, \dots, N.$$

This equation is solved on a bounded domain  $\Omega \subset [0, \infty)^N$  in which the vast majority of the invariant probability density sits [18]. On the boundary  $\partial\Omega$  we introduce the homogeneous Neumann boundary condition,

$$(2.5) \quad [D(\mathbf{x})\nabla p(\mathbf{x}) - \mathbf{v}(\mathbf{x})p(\mathbf{x})] \cdot \mathbf{n}(\mathbf{x}) = 0 \quad \text{for } \mathbf{x} \in \partial\Omega,$$

where  $\mathbf{n}(\mathbf{x})$  is the outward facing normal at  $\mathbf{x} \in \partial\Omega$ . We seek a solution of (1.1) which corresponds to the probability distribution function. Therefore, we impose the following normalization condition:

$$(2.6) \quad \int_{\Omega} p(\mathbf{x}) \, d\mathbf{x} = 1.$$

The choice of the computational domain  $\Omega$  might be problematic if we have no a priori information about the problem (1.1) with (2.5) and (2.6). In this case, the stochastic simulations provide a reliable tool to determine the correct  $\Omega$  as we will show in section 4.2.

Our aim in this work is to approximate the steady state solution of the chemical master equation (2.2) through solution of the chemical Fokker–Planck equation (1.1), which is a continuous approximation of the master equation. However, this approximation of the master equation can be poor if the system only rarely enters parts of the state space for which the conditions specified in [26] hold. This puts further restrictions on the region  $\Omega$ . However, before discussing the details of the saFEM and its applicability, we have to introduce some finite element terminology.

**3. FEM.** Equation (1.1) with boundary condition (2.5) can be numerically solved by the FEM. Since the finite element formulation is based on the corresponding weak formulation, we first introduce the weak solution  $p \in H^1(\Omega)$  by the equality

$$(3.1) \quad a(p, \phi) = 0 \quad \forall \phi \in H^1(\Omega),$$

where the bilinear form  $a(\cdot, \cdot)$  is given by

$$(3.2) \quad a(p, \phi) = \int_{\Omega} (D(\mathbf{x})\nabla p(\mathbf{x}) - \mathbf{v}(\mathbf{x})p(\mathbf{x})) \cdot \nabla \phi(\mathbf{x}) \, d\mathbf{x}.$$

Further,  $p$  must satisfy the condition (2.6).

The finite element formulation is obtained by projecting the weak formulation (3.1) into a finite dimensional subspace  $V_h$  of  $H^1(\Omega)$ . Thus, we seek  $p_h \in V_h$  such

that

$$(3.3) \quad a(p_h, \phi_h) = 0 \quad \forall \phi_h \in V_h$$

with the normalization condition

$$(3.4) \quad \int_{\Omega} p_h(\mathbf{x}) \, d\mathbf{x} = 1.$$

We note that taking a basis  $\phi_1, \phi_2, \dots, \phi_m$  of  $V_h$ , we can express the finite element solution as

$$p_h(\mathbf{x}) = \sum_{i=1}^m p^i \phi_i(\mathbf{x}), \quad \mathbf{x} \in \Omega.$$

Here, the coefficients  $p^i \in \mathbb{R}$  solve the system of linear algebraic equations

$$(3.5) \quad A\mathbf{p} = 0,$$

where  $\mathbf{p} = (p^1, p^2, \dots, p^m)^T$  and the stiffness matrix  $A \in \mathbb{R}^{m \times m}$  is defined by its entries

$$(3.6) \quad A_{ij} = a(\phi_j, \phi_i), \quad i, j = 1, 2, \dots, m.$$

We construct the finite element space  $V_h$  and the corresponding basis functions in the standard way [12]. In what follows, we will focus on computations in three dimensions, i.e.,  $N = 3$ . We consider the finite element mesh  $\mathcal{T}_h$  consisting of tetrahedral elements. The lowest order finite element space  $V_h$  then consists of globally continuous and piecewise linear functions over the mesh  $\mathcal{T}_h$ :

$$(3.7) \quad V_h = \{\phi_h \in H^1(\Omega) : \phi_h|_K \in \mathbb{P}^1(K) \, \forall K \in \mathcal{T}_h\},$$

where  $\mathbb{P}^1(K)$  stands for the space of linear functions over the tetrahedron  $K \in \mathcal{T}_h$ . If  $\mathbf{q}_j \in \mathbb{R}^3$ ,  $j = 1, 2, \dots, m$ , stands for the nodes of the mesh  $\mathcal{T}_h$ , then the standard finite element basis functions  $\phi_i \in V_h$  are uniquely determined by the condition

$$\phi_i(\mathbf{q}_j) = \delta_{ij}, \quad i, j = 1, 2, \dots, m,$$

where  $\delta_{ij}$  stands for Kronecker's symbol.

An efficient solution to problem (3.3) can be obtained by employing adaptively refined meshes [42]. The optimally adapted mesh leads to an approximation with the smallest error, provided the number of degrees of freedom is fixed. Practically, the optimal mesh can be hard to determine, but meshes close to the optimal can be found. These meshes are fine in regions where the solution exhibits steep gradients, boundary layers, interior layers, or singularities, and they are relatively coarse in the other regions.

The standard numerical approach for construction of nearly optimal meshes is the adaptive algorithm based on suitable a posteriori error estimators [4, 42]. This algorithm starts with an initial coarse mesh and refines it adaptively by a sequence of refinement steps. In each step, problem (3.3) has to be solved on the actual mesh, an error indicator has to be computed for each element, and based on these indicators the mesh is refined at suitable places (see section 7.2). Using the mesh refinements assisted by stochastic simulations, we can avoid the sequence of refinement steps and construct a suitably adapted mesh at once. We conclude this introductory section by discussing basic properties of solutions to (3.1) under the constraint (2.6).

**3.1. Existence and uniqueness of the solution.** The existence of a nontrivial solution to (3.1) follows by an application of the Fredholm alternative [21, p. 321]

under the assumption of uniform positive definiteness of matrix  $D(\mathbf{x})$ . This assumption holds if the propensity functions satisfy  $\alpha_k(\mathbf{x}) \geq \varepsilon > 0$  for all  $k = 1, 2, \dots, M$  and if the matrix  $\nu^T \nu$  is positive definite. In order to apply the Fredholm alternative we define the corresponding nonhomogeneous problem: find  $p \in H^1(\Omega)$  such that

$$(3.8) \quad a(p, \phi) = \int_{\Omega} f \phi \, d\mathbf{x} \quad \forall \phi \in H^1(\Omega),$$

where  $f \in L^2(\Omega)$ . Similarly, the corresponding adjoint problem reads: find  $z \in H^1(\Omega)$  such that

$$(3.9) \quad a(\psi, z) = 0 \quad \forall \psi \in H^1(\Omega).$$

We observe that any constant function  $z$  solves the adjoint problem. Further, it is easy to find a function  $f \in L^2(\Omega)$  such that  $\int_{\Omega} f z \, d\mathbf{x} \neq 0$  for a constant solution  $z$  of the adjoint problem. Thus, the Fredholm alternative implies that the nonhomogeneous problem (3.8) does not have a solution for this  $f$  and consequently there must be a nontrivial solution of the homogeneous problem (3.1). Note that the Fredholm alternative for linear elliptic operators was proved in [21, p. 321] for Dirichlet boundary conditions. Generalization to Neumann boundary conditions (2.5) is an easy exercise.

The uniqueness of the weak solution  $p$  is guaranteed by the normalization condition (2.6) and by the fact that the space of all solutions to the homogeneous problem (3.1) is one-dimensional. The one-dimensionality can be easily shown for domains satisfying the interior ball condition and under the assumption of sufficient regularity of the adjoint problem (3.9). In particular, let  $z \in C^2(\Omega) \cap C(\overline{\Omega})$  be a solution to the adjoint problem (3.9). Then this  $z$  satisfies

$$(3.10) \quad \operatorname{div}(D(\mathbf{x})\nabla z(\mathbf{x})) + \mathbf{v}(\mathbf{x}) \cdot \nabla z(\mathbf{x}) = 0 \quad \text{in } \Omega \quad \text{and} \quad (D(\mathbf{x})\nabla z(\mathbf{x})) \cdot \mathbf{n}(\mathbf{x}) = 0 \quad \text{on } \partial\Omega$$

in the classical pointwise sense. We consider  $\mathbf{x}_0 \in \overline{\Omega}$  such that  $z(\mathbf{x}_0) = \max\{z(\mathbf{x}) : \mathbf{x} \in \overline{\Omega}\}$  and we distinguish two cases. First, if  $\mathbf{x}_0$  lies in the interior of  $\Omega$ , then the strong maximum principle [21, p. 349] implies that  $z$  is constant. Second, if  $\mathbf{x}_0$  lies on the boundary  $\partial\Omega$ , then we first assume that  $z(\mathbf{x}_0) > z(\mathbf{x})$  for all  $\mathbf{x}$  in the interior of  $\Omega$ . A slight modification of Hopf's lemma [21, p. 347] then yields inequality  $(D(\mathbf{x}_0)\nabla z(\mathbf{x}_0)) \cdot \mathbf{n}(\mathbf{x}_0) > 0$ , which is a contradiction to the boundary condition for the adjoint problem (3.10). Thus, there has to be a point  $\mathbf{x}_1$  in the interior of  $\Omega$  such that  $z(\mathbf{x}_0) = z(\mathbf{x}_1)$ . Repeating the argument from the first case for  $\mathbf{x}_1$ , we conclude that  $z$  is constant even in the second case. Hence, we showed that the space of all solutions to the adjoint problem is the one-dimensional space of constant functions. The Fredholm alternative then implies that the space of solutions to the homogeneous problem (3.1) is one-dimensional as well. Consequently, there exists a unique solution of problem (3.1) with normalization condition (2.6).

Thus, solutions always exist once we restrict the problem to a bounded domain  $\Omega$  with Neumann boundary conditions, but these solutions are useful only if the underlying stochastic problem is ergodic, i.e., if it has a unique invariant distribution. If the system is not ergodic, then restricting the chemical Fokker–Planck equation to  $\Omega$  will result in a solution which has a nonnegligible probability density close to the boundary of  $\Omega$ . Necessary and sufficient conditions for the ergodicity of the chemical Langevin equation are not trivial [36], and thus ergodicity is assumed.

Finally, if the finite element space  $V_h$  contains constant functions (and our choice does), then the existence of nontrivial solutions to the finite element problem (3.3) follows by the same argument as for the original problem (3.1).

TABLE 4.1

*Outline of the a priori mesh generation methodology using SSA trajectories within the saFEM.*

- |   |
|---|
| <p>[1] Identify steady states (stable and unstable) of the mean field approximation of the system or, in the case of oscillatory systems, one (or more) coordinates along the limit cycle. We denote these <math>S \in \mathbb{N}</math> points by <math>\{\mathbf{y}_k\}_{k=1}^S</math>.</p> <p>[2] Run one (or more) Gillespie simulations of length <math>B + T &gt; 0</math> for each of the initial conditions <math>\{\mathbf{y}_k\}_{k=1}^S</math>. Here, <math>B \geq 0</math> is the length of a possible initial transient and <math>T &gt; 0</math>. Take a subsample of <math>Q &gt; 0</math> points per unit time per trajectory in the time interval <math>[B, B + T]</math> and denote them as in (4.2). We also denote by <math>x_i^{\max}</math> (resp., <math>x_i^{\min}</math>) the maximal (resp., minimal) value of the <math>i</math>th chemical species, <math>i = 1, 2, 3</math>, during time intervals <math>[B, B + T]</math> of all <math>S</math> simulations.</p> <p>[3] Use (4.3) to define a neighborhood <math>\Gamma \subset [0, \infty)^N</math> of the points (4.2) as the region of the domain in which we require the finest level of refinement of the mesh. This is made up of ellipsoids around each point with radii given by (4.5) with parameter <math>\beta_1 &gt; 0</math>.</p> <p>[4] Define the domain of solution <math>\Omega</math> by (4.6) using parameter <math>\beta_2 &gt; 0</math>.</p> <p>[5] Start with the mesh as one single cuboid covering the whole of <math>\Omega</math>.</p> <p>[6] Loop over all cuboids in the mesh. If the cuboid is sufficiently close (as given by (4.7)) to <math>\Gamma</math>, then split the cuboid into eight equally sized cuboids. Update the list of hanging nodes/hang type (face/edge hanging node as shown in Figure 4.1 (left)).</p> <p>[7] Repeat step [6] until the maximum resolution has been reached after <math>H \in \mathbb{N}</math> iterations, or the maximum total number of cuboids has been reached.</p> |
|---|

**4. Adaptive mesh refinement assisted by stochastic simulations.** Since the chemical Fokker–Planck equation (1.1) is a continuous approximation of the stationary probability density given by the Gillespie SSA [24], one can expect that trajectories simulated from the SSA will be informative. Once the trajectory has reached probabilistic equilibrium,<sup>1</sup> regions surrounding the path of the trajectory are likely to be regions with nonnegligible invariant density with respect to the steady state Fokker–Planck equation. We should be aiming to refine the finite element mesh in regions where the rate of change of the gradient of the invariant density is larger. These regions can be well approximated by the region which has nonnegligible invariant density. Therefore, stochastic simulations of the chemical system can be informative about a good choice of finite element mesh.

The construction of the locally adapted mesh assisted by stochastic simulations is done in three stages. In Stage I, we use the stochastic simulations to identify those regions of the state space where the system spends most of its time and hence where we are interested in resolving the problem with the highest accuracy. In Stage II, we identify the computational domain  $\Omega$  as a cuboid that covers the region from Stage I and its neighborhood. In Stage III, we construct the actual mesh in the computational domain  $\Omega$  based on the information from Stage I. In what follows, we provide detailed descriptions of these three stages in the case of  $N = 3$  chemical species. Following the notation of section 2, the chemical species will be denoted as  $X_1$ ,  $X_2$ , and  $X_3$ . The mesh generation algorithm is summarized in Table 4.1.

**4.1. Stage I: Stochastic simulation.** In step [1] of Table 4.1, we identify structures that a priori should exist in the probability density. An indication of the regions which may contain significant amounts of invariant density can be found by analyzing the three-dimensional system of ODEs

$$(4.1) \quad \frac{dx_i}{dt} = v_i(x_1, x_2, x_3), \quad i \in \{1, 2, 3\},$$

<sup>1</sup>Equilibrium can be reached simply by running the SSA until the state of the trajectory is sufficiently decorrelated from its starting position, or by starting at a steady state of the mean field approximation (4.1) of the chemical system.

which is closely related to the mean field approximation of the chemical system. Steady states of ODE system (4.1) will often coincide with regions of the solution of the steady state Fokker–Planck equation (1.1) that have large density. The most important things to identify are stable steady states of (4.1), which can be found by solving an algebraic system  $[v_1(x_1, x_2, x_3), v_2(x_1, x_2, x_3), v_3(x_1, x_2, x_3)] = 0$ . In oscillating systems, one can identify limit cycles. There are several tools in the literature for analysis of ODEs of the form (4.1), such as AUTO [15].

In step [2], these steady states can then be used as starting points for SSA trajectories, once the numbers of molecules of each species have been rounded to the nearest integer. Since we cannot guarantee that steady states of the mean field approximation are areas of interest with respect to the chemical Fokker–Planck equation, we might additionally wish to ensure that each chain is starting in probabilistic equilibrium by running a short simulation of length  $B \geq 0$  from these points before starting the sampling procedure. The Gillespie SSA is detailed in Table 2.1. In the case of limit cycles, several points along the limit cycle can be used as starting points for the SSA. Either way, we denote the  $S$  starting positions for the trajectories by  $\{\mathbf{y}_k\}_{k=1}^S \subset \mathbb{R}^3$ . The trajectories simulated up to some time  $B + T > 0$  using the SSA can help inform us about the regions of domain which will have nonnegligible invariant density. Since the trajectories will contain many points, we cannot store all the points that are simulated. Instead, we subsample from the trajectories at equidistant time points, at a rate  $Q > 0$  points per unit time. This leaves us with a set of sampled points

$$(4.2) \quad \{\mathbf{z}_{k,l}\}_{k=1,l=1}^{S,[QT]} \subset \mathbb{R}^3$$

from the invariant distribution, where  $[QT]$  denotes the integer part of the real number  $QT$ . We hope to recover, from this set of points (4.2), information about what might be an optimal finite element mesh. We also denote by  $x_i^{\max}$  (resp.,  $x_i^{\min}$ ) the maximal (resp., minimal) value of the  $i$ th chemical species,  $i = 1, 2, 3$ , during time intervals  $[B, B + T]$  of all  $S$  simulations. These numbers will be useful in (4.4)–(4.5). In step [3], we make the approximation that the region which contains the majority of the invariant density is a subset of the union of a set of ellipsoids centered at each point (4.2), namely,

$$(4.3) \quad \Gamma \equiv \bigcup_{k,l} \mathcal{E}_{\mathbf{r}}(\mathbf{z}_{k,l}).$$

Here,  $\mathcal{E}_{\mathbf{r}}(\mathbf{z}_{k,l})$  is an ellipsoid with radii  $\mathbf{r} = [r_1, r_2, r_3]$  centered at point  $\mathbf{z}_{k,l}$  for  $k = 1, 2, \dots, S$ ,  $l = 1, 2, \dots, [QT]$ . These radii can be picked to be proportional to the range of each chemical species using the parameter  $\beta_1 > 0$  and numbers  $x_i^{\min}$ ,  $x_i^{\max}$ ,  $i = 1, 2, 3$ , computed in step [2]. Namely, we define

$$(4.4) \quad x_i^{\text{range}} = x_i^{\max} - x_i^{\min}, \quad i = 1, 2, 3,$$

and

$$(4.5) \quad \mathbf{r} = \beta_1 (x_1^{\text{range}}, x_2^{\text{range}}, x_3^{\text{range}}).$$

One should note that the parameters  $B$  and  $T$  might in principle be different for different initial conditions  $y_k$ , but to simplify the notation, we do not stress this fact by using the notation  $B_k$  and  $T_k$  and use simply  $B$  and  $T$ .

For some systems, the calculation of the position of all the steady states in system (4.1) could be nontrivial. In this case, this a priori information may not be available

to us, and as such longer SSA trajectories should be calculated in order to find all the regions close to steady states. In some cases, this may be costly and the advantages of using this methodology are lost.

**4.2. Stage II: Automatic detection of the domain.** Next, we would like to identify the domain  $\Omega$  on which we wish to solve (1.1). Since we already have an approximation of the region which contains the majority of the probability density, we can simply fit a cuboid around this region and then extend it by a factor. The only other condition that we enforce is that the domain must be contained by the positive quadrant of the state space. In step [4], we pick a parameter  $\beta_2 > 0$  to define how much we would like to extend the cuboid:

$$(4.6) \quad \Omega = \mathcal{A}_1 \times \mathcal{A}_2 \times \mathcal{A}_3,$$

where

$$\mathcal{A}_i = (\max\{0, x_i^{\min} - \beta_2 x_i^{\text{range}}\}, x_i^{\max} + \beta_2 x_i^{\text{range}}), \quad i = 1, 2, 3.$$

We then wish to solve (1.1) on the domain  $\Omega$  with boundary conditions (2.5) on  $\partial\Omega$ .

This method follows from [40], where the domain is chosen such that the solution can be well approximated by zero outside it. The longer we run the SSA trajectories, the better our approximation of the regions with nonnegligible invariant density.

**4.3. Stage III: Construction of the adaptive mesh.** In this stage, we construct the adaptively refined mesh. In the previous stage we naturally defined the computational domain as a cuboid (4.6). Therefore, we base our refinement approach on simple splitting of a cuboid into eight congruent subcuboids. However, any other standard mesh refinement technique can be utilized instead.

An advantage of the refinement of cuboids into eight subcuboids is its simplicity. A disadvantage is that these refinements produce so-called hanging nodes in the mesh; see Figure 4.1(a).

In step [5] of the mesh generation, our mesh consists of a single large cuboid. We then, in step [6], iteratively refine cuboids in the following fashion. In each iteration of the refinement procedure, we loop over all the current cuboids that exist following the previous iteration of the method. For each of the cuboids  $I_1 \times I_2 \times I_3$ , where

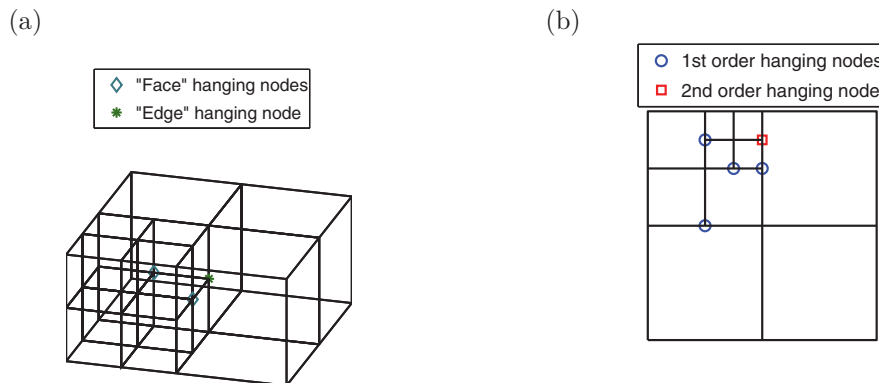


FIG. 4.1. (a) First order hanging nodes of different types in cuboid meshes. (b) Hanging nodes of order 1 and order 2 on a two-dimensional mesh.

$I_1, I_2, I_3 \subset [0, \infty)$  are intervals, we compute the minimum distance between all points in this cuboid and the set of points (4.3) in each coordinate, namely,

$$\text{dist}(\Gamma, I_i) = \inf\{|a_i - b_i| : [a_1, a_2, a_3] \in \Gamma, b_i \in I_i\}, \quad i = 1, 2, 3.$$

We refine the given cuboid  $I_1 \times I_2 \times I_3$  if

$$(4.7) \quad \text{dist}(\Gamma, I_i) < |I_i| \quad \text{is satisfied for every } i = 1, 2, 3.$$

If a cuboid is to be refined, it is split into eight cuboids of equal volume. As detailed in step [7], this refinement condition can be iterated for a set number of times  $H \in \mathbb{N}$ , where  $\mathbb{N} \equiv \{1, 2, 3, \dots\}$ , or until the number of cuboids present in the mesh is as large as we would like or can deal with given our computational resources.

Notice that the algorithm described above produces hanging nodes of order 1 only. This means that either two neighboring elements in the mesh are of equal size or one of them has eight times' greater volume than the other one. This is important for practical implementation, because the hanging nodes of order 1 are much easier to work with than the hanging nodes of higher orders. See [41] for more details and Figure 4.1(b) for an illustration in two dimensions. In order to guarantee the continuity of the approximation, the value of the function at the hanging node is necessarily decided by the values of the function at the vertices of the less refined cube. Therefore the hanging nodes are not in fact additional degrees of freedom in the problem. Thus, the dimension of problem (3.5) is equal to the total number of vertices in the generated mesh, less the number of hanging nodes. This special treatment of hanging nodes actually enforces on us a mesh which is highly refined in the regions which we wish it to be, and then the mesh becomes gradually coarser as we move away from those regions.

**4.4. Tetrahedral mesh.** Once the cuboid mesh has been generated by steps [1]–[7] of Table 4.1, we can then implement an FEM on it. In the numerics shown in this paper, we further split each cuboid into six path tetrahedra. However, there is nothing to say that we could not use cubic elements, but we use tetrahedra to simplify some implementational issues.

We choose a refinement regime in which the elements are clustered in groups of six nonobtuse<sup>2</sup> tetrahedra which together form each cuboid [8]. Figure 4.2(a) shows how six tetrahedra of equal size and shape can tessellate into a cube; this gives us the basis of our refinement scheme. The idea is that if we map each cuboid in our mesh to the unit cube, then each element is of exactly the same shape and size.

A lot of refinement methods in the literature involve splitting each element into several smaller elements [33]. However, if this is not done in a clever way, it can lead to degradation of the quality of the elements. That is, some of the angles of the elements may become too small, leading to long thin elements, which can lead to less accuracy in the approximation [9].

In the method that we propose here, instead of splitting each element, we split each cuboid into eight equally sized cuboids. Each cuboid is then split, as before, into six equally sized and shaped (after a linear transformation if not a cube) tetrahedral elements. Figure 4.2(b) shows a cube which has been refined once and been split into 48 elements with 27 vertices.

<sup>2</sup>All six dihedral angles between its faces are less than or equal to  $\pi/2$ .

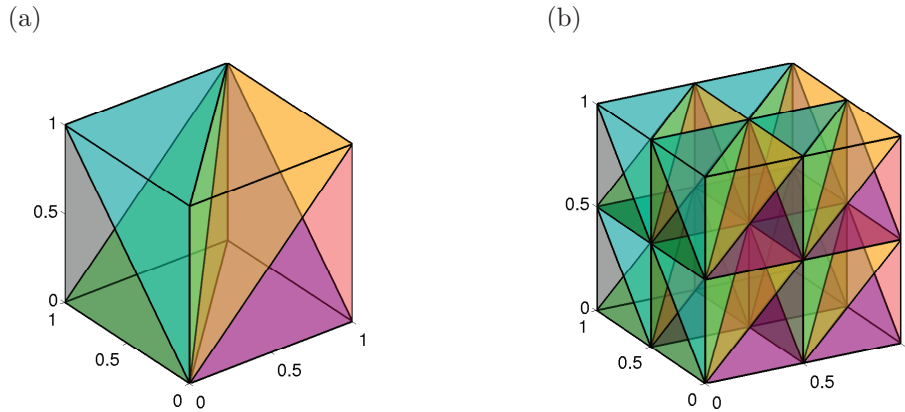


FIG. 4.2. (a) Example of a unit cube split into six elements of equal size and shape. (b) Example of a unit cube after a refinement iteration.

**5. Parameter selection.** The algorithm has several parameters,  $S$ ,  $T$ ,  $Q$ ,  $B$ ,  $\beta_1$ ,  $\beta_2$ , and  $H$ , whose values must be decided by the user. The parameters  $S$ ,  $T$ ,  $Q$ ,  $B$  relate to the implementation of the Gillespie SSA, and the remaining three,  $\beta_1$ ,  $\beta_2$ ,  $H$ , relate to the selection of the domain and the mesh, given the recorded output of the SSA. In this section we suggest how one might go about selecting values for these parameters.

**5.1. SSA parameters.** The purpose of simulating the SSA trajectories, as described in subsection 4.1, is to get an indication of the regions of the domain which contain the majority of the invariant probability density. There are four parameters to consider,  $S \in \mathbb{N}$ ,  $T > 0$ ,  $Q > 0$ , and  $B \geq 0$ . The parameter  $B$  represents the length of the simulated trajectory that we ignore at the beginning of the simulation. This period of simulation is simply used to ensure that the Markov chain has entered probabilistic equilibrium. This ensures that it is highly likely the point in state space where the rest of the trajectory starts has nonnegligible density with respect to the invariant distribution. The final parameter,  $Q$ , represents the rate (per unit time) at which we record samples from each trajectory in the time interval  $[B, B+T]$ . This parameter is necessary since if we recorded every single state that the trajectory passed through, we could quickly run out of memory.

Choosing sensible values for these parameters is not easy, since it is problem dependent. However, our aim is not to run such a long trajectory that simply using a histogram of the values would lead to an accurate approximation of the invariant density, as this would completely negate the need for the rest of the algorithm. Likewise, if the trajectory is too short, we will have a very poor approximation of the areas of the domain which contain the majority of the probability density. Our aim, therefore, is to get as good an approximation of this region as possible with the shortest possible trajectories.

As with many Monte Carlo estimates, due to the law of large numbers, the error decays at a rate proportional to  $1/\sqrt{T}$ . It is reasonable to expect that the approximation of the region which contains the majority of the invariant density should decay at a similar rate. Since convergence diagnostics relating to Markov chains are an open area with no hard and fast solution [14], we suggest that a sample trajectory be created a priori to ascertain the relaxation time of the system. The parameter  $B$

can simply be set to be a proportion of the length  $T$ , for instance,  $B = T/10$ . The parameter  $Q$  should be picked according to how much memory one would like to set aside to store the sampled points. Since we are calculating minimum distances to any point in this set in step [6] of the algorithm, the more points there are, the longer the algorithm will take to run.

**5.2. Domain and mesh generation parameters.** The domain and mesh generation parameters are given by  $\beta_1 > 0$ ,  $\beta_2 > 0$ , and  $H \in \mathbb{N}$ . The parameter  $\beta_2$  defines the size of the domain as seen in (4.6). Unless the trajectories from the SSA are very short, the estimation of the domain should be relatively trivial, and a value of order 1 should suffice in most instances. The value of  $\beta_1$ , which indicates how much error we estimate there to be in our estimation of the region which contains the majority of the invariant density, should be directly proportional to  $1/\sqrt{T}$ . Too large a value will lead to unnecessary refinement in some areas, while too small a value will lead to not enough refinement and more error. The parameter  $H$  indicates the maximum number of splits that the initial cubes of the mesh may undergo. In practice this can be found at runtime by capping the total number of elements we allow in the mesh due to memory and CPU constraints.

**5.3. A numerical test of accuracy.** Although the method has seven parameters which have to be specified, it is relatively easy to check that the computed results are numerically accurate. Given the parameter set  $\{S, T, Q, B, \beta_1, \beta_2, H\}$ , we can compare the computed results with the results obtained with the parameter set  $\{S, 2T, 2Q, 2B, 2\beta_1, \beta_2, H+1\}$ . If the result does not change significantly, then we can conclude that our parameters were chosen sufficiently large. Here, the parameters  $T$ ,  $Q$ ,  $B$ , and  $\beta_1$  are multiplied by 2 because increasing any of these parameters will increase the accuracy of the solution. In a similar way  $H$  can be increased by one, which means that the finest level of refinement in the mesh becomes smaller by a factor of two in each coordinate. However, we do not propose to modify  $S$  and  $\beta_2$  for the following reasons. The number of starting points  $S$  can be determined by the number of stable equilibria of (4.1). In this case, the parameter  $S$  does not have to be changed during this a posteriori test of accuracy. Furthermore,  $\beta_2$  is used only to choose the size of the domain, and multiplying the size of the domain by 2 would lead to a smaller proportion of the domain being covered by  $\Gamma$  (given by (4.3)), and as such a more accurate solution would not be guaranteed. If one wishes to test whether  $\beta_2$  is large enough, the parameter  $H$  may have to be increased as a function of  $\beta_2$  in order to maintain the same level of refinement in  $\Gamma$ .

**6. Implementation of the saFEM.** As soon as the computational mesh is determined, the approximation  $p_h \in V_h$  of the stationary distribution  $p(\mathbf{x})$  is computed by the standard FEM approach [12, 41]. In order to compute the entries of the stiffness matrix (3.6), it is necessary to use suitable quadrature routines. In numerical examples below, we consider chemical reactions of at most second order. Therefore, the diffusion  $D(\mathbf{x})$  and drift  $\mathbf{v}(\mathbf{x})$  coefficients are polynomials of degree at most two. Since the finite element space consists of piecewise linear functions, it suffices to use a tetrahedral quadrature rule of order three which integrates cubic polynomials exactly. The optimal (Gauss) quadrature rule on tetrahedra of order three has five points [41].

Since the dimension of the resulting algebraic system (3.5) can be very large, even for relatively coarse approximations, parallelization of the assembly process is of paramount importance. Several packages exist for constructing matrices in parallel. The Portable Extensible Toolkit for Scientific Computation (PETSc) is a suite of data

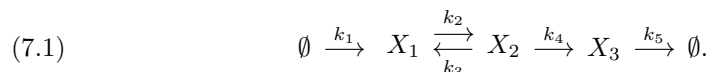
structures and routines for the scalable (parallel) solution of scientific applications modeled by PDEs [6, 7]. The matrix is split into sections which are controlled by each processor. If the degrees of freedom are ordered in a sensible way, then Message Passing Interface traffic between the processors can be kept to a minimum, leading to excellent scaling of processors versus computation time [7].

Once the stiffness matrix (3.6) is assembled, we have to find a nontrivial solution of the algebraic system (3.5). Practically, we look for the eigenvector corresponding to the zero eigenvalue of the matrix  $A$ . This eigenvector, once normalized, corresponds to the approximation of solution of (1.1) projected onto the vector space  $V_h$ . To find this eigenvector in parallel, we use a sister package of PETSc, which is called the Scalable Library for Eigenvalue Problem Computations (SLEPc) [30]. In particular, we used the MULTifrontal Massively Parallel sparse direct Solver [1, 2] package for the preconditioning and a power method to solve the resulting eigenvalue problem [34]. This is made easier since we know the exact value for which we wish to compute the eigenvector (i.e., zero). Feeding this information to the power method implementation within SLEPc allows us to quickly and accurately approximate the nontrivial solution of (3.5).

Once we have calculated the eigenvector, it is then necessary to reconstruct the solution of (3.1). For this, we use the information regarding the hanging nodes so that we reconstruct all the vertices on the mesh. The function in question approximates a probability density and therefore we normalize the obtained finite element solution  $p_h$  such that it satisfies the condition (3.4).

**7. Numerical results.** In this section, we first use a simple example chemical system from [13] and implement the saFEM on a range of different mesh sizes and with different algorithmic parameters. Then we present the results of saFEM for the Oregonator [23].

**7.1. Convergence of the numerical method.** We will study the system of three chemical species  $X_1$ ,  $X_2$ , and  $X_3$  which are subject to the following system of five chemical reactions [13]:



Then the propensity functions are defined by

$$\begin{aligned} \alpha_1(\mathbf{x}) &= k_1 V, & \alpha_2(\mathbf{x}) &= k_2 x_1, & \alpha_3(\mathbf{x}) &= k_3 x_2, \\ \alpha_4(\mathbf{x}) &= k_4 x_2, & \alpha_5(\mathbf{x}) &= k_5 x_3, \end{aligned}$$

where  $V$  is the volume of the reactor [13, 19]. In particular,  $N = 3$  and  $M = 5$  and the stoichiometric matrix of the chemical system (7.1) is a  $5 \times 3$  matrix

$$\nu = \begin{pmatrix} 1 & 0 & 0 \\ -1 & 1 & 0 \\ 1 & -1 & 0 \\ 0 & -1 & 1 \\ 0 & 0 & -1 \end{pmatrix}.$$

We consider this system with the following set of nondimensionalized parameters:

$$(7.2) \quad k_1 V = 100, \quad k_2 = k_3 = 5, \quad k_4 = k_5 = 1.$$

As discussed in section 5, there are seven different algorithmic parameters, whose values determine the accuracy and efficiency of the methods. In this section we will analyze the effects and convergence of the method due to altering the three most important of these parameters.

**7.2. Convergence of approximation.** First we consider how the error in the approximation of the invariant distribution  $p$  decays as we refine the mesh, each time using the same stochastic simulation, with  $S = 1$ ,  $T = 10^5$ ,  $Q = 0.1$ ,  $B = 10^3$ ,  $\beta_1 = 0.01$ , and  $\beta_2 = 0.55$ . We choose  $S = 1$  since the corresponding ODE approximation given by (4.1) with parameters (7.2) gives us

$$\begin{aligned}\frac{dx_1}{dt} &= 100 - 5x_1 + 5x_2, \\ \frac{dx_2}{dt} &= 5x_1 - 6x_2 - 0.5, \\ \frac{dx_3}{dt} &= x_2 - x_3,\end{aligned}$$

which has a single steady state at  $(119.5, 99.5, 99.5)$ . Therefore we pick the single starting point for the SSA trajectory to be at  $(120, 100, 100)$ . Note that the constant term  $-0.5$  in the second equation comes from the derivative of the diffusion terms as given by (2.4), while these derivatives sum to zero for the first and third equations.

We compare the convergence of the saFEM with the standard adaptive algorithm, which uses a posteriori error indicators to mark elements to be refined. In the numerics that follow, we choose the usual explicit residual error indicator [4, 5, 42],  $\eta_E$ , given by

$$(7.3) \quad \eta_E = h_E \|r_E\|_E + \sum_{F \in \mathcal{F}(E)} h_F^{1/2} \|J_F\|_F,$$

where  $E$  is an element (tetrahedron) of the mesh,  $\mathcal{F}(E)$  stands for the set of faces of  $E$ ,  $h_E$  and  $h_F$  are diameters of the element  $E$  and face  $F$ , respectively, and

$$(7.4) \quad r_E = -\operatorname{div}(D\nabla p_h - \mathbf{v}p_h),$$

$$(7.5) \quad J_F = (D\nabla p_h^+ - D\nabla p_h^-) \cdot n_F.$$

Here  $p_h$  is given by (3.3)–(3.4),  $n_F$  is the outward normal to the face  $F$ , and  $p_h^\pm(\mathbf{x}) = \lim_{s \rightarrow 0^+} p_h(\mathbf{x} \pm sn_F)$ . Note that  $J_F$  is the jump in conormal derivative. The norm  $\|\cdot\|_E$  (resp.,  $\|\cdot\|_F$ ) is the standard  $L^2$  norm on the domain  $E$  (resp.,  $F$ ).

The global error indicator  $\eta_G$  is then given by

$$(7.6) \quad \eta_G^2 = \sum_E \eta_E^2,$$

where the sum is taken over all elements in the mesh. To mark the elements, we use the usual bulk criterion, namely, we reorder the indicators according to size, denote the sorted sequence in descending order by  $\{\eta_{E_i}\}$ , and find the smallest number  $\tilde{n}$  such that

$$(7.7) \quad \sum_{i=1}^{\tilde{n}} \eta_{E_i}^2 \geq \theta \eta_G.$$

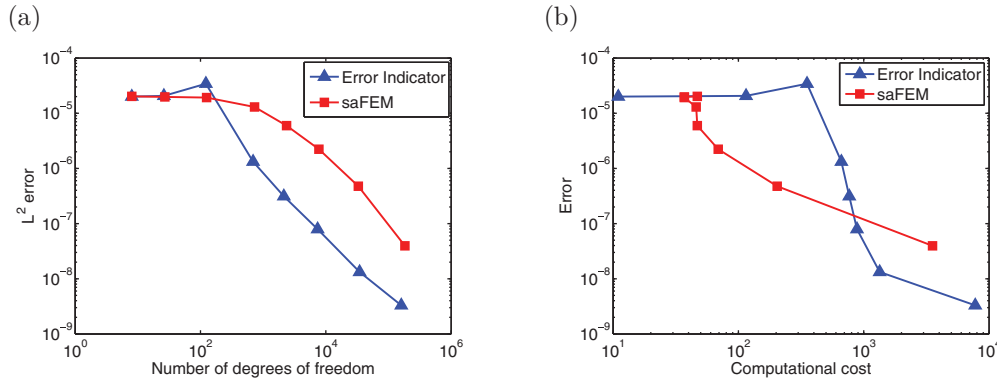


FIG. 7.1. (a) Convergence of the saFEM (red squares) compared with the error indicator approach (blue triangles), with  $S = 1$ ,  $T = 10^5$ ,  $Q = 0.1$ ,  $B = 10^3$ ,  $\beta_1 = 0.01$ , and  $\beta_2 = 0.55$  over a range of mesh refinements, for the system (7.1). Errors are given by (7.8). (b) Comparison of cost/error for the saFEM (red squares) and the standard error indicator approach (blue triangles), with  $S = 1$ ,  $T = 10^5$ ,  $Q = 0.1$ ,  $B = 10^3$ ,  $\beta_1 = 0.01$ , and  $\beta_2 = 0.55$  over a range of mesh refinements, for the system (7.1). Errors are given by (7.8).

The elements  $E_1, E_2, \dots, E_{\tilde{n}}$  are then marked for refinement. The parameter  $\theta \in (0, 1)$  can be freely chosen. We use  $\theta = 0.5$ .

In the following numerical experiments, we start the standard adaptive algorithm with the same cuboid domain as identified using the saFEM with initial mesh given by the same six path tetrahedra as used in the saFEM. Elements marked for refinement are split according to the longest edge bisection algorithm [39].

Figure 7.1(a) shows the convergence of the saFEM and the standard error indicator approach as the number of degrees of freedom is increased. The error is calculated for each method by comparison with an approximation using a very fine mesh for that method:

$$(7.8) \quad \text{Error} = \left( \int_{\Omega} |p_{\text{fine}}(\mathbf{x}) - p_h(\mathbf{x})|^2 d\mathbf{x} \right)^{1/2},$$

where  $p_{\text{fine}}$  is the saFEM solution with the finest mesh (with  $H = 8$ ), which has smallest elements with diameter 1.40 and largest elements with diameter 44.95, and the mesh has  $1.06 \times 10^6$  degrees of freedom. Similarly, the finest mesh in the error indicator approach has just over  $1.24 \times 10^6$  degrees of freedom. Note that the error is small (almost of order  $10^{-5}$ ) even for a very small number of degrees of freedom. This is simply due to the fact that the probability density has  $L^1$  norm equal to 1, and as the domain is large, it has very small  $L^2$  norm (equal to  $1.96 \times 10^{-5}$ ).

The error as a function of computational cost (in CPU time) is plotted in Figure 7.1(b). Since some parts of the computation are parallelized, we defined the computational cost for each part of the program as the runtime multiplied by the number of processors used. The plotted computational cost (in seconds) is then given by the sum of these costs.

When using a relatively coarse mesh, it is possible for the value of the approximation to be negative in regions. Since the function represents a probability density, which is strictly nonnegative, this does not make sense. These negative areas can also complicate the normalization of the function. We solve this problem by simply setting all negative values to zero, before normalizing by (2.6). This problem is standard in

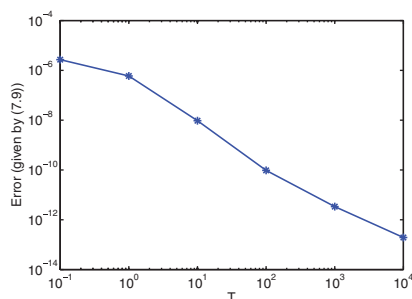


FIG. 7.2. Convergence of the saFEM with  $S = 1$ ,  $Q = 10$ ,  $B = 10^3$ ,  $\beta_1 = 0.01$ ,  $\beta_2 = 0.4$ , and  $H = 6$  over a range of SSA lengths  $T$  for the system (7.1). Error is given by (7.9).

finite element computations and it comes from the fact that the probability density function is very close to zero for large parts of the domain. Small numerical errors can then lead to negative values in these regions. These negatives are very close to zero even for moderately converged approximations. Moreover, since the exact solution is nonnegative, rounding up the negative values to zero yields a better approximation in the  $L^2$  sense.

In Figure 7.1 we see that the standard method creates a more efficient mesh in terms of error per degree of freedom in the mesh. However, the saFEM is still comparable if the drift and diffusion coefficients are expensive to compute, as mentioned in the introduction with regards to multiscale methods [13, 20]. It also allows us to construct nonconforming meshes while ensuring that hanging nodes of order higher than one are not present, allowing for greater flexibility in the choice of mesh generation.

Furthermore, in the numerics above, the standard method is taking some input from the saFEM, in terms of the computational domain chosen to solve the problem on. The automatic detection of this domain is a key feature of the saFEM which allows for efficient approximation of the stationary density of the system. If too large a domain is chosen on which to solve the problem, this introduces redundant degrees of freedom in regions which have negligible stationary probability density  $p(\mathbf{x})$ . Selection of too small a domain leads to an incorrect approximation of the invariant density.

**7.3. Length of stochastic simulation.** Next we show convergence of the method as  $T$  is increased with  $S = 1$ ,  $Q = 10$ ,  $B = 10^3$ ,  $\beta_1 = 0.01$ ,  $\beta_2 = 0.4$  and  $H = 6$ . As in the previous section, the SSA trajectory is given initial condition  $(X_1, X_2, X_3) = (120, 100, 100)$ .

As the system is ergodic, we would expect that as the length of stochastic simulation increases and the number of samples increases, our sample becomes increasingly representative of the stationary probability distribution  $p(\mathbf{x})$ . This gives us more information about where we should be refining our mesh, which in turn should give us a more accurate approximation.

Figure 7.2 shows the convergence of approximation as the length of stochastic simulation is increased. The plotted error is the  $L^2$  difference between the approximations with varying  $T$  and the approximation calculated with  $T = 10^5$ , namely,

$$(7.9) \quad \text{Error}(t) = \left( \int_{\Omega} |p_{T=10^5}(\mathbf{x}) - p_{T=t}(\mathbf{x})|^2 d\mathbf{x} \right)^{1/2}.$$

Note that in order to more easily compare the distributions, the domain selection for

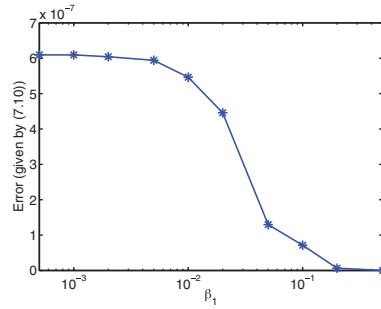


FIG. 7.3. Convergence of the saFEM with  $S = 1$ ,  $T = 10^5$ ,  $Q = 0.1$ ,  $B = 10^3$ ,  $\beta_2 = 0.4$ , and  $H = 6$  over a range of radii of uncertainty  $\beta_1$  for the system (7.1). The error is defined by (7.10).

all the data points was made using the longest stochastic simulation with  $T = 10^5$ . This shows that as the simulation length is increased we see a convergence of the chosen mesh to one which well represents the region containing the invariant density.

**7.4. Parameter  $\beta_1$ .** In certain situations the stochastic simulations may be expensive and we may only be able to take a few samples from the stochastic trajectory. In this case we still have a large amount of uncertainty about the region which contains the vast majority of the invariant density. Therefore we can only hope to approximate this by increasing the size of the ellipsoid around each sampled point (4.2) that we include in the region  $\Gamma$  given by (4.3). Figure 7.3 shows the convergence of the method as  $\beta_1$  is increased, with  $S = 1$ ,  $T = 10^5$ ,  $Q = 0.1$ ,  $B = 10^3$ ,  $\beta_2 = 0.4$ , and  $H = 6$ .

The error as plotted in Figure 7.3 is given by

$$(7.10) \quad \text{Error}(\beta) = \left( \int_{\Omega} |p_{\beta_1=1}(\mathbf{x}) - p_{\beta_1=\beta}(\mathbf{x})|^2 d\mathbf{x} \right)^{1/2}.$$

Note that as  $\beta_1$  increases, it reaches a point where no improvement is seen in the error, because the whole of the domain is covered by the union of ellipses  $\Gamma$  for sufficiently large values of  $\beta_1$ . Once this happens, the mesh generator returns a uniform mesh, and increasing  $\beta_1$  further does not affect the produced mesh.

**7.5. A numerical test of accuracy.** In section 5.3 a brief test for sufficient convergence was suggested, where two solutions were calculated, one with parameters

$$\{S, T, Q, B, \beta_1, \beta_2, H\},$$

and the second with parameters

$$(7.11) \quad \{S, 2T, 2Q, 2B, 2\beta_1, \beta_2, H + 1\}.$$

If the two solutions are close up to some tolerance, then we can be fairly sure that the method has converged to a reasonable degree.

The solution for the system (7.1) calculated with the parameter set

$$\{S = 1, T = 10^5, Q = 0.05, B = 500, \beta_1 = 10^{-3}, \beta_2 = 0.45, H = 7\},$$

when compared with the solution using the parameters

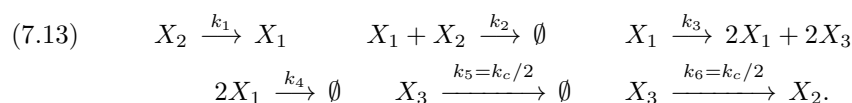
$$\{S = 1, T = 2 \times 10^5, Q = 0.1, B = 10^3, \beta_1 = 2 \times 10^{-3}, H = 8\},$$

gives  $L^2$  error of  $1.99 \times 10^{-4}$ , which is significantly less than the  $L^2$  norm of the solution ( $4.51 \times 10^{-3}$ ). This gives a relative  $L^2$  error of  $4.41 \times 10^{-2}$ . For ease of comparison the domain  $\Omega$  chosen using parameters (7.11) was used for both meshes.

**7.6. Oregonator.** Our final example is the Oregonator [23], which is a three-species chemical system motivated by the Belousov–Zhabotinsky reaction. It exhibits oscillatory behavior and is traditionally given by the following system of ODEs:

$$(7.12) \quad \begin{aligned} \frac{dx_1}{dt} &= k_1 x_2 - k_2 x_1 x_2 + k_3 x_1 - 2k_4 x_1^2, \\ \frac{dx_2}{dt} &= -k_1 x_2 - k_2 x_1 x_2 + \frac{1}{2} k_c x_3, \\ \frac{dx_3}{dt} &= 2k_3 x_1 - k_c x_3. \end{aligned}$$

We now construct a set of reactions whose behavior is given by (7.12) in the mean-field limit:



Note that the reaction with parameter  $k_c$  has been split into two reactions  $R_5$  and  $R_6$  in order to get the factor of half in the equation for the rate of change of  $x_2$  with  $k_5 = k_6 = k_c/2$ . This system (7.13) of  $M = 6$  reactions of  $N = 3$  chemical species has the following propensity functions:

$$\begin{aligned} \alpha_1(\mathbf{x}) &= k_1 x_2, & \alpha_2(\mathbf{x}) &= k_2 x_1 x_2, & \alpha_3(\mathbf{x}) &= k_3 x_1, \\ \alpha_4(\mathbf{x}) &= k_4 x_1 (x_1 - 1), & \alpha_5(\mathbf{x}) &= k_5 x_3, & \alpha_6(\mathbf{x}) &= k_6 x_3. \end{aligned}$$

The stoichiometric matrix is given by

$$\nu = \begin{pmatrix} 1 & -1 & 0 \\ -1 & -1 & 0 \\ 1 & 0 & 2 \\ -2 & 0 & 0 \\ 0 & 0 & -1 \\ 0 & 1 & -1 \end{pmatrix}.$$

We consider this system with the following set of dimensionless parameters:

$$(7.14) \quad k_1 = 0.3, \quad k_2 = 4000, \quad k_3 = 5, \quad k_4 = 1200, \quad k_c = 0.02.$$

In this parameter regime the stochastic description of these reactions exhibits oscillatory behavior. Figure 7.4(a) shows normalized trajectories of the Oregonator (7.13) with parameters given by (7.14), simulated using the Gillespie SSA. This figure demonstrates the oscillatory behavior of the Oregonator in this parameter regime.

One thing of note should be mentioned at this point, regarding the ergodicity of this system. The zero state, at the origin, is an absorbing state for this system, and so trivially the invariant distribution for this system is a Dirac measure on this state. However, we are interested in the behavior of this system conditioned on nonextinction of the species. Therefore we ensure that our domain of solution does not include this

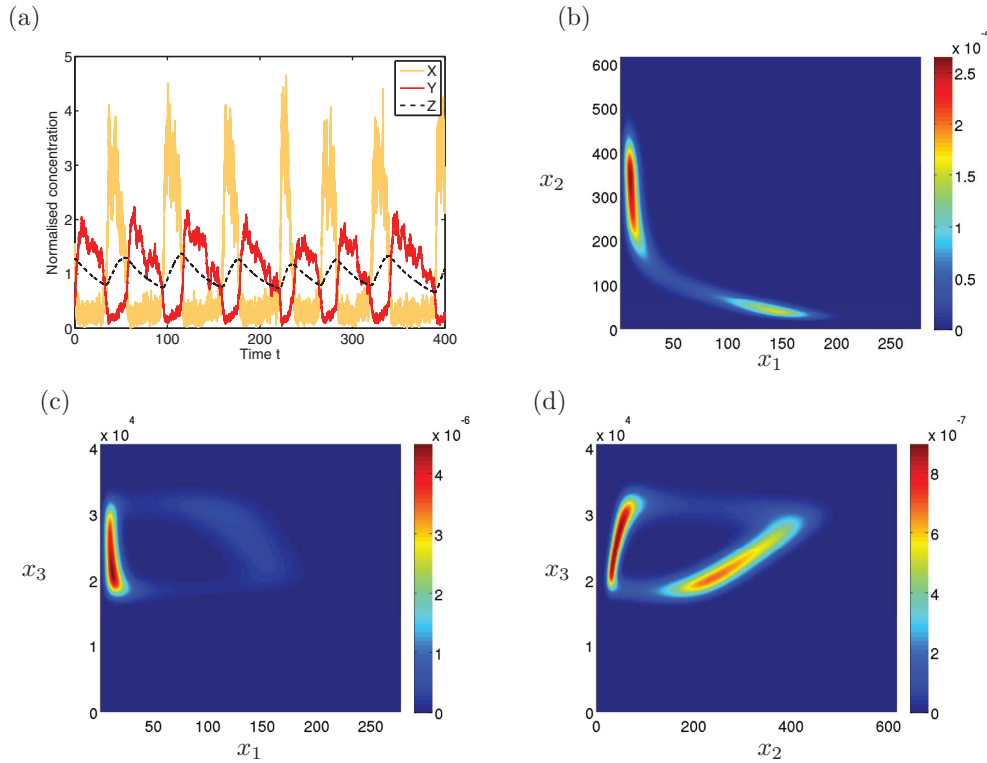


FIG. 7.4. (a) Trajectories of the Oregonator (7.13) with parameters given by (7.14), simulated using the Gillespie SSA. Plots show normalized trajectories with actual molecule numbers divided by a time average ( $47.7$ ,  $211.5$ , and  $2.4 \times 10^4$ , resp.). (b)–(d) Approximation of the marginal invariant distributions (7.16) (conditioned on nonextinction of species  $X_1$ ) of (7.13) with system parameters (7.14) in the (b)  $x_1$ - $x_2$  plane, (c)  $x_1$ - $x_3$  plane, and (d)  $x_2$ - $x_3$  plane by the saFEM with algorithmic parameters given by (7.15).

state, and thus the transient states involved in the oscillation behavior now form the regions with nonzero invariant density.

In the figures that follow, these algorithmic parameters were used to approximate the steady state distribution (conditioned on nonextinction of species  $X_1$ ):

$$(7.15) \quad S = 1, \quad T = 10^6, \quad Q = 10^{-2}, \quad B = 10^3, \quad H = 8.$$

Since the system (7.12) has a single limit cycle, step [1] of the algorithm in Table 4.1 can be replaced by running the Gillespie SSA for a period of algorithm time until we are reasonably sure that the chain has entered probabilistic equilibrium and using the endpoint of this simulation as the starting point for the SSA in step [2]. We have  $S = 1$  and the initial condition  $(x_1, x_2, x_3) = (100, 100, 100)$  was chosen.

Since we wish to condition on nonextinction of all of the species, we must ensure that  $X_1$  never drops below 1. We can do this by slightly altering the domain  $\Omega$  of solution of the Fokker–Planck equation. We do this by replacing the first line of (4.6) by  $\Omega = \mathcal{A}_1^* \times \mathcal{A}_2 \times \mathcal{A}_3$ , where

$$\mathcal{A}_1^* = (\max\{1, x_1^{\min} - \beta_2 x_1^{\text{range}}\}, x_1^{\max} + \beta_2 x_1^{\text{range}}).$$

The longer the simulation in step [2] of the saFEM, the more sure we can be of the

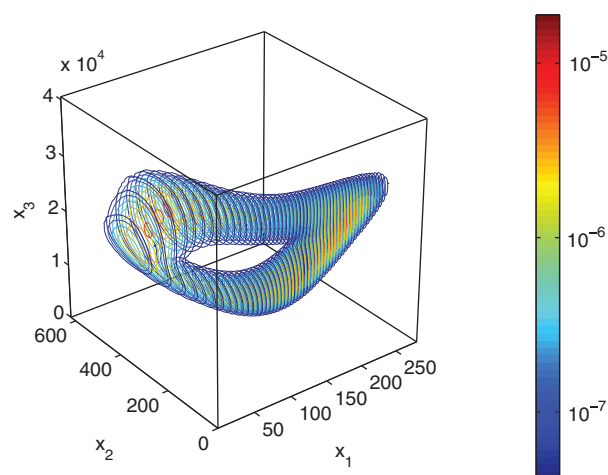


FIG. 7.5. Approximation of the stationary probability distribution (conditioned on nonextinction) of (7.13) with system parameters (7.14). Contours plots shown in several slices.

regions which have nonnegligible probability density and therefore the larger we can make  $\beta_2$  and  $\beta_1$ . We choose  $\beta_1 = 0.01$  and  $\beta_2 = 0.1$ .

Using these parameters, a mesh was created by the saFEM with  $8.65 \times 10^5$  vertices, out of a possible  $1.70 \times 10^7$  with up to eight splits of each cuboid, with  $1.17 \times 10^5$  hanging nodes. Therefore there were a total of  $7.48 \times 10^5$  degrees of freedom, a mere 4.4% of the degrees of freedom that would have been required to fill the domain with cuboids of the finest refinement level. The mesh contained  $4.79 \times 10^6$  individual tetrahedral elements. The finest saFEM mesh had the smallest (resp., largest) element diameters of  $1.58 \times 10^2$  (resp.,  $1.01 \times 10^4$ ). Note that these diameters are large due to the anisotropic refinements in the algorithm due to the domain shape.

Figure 7.5 shows several slices of the approximated invariant probability density conditioned on nonextinction of species  $X_1$ , represented by contour plots. Since visualization of a three-dimensional function on a two-dimensional page is problematic, we present also in Figure 7.4(b)–(d) the marginal densities of the approximation of the invariant density of the Oregonator system with parameters (7.14) in three different planes. The marginal density in the  $x_i - x_j$  plane is defined to be

$$(7.16) \quad p_{x_i - x_j}(\mathbf{x}) = \int_{x_k} p(\mathbf{x}).$$

To verify that the method is accurately representing the invariant distribution, we can test the approximation as suggested in section 5.3. The approximation was compared with one with the parameters given as follows:

$$S = 1, \quad T = 5 \times 10^5, \quad Q = 5 \times 10^{-3}, \quad B = 5 \times 10^2, \quad H = 7, \quad \beta_1 = 5 \times 10^{-3}.$$

We omit  $\beta_2$  here as the same domain chosen for the original parameter set was used for ease of comparison of the two densities. The  $L^2$  difference between the two solutions was  $2.89 \times 10^{-11}$  with the original solution having an  $L^2$  norm of  $1.74 \times 10^{-8}$ , giving a relative  $L^2$  error of  $1.66 \times 10^{-3}$ , verifying that our solution is well converged.

Furthermore, we can undertake a comparison with the results of the standard adaptive algorithm with a posteriori error indicator as described in section 7.2. Figure 7.6(a) shows the errors of the two methods, as computed by comparison with an

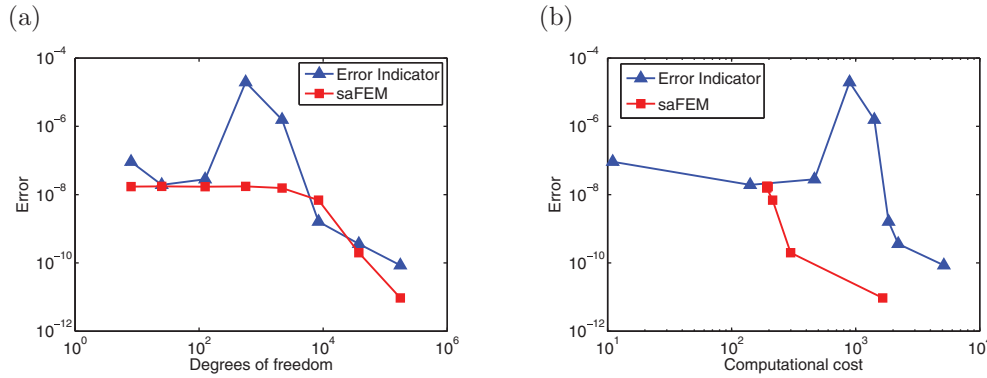


FIG. 7.6. (a) Convergence of the saFEM (red squares) compared with the standard error indicator approach (blue triangles), with  $S = 1$ ,  $T = 1 \times 10^6$ ,  $Q = 0.01$ ,  $B = 10^3$ ,  $\beta_1 = 0.02$ , and  $\beta_2 = 0.1$  over a range of mesh refinements, for the system (7.13). Errors are given by (7.8). (b) Comparison of cost/error for the saFEM (red squares) and the standard error indicator approach (blue triangles), with  $S = 1$ ,  $T = 1 \times 10^6$ ,  $Q = 0.01$ ,  $B = 10^3$ ,  $\beta_1 = 0.02$ , and  $\beta_2 = 0.1$  over a range of mesh refinements, for the system (7.1). Errors are given by (7.8).

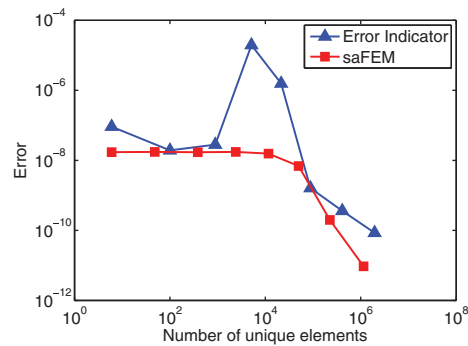


FIG. 7.7. Convergence of the saFEM (red squares) compared with the standard error indicator approach (blue triangles) as a function of the total number of unique elements used, with  $S = 1$ ,  $T = 1 \times 10^6$ ,  $Q = 0.01$ ,  $B = 10^3$ ,  $\beta_1 = 0.02$ , and  $\beta_2 = 0.1$  over a range of mesh refinements, for the system (7.13). Errors are given by (7.8).

approximation with a very fine mesh ( $1.06 \times 10^6$  and  $1.02 \times 10^6$  degrees of freedom, resp.). Note that as in Figure 7.1, the errors are of the order  $10^{-8}$  even for low numbers of degrees of freedom as the  $L^2$  norm of the solution itself is of the order  $10^{-8}$ . The computational cost plotted in Figure 7.6(b) was calculated in the same way as Figure 7.1(b). Note that the saFEM incurs less error for the number of degrees of freedom than the standard method. Since the computational cost for a given number of degrees of freedom is less for the saFEM than the standard approach, the saFEM is the better choice for this system.

This difference would be in even greater evidence if the drift and diffusion coefficients were expensive to calculate, as discussed in section 1. This can be seen in Figure 7.7, which shows the decay of error as a function of the total number of unique elements used in the approximation of the invariant density for the two methods. For the saFEM, this is simply the number of elements  $L \in \mathbb{N}$  present in the mesh. For the standard method, this includes all the unique elements used in all the meshes in

all previous iterations of the method. Since we start with a mesh with six elements, this is equal to  $2 \times L^{\text{final}} - 6$ , where  $L^{\text{final}}$  stands for the number of elements in the final, i.e., the finest, mesh.

Each unique element would require a number of evaluations of the drift and diffusion terms, depending on the quadrature rule used. Therefore, if the cost of estimation of these terms dominates the complexity of the problem, then the total number of unique elements used is approximately proportional to the computational cost. Figure 7.7 shows that the saFEM requires far fewer function evaluations than the standard method for a given order of accuracy for this system.

**8. Discussion.** In this paper we presented a method for solution of the steady state Fokker–Planck equation for chemical systems. The method is based on the use of stochastic trajectories to estimate the region in which we must refine our finite element mesh the most. We showed numerically that the saFEM is comparable with standard methods for the systems analyzed. Moreover, for systems which are sensitive to global mesh structure, such as oscillatory systems, the saFEM was found to outperform the standard error indicator method.

The use of the Fokker–Planck equation as a means to model chemical reactions of the mesoscale has been present in the literature [25, 40]. As computational resources have improved, the solving of such equations in more than one dimension has become tractable. It has been shown using finite volume methods that the solution of the Fokker–Planck equation is more efficient than using an SSA for the computation of both time-dependent and steady state solutions, and with a high degree of accuracy [22, 40].

The Fokker–Planck equation does suffer badly from the curse of dimensionality, and different methods exist which find approximations of the solution of these equations efficiently. Some involve dimension reduction of the equations themselves, through simplifying assumptions, and through coupling with macroscopic reaction rate equations for some of the species [35]. Sparse grid methods have also been used to find the invariant density of the chemical master equation [29]. Sparse grids overcome the curse of dimensionality by reducing greatly the number of degrees of freedom considered. This solution of the chemical master equation can also be coupled to Fokker–Planck equations for the more abundant species using a hybrid method [29]. Other methods have been used to approximate the solution of the chemical master equation, including adaptive wavelet compression [31] and finite state projection [16, 37].

There are several alternative ways that stochastic trajectories could be calculated in the saFEM, other than the standard SSA as detailed in Table 2.1. The  $\tau$ -leap method [27] approximates the trajectory by modeling several reactions in each iteration of the algorithm, up to a fixed time step. This can accelerate the simulation of the trajectory but could incur some errors, including the possibility that the trajectory may become negative in one or more of the chemical species. Likewise, one could use numerical approximations of the chemical Langevin equation [26], a stochastic differential equation (SDE) with a corresponding Fokker–Planck equation given by (1.1), in step [2] in Table 4.1. As with the  $\tau$ -leap method, the trajectories from this SDE can become negative, and so with both these methods proper treatment of boundary conditions is key to accurately approximating the region of the positive quadrant that contains the majority of the probability density. Hybrid methods can be formulated where the trajectory is simulated using the SSA in regions close to zero and by the SDE elsewhere [29]. In summary, any approach that is ordinarily used in order to accelerate the sampling of a trajectory could be used.

More information could also be extracted from the stochastic trajectories. A rudimentary approximation of the density could be made using the samples taken from the simulations and used as a preconditioner for the eigensolver. As the eigensolver used is iterative, even a rough guess of the form of the solution could help to reduce computation times. It has been previously shown that combining stochastic simulations and solutions of differential equations can be used for preconditioning of computations [38].

Our aim in this work is to efficiently approximate the steady state solution of the chemical master equation (2.2) through approximation of the solutions of the stationary chemical Fokker–Planck equation (1.1), itself a continuous approximation of (2.2). The saFEM can still be used only for systems for which this continuous approximation to the solution of (2.2) is “good enough,” i.e., systems which only rarely enter regions where the copy number of one (or more) of the chemical species becomes low [26]. In some biological systems, chemical species can oscillate between low copy numbers and regions of the state space where they are more abundant. Thus, a simultaneous treatment of the two domains is key to understanding these types of systems. This method is efficient only for systems which do not have large regions of the domain with nonnegligible invariant density that can be accurately approximated by a linear function. Our assumption that regions which have high enough probability density require high levels of refinement would be invalid in this case, and the mesh produced would be inefficient.

Finally, the ideas explored in this paper could be used in conjunction with other numerical methods for finding the solution of the stationary Fokker–Planck equation, for instance, finite volume methods [40]. Exploiting the links between the PDE and the stochastic process can lead to efficient and accurate methods for approximation of the solution of the PDE.

## REFERENCES

- [1] P. AMESTOY, I. DUFF, J. L’EXCELLENT, AND J. KOSTER, *A fully asynchronous multifrontal solver using distributed dynamic scheduling*, SIAM J. Matrix Anal. Appl., 23 (2001), pp. 15–41.
- [2] P. AMESTOY, A. GUERMOUCHE, J. L’EXCELLENT, AND S. PRALET, *Hybrid scheduling for the parallel solution of linear systems*, Parallel Comput., 32 (2006), pp. 136–156.
- [3] A. ARKIN, J. ROSS, AND H. MCADAMS, *Stochastic kinetic analysis of developmental pathway bifurcation in phage *l*-infected *Escherichia coli* cells*, Genetics, 149 (1998), pp. 1633–1648.
- [4] I. BABUŠKA AND W. RHEINBOLDT, *Error estimates for adaptive finite element computations*, SIAM J. Numer. Anal., 15 (1978), pp. 736–754.
- [5] I. BABUŠKA AND W. RHEINBOLDT, *On the reliability and optimality of the finite element method*, Comput. & Structures, 10 (1979), pp. 87–94.
- [6] S. BALAY, J. BROWN, K. BUSCHELMAN, W. GROPP, D. KAUSHIK, M. KNEPLEY, L. MCINNES, B. SMITH, AND H. ZHANG, *PETSc*, <http://www.mcs.anl.gov/petsc> (2011).
- [7] S. BALAY, W. GROPP, L. MCINNES, AND B. SMITH, *Efficient management of parallelism in object oriented numerical software libraries*, in Modern Software Tools in Scientific Computing, Birkhäuser Boston, Cambridge, MA, 1997, pp. 163–202.
- [8] L. BEILINA, S. KOROTOV, AND M. KRÍŽEK, *Nonobtuse tetrahedral partitions that refine locally towards Fichera-like corners*, Appl. Math., 50 (2005), pp. 569–581.
- [9] J. BEY, *Tetrahedral grid refinement*, Computing, 55 (1995), pp. 355–378.
- [10] Y. CAO, D. GILLESPIE, AND L. PETZOLD, *Multiscale stochastic simulation algorithm with stochastic partial equilibrium assumption for chemically reacting systems*, J. Comput. Phys., 206 (2005), pp. 395–411.
- [11] Y. CAO, D. GILLESPIE, AND L. PETZOLD, *The slow-scale stochastic simulation algorithm*, J. Chem. Phys., 122 (2005), 14116.
- [12] P. CIARLET, *The Finite Element Method for Elliptic Problems*, Stud. Math. Appl. 4, North-Holland, Amsterdam, 1978.

- [13] S. COTTER, K. ZYGALAKIS, I. KEVREKIDIS, AND R. ERBAN, *A constrained approach to multiscale stochastic simulation of chemically reacting systems*, J. Chem. Phys., 135 (2011), 094102.
- [14] M. COWLES AND B. CARLIN, *Markov chain Monte Carlo convergence diagnostics: A comparative review*, J. Amer. Statist. Assoc., 91 (1996), pp. 883–904.
- [15] E. DOEDEL, A. CHAMPNEYS, T. FAIRGRIEVE, Y. KUZNETSOV, B. SANDSTEDTE, AND X. WANG, *AUTO 97: Continuation and bifurcation software for ordinary differential equations (with HomCont)*, 1997, <http://indy.cs.concordia.ca/auto/>.
- [16] B. DRAWERT, M. LAWSON, L. PETZOLD, AND M. KHAMMASH, *The diffusive finite state projection algorithm for efficient simulation of the stochastic reaction-diffusion master equation*, J. Chem. Phys., 132 (2010), 074101.
- [17] W. E, D. LIU, AND E. VANDEN-EIJNDEN, *Nested stochastic simulation algorithm for chemical kinetic systems with disparate rates*, J. Chem. Phys., 123 (2005), 194107.
- [18] R. ERBAN, S. J. CHAPMAN, I. KEVREKIDIS, AND T. VEJCHODSKÝ, *Analysis of a stochastic chemical system close to a SNIPER bifurcation of its mean-field model*, SIAM J. Appl. Math., 70 (2009), pp. 984–1016.
- [19] R. ERBAN, S. J. CHAPMAN, AND P. MAINI, *A Practical Guide to Stochastic Simulations of Reaction-Diffusion Processes*, <http://arxiv.org/abs/0704.1908>, 2007.
- [20] R. ERBAN, I. KEVREKIDIS, D. ADALSTEINSSON, AND T. ELSTON, *Gene regulatory networks: A coarse-grained, equation-free approach to multiscale computation*, J. Chem. Phys., 124 (2006), 084106.
- [21] L. EVANS, *Partial Differential Equations*, Grad. Stud. Math. 19, AMS, Providence, RI, 1998.
- [22] L. FERM, P. LÖTSTEDT, AND P. SJÖBERG, *Conservative solution of the Fokker-Planck equation for stochastic chemical reactions*, BIT, 46 (2006), pp. 61–83.
- [23] R. FIELD AND R. NOYES, *Oscillations in chemical systems. IV. Limit cycle behavior in a model of a real chemical reaction*, J. Chem. Phys., 60 (1974), pp. 1877–1884.
- [24] D. GILLESPIE, *Exact stochastic simulation of coupled chemical reactions*, J. Phys. Chem., 81 (1977), pp. 2340–2361.
- [25] D. GILLESPIE, *Approximating the master equation by Fokker-Planck equations for single-variable chemical systems*, J. Chem. Phys., 72 (1980), pp. 5363–5370.
- [26] D. GILLESPIE, *The chemical Langevin equation*, J. Chem. Phys., 113 (2000), pp. 297–306.
- [27] D. GILLESPIE, *Approximate accelerated stochastic simulation of chemically reacting systems*, J. Chem. Phys., 115 (2001), pp. 1716–1733.
- [28] R. GRIMA, *Construction and accuracy of partial differential equation approximations to the chemical master equation*, Phys. Rev. E, 84 (2011), 056109.
- [29] M. HEGLAND, A. HELLANDER, AND P. LÖTSTEDT, *Sparse grids and hybrid methods for the chemical master equation*, BIT, 48 (2008), pp. 265–283.
- [30] V. HERNANDEZ, J. ROMAN, AND V. VIDAL, *SLEPc: A scalable and flexible toolkit for the solution of eigenvalue problems*, ACM Trans. Math. Software, 31 (2005), pp. 351–362.
- [31] T. JAHNKE AND T. UDRESCU, *Solving chemical master equations by adaptive wavelet compression*, J. Comput. Phys., 229 (2010), pp. 5724–5741.
- [32] S. KAR, W. BAUMANN, M. PAUL, AND J. TYSON, *Exploring the roles of noise in the eukaryotic cell cycle*, Proc. Nat. Acad. Sci. USA, 106 (2009), pp. 6471–6476.
- [33] S. KOROTOV AND M. KEK, *Acute type refinements of tetrahedral partitions of polyhedral domains*, SIAM J. Numer. Anal., 39 (2002), pp. 724–733.
- [34] V. KUBLANOVSKAYA, *On some algorithms for the solution of the complete eigenvalue problem*, USSR Comput. Math. Math. Phys., 1 (1961), pp. 555–570.
- [35] P. LÖTSTEDT AND L. FERM, *Dimensional reduction of the Fokker-Planck equation for stochastic chemical reactions*, Multiscale Model. Simul., 5 (2006), pp. 593–614.
- [36] J. MATTINGLY, A. STUART, AND D. HIGHAM, *Ergodicity for SDEs and approximations: Locally Lipschitz vector fields and degenerate noise*, Stochastic Process. Appl., 101 (2002), pp. 185–232.
- [37] B. MUNSKY AND M. KHAMMASH, *The finite state projection algorithm for the solution of the chemical master equation*, J. Chem. Phys., 124 (2006), 044104.
- [38] L. QIAO, R. ERBAN, C. KELLEY, AND I. KEVREKIDIS, *Spatially distributed stochastic systems: Equation-free and equation-assisted preconditioned computation*, J. Chem. Phys., 125 (2006), 204108.
- [39] K. SIKORSKI, *A three-dimensional analogue to the method of bisections for solving nonlinear equations*, Math. Comput., 33 (1979), pp. 722–738.
- [40] P. SJÖBERG, P. LÖTSTEDT, AND J. ELF, *Fokker-Planck approximation of the master equation in molecular biology*, Comput. Vis. Sci., 12 (2009), pp. 37–50.
- [41] P. ŠOLÍN, J. ČERVENÝ, AND I. DOLEŽEL, *Arbitrary-level hanging nodes and automatic adaptivity in the hp-FEM*, Math. Comput. Simulation, 77 (2008), pp. 117–132.

- [42] R. VERFÜRTH, *A Review of A Posteriori Error Estimation and Adaptive Mesh-Refinement Techniques*, John Wiley & Sons, New York, 1996.
- [43] J. VILLAR, H. KUEH, N. BARKAI, AND S. LEIBLER, *Mechanisms of noise-resistance in genetic oscillators*, Proc. Nat. Acad. Sci. USA, 99 (2002), pp. 5988–5992.
- [44] C. ZENGER, *Sparse grids*, in Parallel Algorithms for Partial Differential Equations, W. Hackbusch, ed., Notes Numer. Fluid Mech. 31, Vieweg-Verlag, Braunschweig, Germany, 1991, pp. 241–251.