Adaptive Multilevel Monte Carlo Simulation

Håkon Hoel, Erik von Schwerin, Anders Szepessy, and Raúl Tempone

Abstract This work generalizes a multilevel forward Euler Monte Carlo method introduced in [3] for the approximation of expected values depending on the solution to an Itô stochastic differential equation. The work [3] proposed and analyzed a forward Euler multilevel Monte Carlo method based on a hierarchy of uniform time discretizations and control variates to reduce the computational effort required by a standard, single level, Forward Euler Monte Carlo method. This work introduces an adaptive hierarchy of non uniform time discretizations, generated by an adaptive algorithm introduced in [10, 9, 2]. This form of the adaptive algorithm generates stochastic, path dependent, time steps and is based on a posteriori error expansions first developed in [12]. Our numerical results for a stopped diffusion problem, exhibit savings in the computational cost to achieve an accuracy of $\mathscr{O}(\mathrm{TOL})$, from $\mathscr{O}\left(\mathrm{TOL}^{-3}\right)$ using a single level version of the adaptive algorithm to $\mathscr{O}\left(\left(\mathrm{TOL}^{-1}\log(\mathrm{TOL})\right)^2\right)$.

H. Hoel
Department of Numerical Analysis, CSC, Royal Institute of Technology (KTH), Stockholm, Sweden, e-mail: hhoel@csc.kth.se

E. von Schwerin
Applied Mathematics and Computational Sciences, KAUST, Thuwal, Saudi Arabia, e-mail: erik.vonschwerin@kaust.edu.sa

A. Szepessy
Department of Mathematics, Royal Institute of Technology (KTH), Stockholm, Sweden, e-mail: szepessy@kth.se

R. Tempone
Applied Mathematics and Computational Sciences, KAUST, Thuwal, Saudi Arabia, e-mail: raul.tempone@kaust.edu.sa

1 Introduction

This work develops a multilevel version of an adaptive algorithm for weak approximation of Itô stochastic differential equations (SDEs)

$$dX(t) = a(t,X(t))dt + b(t,X(t))dW(t), \qquad 0 < t < T, \tag{1}$$

where $X(t;\omega)$ is a stochastic process in $\mathbb{R}^d$, with randomness generated by a k-dimensional Wiener process with independent components, $W(t;\omega)$, on the probability space $(\Omega,\mathscr{F},P)$; see [7, 11]. The functions $a(t,x) \in \mathbb{R}^d$ and $b(t,x) \in \mathbb{R}^{d\times k}$ are given drift and diffusion fluxes.

Our goal is to, for any given sufficiently well behaved function $g : \mathbb{R}^d \to \mathbb{R}$, approximate the expected value $E[g(X(T))]$ by adaptive multilevel Monte Carlo methods. A typical example of such an expected value is to compute option prices in mathematical finance; see [6] and [5]. Other models based on stochastic dynamics are used for example in molecular dynamics simulations at constant temperature and for stochastic climate prediction; cf. [1] and [8].

The multilevel Monte Carlo method based on uniform time stepping was introduced by Giles in [3]. He developed a clever control variate type variance reduction technique for a numerical method, denoted here by $\overline{X}$, that approximates the solution of the SDE (1). The key to the variance reduction in [3] is to compute approximate solutions, $\overline{X}_\ell$, on hierarchies of uniform time meshes with size

$$\Delta t_\ell = C^{-\ell}\Delta t_0, \qquad C \in \{2,3,\ldots\} \quad \text{and} \quad \ell \in \{0,1,\ldots,L\}, \tag{2}$$

thereby generating sets of realizations on different mesh levels. After computing numerical approximations on a mesh hierarchy, the expected value $E[g(X(T))]$ is approximated by the multilevel Monte Carlo estimator

$$\begin{aligned}\mathscr{E}_{\{\mathscr{S}_\ell\}_{\ell=0}^{L}}\left(g(\overline{X}_L(T))\right) = &\sum_{i=1}^{M_0} \frac{g(\overline{X}_0(T;\omega_{i,0}))}{M_0} \\ &+ \sum_{\ell=1}^{L}\sum_{i=1}^{M_\ell} \frac{g(\overline{X}_\ell(T;\omega_{i,\ell})) - g(\overline{X}_{\ell-1}(T;\omega_{i,\ell}))}{M_\ell}.\end{aligned} \tag{3}$$

Here $\{\mathscr{S}_\ell\}_{\ell=0}^{L}$ denotes mutually independent sample sets on the respective meshes, each with M_ℓ independent samples. To reduce the variance in the above estimator, the realization pairs $\overline{X}_\ell(T;\omega_{i,\ell})$ and $\overline{X}_{\ell-1}(T;\omega_{i,\ell})$ of the summands $g(\overline{X}_\ell(T;\omega_{i,\ell})) - g(\overline{X}_{\ell-1}(T;\omega_{i,\ell}))$ for each level $\ell > 0$ are generated by the same Brownian path, $W_t(\omega_i)$, but they are realized on different temporal grids with uniform time steps, Δt_ℓ and $\Delta t_{\ell-1}$, respectively. The efficiency of this computation relies on an a priori known order of strong convergence for the numerical method employed on each level of the hierarchy.

Let TOL > 0 be a desired accuracy in the approximation of $E[g(X(T))]$. The main result of Giles' work [3] is that the computational cost needed to achieve the

Mean Square Error (MSE)

$$E\left[\left(\mathscr{E}_{\{\mathscr{S}_\ell\}_{\ell=0}^L}\left(g(\overline{X}_L(T))\right) - E[g(X(T))]\right)^2\right] = \mathscr{O}\left(\mathrm{TOL}^2\right), \tag{4}$$

when using the Forward Euler method to create the approximate realizations $\overline{X}_\ell(T;\omega)$, can be reduced to

$$\mathscr{O}\left((\mathrm{TOL}^{-1}\log(\mathrm{TOL}^{-1}))^2\right),$$

with Giles' multilevel Monte Carlo method; the corresponding complexity using the standard Monte Carlo method is $\mathscr{O}\left(\mathrm{TOL}^{-3}\right)$ since the Forward Euler method has weak order of convergence 1 and the Monte Carlo sampling order $1/2$ by the Central Limit Theorem. Furthermore, whenever the function g is Lipschitz and for scalar Itô stochastic differential equations, the computational cost can be further reduced to $\mathscr{O}\left(\mathrm{TOL}^{-2}\right)$ using the first order strong convergence Milstein method. In addition, the work [4] shows how to apply the Milstein method for several scalar SDE cases where the Lipschitz condition is not fulfilled and still obtain the cost $\mathscr{O}\left(\mathrm{TOL}^{-2}\right)$.

In this work we use the Forward Euler method with non uniform time steps. Let $0 = t_0 < t_1 < \cdots < t_N = T$ denote a given time discretization, without reference to its place in the hierarchies, and $\{0 = W(t_0;\omega), W(t_1;\omega), \ldots, W(t_N;\omega)\}$ denote a generated sample of the Wiener process on that discretization. Then the Forward Euler method computes an approximate solution of (1) by the scheme

$$\begin{aligned}\overline{X}(t_0;\omega) &= X(0),\\ \overline{X}(t_{n+1};\omega) &= a(\overline{X}(t_n;\omega),t_n)\Delta t_n + b(\overline{X}(t_n;\omega),t_n)\Delta W(t_n;\omega), \qquad n \geq 0,\end{aligned} \tag{5}$$

where $\Delta t_n = t_{n+1} - t_n$ and $\Delta W(t_n;\omega) = W(t_{n+1};\omega) - W(t_n;\omega)$ are the time steps and Wiener increments, respectively.

The contribution of the present paper to the multilevel Monte Carlo method is the development of a novel algorithm with adaptive, non uniform time steps. The algorithm uses adaptive mesh refinements to stochastically create a path dependent mesh for each realization. The construction and analysis of the adaptive algorithm is inspired by the work on single level adaptive algorithms for weak approximation of ordinary stochastic differential equations [9], and uses the adjoint weighted global error estimates first derived in [12]. The goal of the adaptive algorithm is to choose the time steps and the number of realizations such that the event

$$\left|\mathscr{E}_{\{\mathscr{S}_\ell\}_{\ell=0}^L}\left(g(\overline{X}_L(T))\right) - E[g(X(T))]\right| \leq \mathrm{TOL}, \tag{6}$$

holds with probability close to one.

It should be noted that in the setting of adaptive mesh refinement there is no given notion of mesh size, so a hierarchy of meshes can no longer be described as in the constant time step case (2). Instead, we generate a hierarchy of meshes by successively increasing the accuracy in our computations: setting the tolerance levels

$$\mathrm{TOL}_\ell = \frac{\mathrm{TOL}_0}{2^\ell}, \quad \text{for} \quad \ell \in \{0,1,\ldots,L\}, \tag{7}$$

and (by adaptive refinements based on error indicators) finding corresponding meshes so that for each level $\ell \in \{0,1,\ldots,L\}$,

$$\left|E[g(X(T))] - E[g(\overline{X}_\ell(T))]\right| \lesssim \frac{\mathrm{TOL}_\ell}{2}.$$

The efficiency and accuracy of the multilevel adaptive Monte Carlo algorithm is illustrated by a numerical example, in the case of the stopped diffusion problems used to test the single level version of the algorithm in [2]. For this example multilevel Monte Carlo based on adaptive time steps requires a computational work $\mathscr{O}\left(\mathrm{TOL}^{-2}\log(\mathrm{TOL}^{-1})^2\right)$ while a direct application of the multilevel Monte Carlo method based on uniform time steps would be less efficient since the underlying Euler–Maruyama method has reduced orders of weak and strong convergence for the barrier problem.

The rest of this paper is organized as follows: Subsection 1.1 introduces the notion of error density and error indicators, and recalls useful results for single level adaptive forward Euler Monte Carlo methods. Sect. 2 describes the new adaptive multilevel Monte Carlo algorithm. Sect. 3 presents results from the numerical experiment.

1.1 A Single Level Posteriori Error Expansion

Here we recall previous single level results that are used for constructing the multilevel algorithm in Sect. 2. In particular, we recall adjoint based error expansions with computable leading order term. Assume that the process X satisfies (1) and its approximation, $\overline{X}$, is given by (5); then the error expansions in theorems 1.2 and 2.2 of [12] have the form

$$E[g(X(T)) - g(\overline{X}(T))] = E\left[\sum_{n=1}^{N} \rho_n \Delta t_n^2\right] + \text{higher order terms}, \tag{8}$$

where $\rho_n \Delta t_n^2$ are computable error indicators, that is they provide information for further improvement of the time mesh and ρ_n measures the density of the global error in (8). A typical adaptive algorithm does two things iteratively:

1. if the error indicators satisfy an accuracy condition then it stops; otherwise
2. the algorithm chooses where to refine the mesh based on the error indicators and then makes an iterative step to 1.

In addition to estimating the global error $E[g(X(T)) - g(\overline{X}(T))]$ in the sense of equation (8), the error indicators $\rho_n \Delta t_n^2$ also give simple information on where to refine to reach an optimal mesh, based on the almost sure convergence of the density ρ_n as we refine the discretization, see Sect. 4 in [10].

In the remaining part of this section we state in Theorem 1 a single level error expansion from [12].

Given an initial time discretization $\Delta t[0](t)$ and, for the stochastic time steps algorithm, refining until[1]

$$|\rho(t,\omega)|\big(\Delta t(t)\big)^2 < \text{constant}, \tag{9}$$

we construct a partition $\Delta t(t)$ by repeated halving of intervals so that it satisfies

$$\Delta t(t) = \Delta t[0](t)/2^n \quad \text{for some natural number } n = n(t,\omega).$$

The criterion (9) uses an approximate error density function ρ, satisfying for $t \in [0,T]$ and all outcomes ω the uniform upper and lower bounds

$$\rho_{low}(\mathrm{TOL}) \le |\rho(t,\omega)| \le \rho_{up}(\mathrm{TOL}). \tag{10}$$

The positive functions ρ_{low} and ρ_{up} are chosen so that $\rho_{up}(\mathrm{TOL}) \to +\infty$ as $\mathrm{TOL} \to 0$ while $\rho_{low}(\mathrm{TOL}) \to 0$ such that $\mathrm{TOL}/\rho_{low}(\mathrm{TOL}) \to 0$. We further make the assumption that for all $s,t \in [0,T]$ the sensitivity of the error density to values of the Wiener process can be bounded,

$$|\partial_{W(t)}\rho(s,\omega)| \le D\rho_{up}(\mathrm{TOL}), \tag{11}$$

for some positive function $D\rho_{up}$ such that $D\rho_{up}(\mathrm{TOL}) \to +\infty$ as $\mathrm{TOL} \to 0$. For each realization successive subdivisions of the steps yield the largest time steps satisfying (9). The corresponding stochastic increments ΔW will have the correct distribution, with the necessary independence, if the increments ΔW related to the new steps are generated by Brownian bridges [7], that is the time steps are generated by conditional expected values of the Wiener process.

We begin now by stating in the next lemma the regularity conditions to be used in the analysis of the adaptive multilevel algorithms.

Lemma 1 (Regularity). (a) *Assume that the following regularity conditions hold:*

(1)*The functions $a(t,x)$ and $b(t,x)$ are continuous in (t,x) and are twice continuously differentiable with respect to x.*
(2)*The partial derivatives of first and second order with respect to x of the functions a and b are uniformly bounded.*
(3)*The function g is twice continuously differentiable, and together with its partial derivatives of first and second order it is uniformly bounded.*

Then the cost to go *function, defined by*

$$u(t,x) = E\big[g(X(T)) \mid X(t) = x\big], \tag{12}$$

[1] The precise expression is given in (34) below.

satisfies the Kolmogorov equation

$$\partial_t u(t,x) + a_k \partial_k u(t,x) + d_{kn} \partial_{kn} u(t,x) = 0, \qquad u(T,\cdot) = g, \tag{13}$$

where we have used Einstein summation convention[2]*, and where* $d_{kn} = \frac{1}{2} b_k^l b_n^l$.

(b) *Furthermore, if the following regularity conditions are satisfied:*

(1)*The functions* $\partial_\beta a(t,\cdot)$ *and* $\partial_\beta b(t,\cdot)$ *are bounded uniformly in* t *for multi-indices* β *with* $1 \le |\beta| \le 8$*;*
(2)*The functions* $a(\cdot,x)$*,* $b(\cdot,x)$ *have continuous and uniformly bounded first order time derivatives;*
(3)*The function* g *has spatial derivatives* $\partial_\beta g$*, with polynomial growth for* $|\beta| \le 8$*;*

then the function u *has continuous partial derivatives with respect to* x *up to the order* 8*, satisfying the following polynomial growth condition: for all* $i \in \{0,1,2\}$ *and* $\alpha \in \mathbb{N}^d$ *with* $i + |\alpha| \le 8$ *there exists* $p_{\alpha,i} \in \mathbb{N}$ *and* $C_{\alpha,i} > 0$ *such that*

$$\max_{0 \le t \le T} \left| \partial_t^i \partial_\alpha u(t,x) \right| \le C_{\alpha,i} \left(1 + |x|^{p_{\alpha,i}} \right) \quad \forall x \in \mathbb{R}^d.$$

In what follows, Lemma 2 and Theorem 1 show that although the steps adaptively generated to satisfy (9)–(11) are not adapted to the natural Wiener filtration, the method indeed converges to the correct limit, which is the same as the limit of the forward Euler method with adapted time steps.

Lemma 2 (Strong Convergence). *For* X *the solution of* (1) *suppose that* a*,* b*, and* g *satisfy the assumptions in Lemma 1, that* $\overline{X}$ *is constructed by the forward Euler method, based on the stochastic time stepping algorithm defined in Sect. 2, with step sizes* Δt_n *satisfying* (9)–(11)*, and that their corresponding* ΔW_n *are generated by Brownian bridges. Then*

$$\sup_{0 \le t \le T} E[|X(t) - \overline{X}(t)|^2] = \mathscr{O}\left(\Delta t_{\mathrm{sup}}\right) = \mathscr{O}\left(\frac{\mathrm{TOL}}{\rho_{low}(\mathrm{TOL})}\right) \longrightarrow 0 \tag{14}$$

as $\mathrm{TOL} \to 0$*, where* $\Delta t_{\mathrm{sup}} \equiv \sup_{n,\omega} \Delta t_n(\omega)$.

In Theorem 1 and the rest of this work, we will use Einstein summation convention with respect to functional and spatial indices, but not with respect to the temporal one (usually denoted t_n).

Theorem 1 (Single level stochastic time steps error expansion). *Given the assumptions in Lemma 2 and a deterministic initial value* $X(0)$*, the time discretization error in* (8) *has the following expansion, based on both the drift and diffusion fluxes and the discrete dual functions* φ*,* φ'*, and* φ'' *given in* (17)–(22)*, with computable leading order terms:*

[2] When an index variable appears twice in a single term this means that a summation over all possible values of the index takes place; for example $a_k \partial_k u(t,x) = \sum_{k=1}^d a_k \partial_k u(t,x)$, where d is the space dimension of the SDE ($a, x \in \mathbb{R}^d$).

$$
\begin{aligned}
E[g(X(T))]-E[g(\overline{X}(T))] = {}& E\left[\sum_{n=0}^{N-1}\tilde{\rho}(t_n,\omega)(\Delta t_n)^2\right] \\
&+\mathcal{O}\left(\left(\frac{\mathrm{TOL}}{\rho_{low}(\mathrm{TOL})}\right)^{1/2}\left(\frac{\rho_{up}(\mathrm{TOL})}{\rho_{low}(\mathrm{TOL})}\right)^{\varepsilon}\right)E\left[\sum_{n=0}^{N-1}(\Delta t_n)^2\right],
\end{aligned}
\tag{15}
$$

for any $\varepsilon > 0$ and where

$$
\begin{aligned}
\tilde{\rho}(t_n,\omega) \equiv \frac{1}{2}\Big(&\big(\partial_t a_k + \partial_j a_k a_j + \partial_{ij} a_k d_{ij}\big)\varphi_k(t_{n+1}) \\
&+\big(\partial_t d_{km} + \partial_j d_{km} a_j + \partial_{ij} d_{km} d_{ij} + 2\partial_j a_k d_{jm}\big)\varphi'_{km}(t_{n+1}) \\
&+\big(2\partial_j d_{km} d_{jr}\big)\varphi''_{kmr}(t_{n+1})\Big)
\end{aligned}
\tag{16}
$$

and the terms in the sum of (16) *are evaluated at the a posteriori known points* $(t_n, \overline{X}(t_n))$*, i.e.,*

$$
\partial_\alpha a \equiv \partial_\alpha a(t_n,\overline{X}(t_n)), \quad \partial_\alpha b \equiv \partial_\alpha b(t_n,\overline{X}(t_n)), \quad \partial_\alpha d \equiv \partial_\alpha d(t_n,\overline{X}(t_n)).
$$

Here $\varphi \in \mathbb{R}^d$ is the solution of the discrete dual backward problem

$$
\begin{aligned}
&\varphi_i(t_n) = \partial_i c_j(t_n,\overline{X}(t_n))\varphi_j(t_{n+1}), \quad t_n < T, \\
&\varphi_i(T) = \partial_i g(\overline{X}(T)),
\end{aligned}
\tag{17}
$$

with

$$
c_i(t_n,x) \equiv x_i + \Delta t_n a_i(t_n,x) + \Delta W_n^\ell b_i^\ell(t_n,x) \tag{18}
$$

and its first and second variation

$$
\varphi'_{ij} \equiv \partial_{x_j(t_n)}\varphi_i(t_n) \equiv \frac{\partial \varphi_i(t_n;\overline{X}(t_n)=x)}{\partial x_j}, \tag{19}
$$

$$
\varphi''_{ikm}(t_n) \equiv \partial_{x_m(t_n)}\varphi'_{ik}(t_n) \equiv \frac{\partial \varphi'_{ik}(t_n;\overline{X}(t_n)=x)}{\partial x_m}, \tag{20}
$$

which satisfy

$$
\begin{aligned}
\varphi'_{ik}(t_n) &= \partial_i c_j(t_n,\overline{X}(t_n))\partial_k c_p(t_n,\overline{X}(t_n))\varphi'_{jp}(t_{n+1}) \\
&\quad + \partial_{ik} c_j(t_n,\overline{X}(t_n))\varphi_j(t_{n+1}), \quad t_n < T, \\
\varphi'_{ik}(T) &= \partial_{ik} g(\overline{X}(T)),
\end{aligned}
\tag{21}
$$

and

$$\begin{aligned}\varphi''_{ikm}(t_n) &= \partial_i c_j(t_n,\overline{X}(t_n))\partial_k c_p(t_n,\overline{X}(t_n))\partial_m c_r(t_n,\overline{X}(t_n))\varphi''_{jpr}(t_{n+1})\\ &\quad+\partial_{im} c_j(t_n,\overline{X}(t_n))\partial_k c_p(t_n,\overline{X}(t_n))\varphi'_{jp}(t_{n+1})\\ &\quad+\partial_i c_j(t_n,\overline{X}(t_n))\partial_{km} c_p(t_n,\overline{X}(t_n))\varphi'_{jp}(t_{n+1})\\ &\quad+\partial_{ik} c_j(t_n,\overline{X}(t_n))\partial_m c_p(t_n,\overline{X}(t_n))\varphi'_{jp}(t_{n+1})\\ &\quad+\partial_{ikm} c_j(t_n,\overline{X}(t_n))\varphi_j(t_{n+1}), \quad t_n < T,\\ \varphi''_{ikm}(T) &= \partial_{ikm} g(\overline{X}(T)),\end{aligned} \tag{22}$$

respectively.

Observe that the constant in $\mathscr{O}$ that appears in (15) may not be uniform with respect to the value ε. Thus, in practice one chooses $\varepsilon = \varepsilon(\text{TOL})$ to minimize the contribution of the remainder term to the error expansion (15).

Let us now discuss how to modify the error density $\tilde{\rho}(t_n,\omega)$ in (16) to satisfy the bounds (10) and at the same time guarantee that $\Delta t_{\sup} \to 0$ as $\text{TOL} \to 0$, see Lemma 2.

Consider, for $t \in [t_n, t_{n+1})$ and $n = 1,\dots,N$, the piecewise constant function

$$\rho(t) \equiv \text{sign}(\tilde{\rho}(t_n)) \min\big(\max(|\tilde{\rho}(t_n)|, \rho_{low}(\text{TOL})), \rho_{max}(\text{TOL})\big), \tag{23}$$

where

$$\begin{aligned}&\rho_{low}(\text{TOL}) = \text{TOL}^{\bar{\gamma}}, \quad 0 < \bar{\gamma} < \tfrac{\alpha}{\alpha+2}, \;\; 0 < \alpha < \tfrac{1}{2},\\ &\rho_{max}(\text{TOL}) = \text{TOL}^{-r},\; r > 0,\end{aligned} \tag{24}$$

and with the standard notation for the function sign, that is $\text{sign}(x) = 1$ for $x \geq 0$ and -1 for $x < 0$. The function ρ defined by (23) measures the density of the time discretization error; it is used in (33) and (34) to guide the mesh refinements. From now on, with a slight abuse of notation, $\rho(t_n) = \rho_n$ denotes the modified density (23).

Following the error expansion in Theorem 1, the time discretization error is approximated by

$$|\mathscr{E}_T| = |E[g(X(T)) - g(\overline{X}(T))]| \lesssim E\left[\sum_{n=1}^{N} r(n)\right], \tag{25}$$

using the error indicator, $r(n)$, defined by

$$r(n) \equiv |\rho(t_n)|\Delta t_n^2, \tag{26}$$

with the modified error density defined by (23). According to Corollary 4.3 and Theorem 4.5 in [10], we have the almost sure convergence of the error density to a limit density denoted by $\hat{\rho}$, $\rho \to \hat{\rho}$ as $\text{TOL} \to 0$.

2 Adaptive Algorithms and Multilevel Variance Reduction

In this section we describe the multilevel Monte Carlo algorithm with adaptive stochastic time steps for approximating $E[g(X(T))]$.

Given a tolerance $\mathrm{TOL} > 0$ for which we want the estimate (6) to be fulfilled, we split the tolerance into a time discretization tolerance and a statistical error tolerance,

$$\mathrm{TOL} = \mathrm{TOL_T} + \mathrm{TOL_S}.$$

The optimal way of choosing $\mathrm{TOL_T}$ and $\mathrm{TOL_S}$ in terms of minimizing the computational work can be approximated by Lagrangian optimization. The basis of the error control is to choose the number of samples large enough to make the estimated statistical error smaller than $\mathrm{TOL_S}$ and adaptively refining the time steps, for each realization, until the estimated time discretization error is smaller than $\mathrm{TOL_T}$.

The stochastic time stepping algorithm uses criteria related to (9) with an outer and an inner loop, described below. Given the value of M_0, and mutually independent sample sets $\{\mathscr{S}_\ell\}_{\ell=0}^L$ where each $\mathscr{S}_\ell$ consists of

$$M_\ell = \left\lceil M_0 \frac{\rho_{low}(\mathrm{TOL}_0)\mathrm{TOL}_\ell}{\rho_{low}(\mathrm{TOL}_\ell)\mathrm{TOL}_0} \right\rceil \tag{27}$$

independent realisations of the underlying Wiener process, the *outer loop* uses a multilevel Monte Carlo technique to estimate $E[g(X(T))]$ and, if necessary, update the value M_0. Recall that the lower bound for the error density, ρ_{low}, was introduced in (24). We use the enforced deterministic lower bound

$$M_0 \geq M_{-1} = \mathrm{const} \cdot \mathrm{TOL}^{-1}. \tag{28}$$

The sample set independence makes it possible to estimate $E[g(X(T))]$ by the sum of sample averages

$$\mathscr{E}_{\{\mathscr{S}_\ell\}_{\ell=0}^L}\left(g(\overline{X}_L(T))\right) = \mathscr{A}_{\mathscr{S}_0}\left[g(\overline{X}_0(T))\right] + \sum_{\ell=1}^{L} \mathscr{A}_{\mathscr{S}_\ell}\left[g(\overline{X}_\ell(T)) - g(\overline{X}_{\ell-1}(T))\right],$$

$$\mathscr{A}_{\mathscr{S}_\ell}[f] := M_\ell^{-1} \sum_{\omega \in \mathscr{S}_\ell}^{M_\ell} f(\omega),$$

where the algorithm for constructing $g(\overline{X}_{\ell-1}(T))$ must be identical on levels ℓ and $\ell - 1$ for the telescoping sum to work perfectly; this is described in detail later in this section and explicitly in Algorithm 2.1. Approximate the variance of $\mathscr{E}_{\{\mathscr{S}_\ell\}_{\ell=0}^L}\left(g(\overline{X}_L(T))\right)$ by the sum of sample variances

$$\sigma^2 = \frac{\mathscr{V}_{\mathscr{S}_0}\left[g(\overline{X}_0(T))\right]}{M_0} + \sum_{\ell=1}^{L} \frac{\mathscr{V}_{\mathscr{S}_\ell}\left[g(\overline{X}_\ell(T)) - g(\overline{X}_{\ell-1}(T))\right]}{M_\ell} \tag{29}$$

and aim to control this variance by choosing M_0 sufficiently large so that

$$\sigma < \frac{TOL_S}{C_C}. \tag{30}$$

If $\sigma > \frac{TOL_S}{C_C}$, the number of samples M_0 is increased in the next batch; in the numerical examples of Sect. 3 the size of the new sample set was set to

$$\left\lceil M_{0,old} \max\left\{2\,,\, \min\left\{\sigma^2\left(C_\mathrm{C}/\mathrm{TOL_S}\right)^2\,,\, \mathrm{MCH}\right\}\right\}\right\rceil, \tag{31}$$

with $\mathrm{MCH} = 10$, but we may use the rule $M_{0,new} = 2M_{0,old}$ as well. The parameter *MCH* should not be taken too close to one in order to avoid a large number of iterations with similar M_0 before convergence, yielding a total computational work much larger than the computational work corresponding to the accepted M_0. On the other hand, *MCH* should not be too large in order to avoid using an excessively large M_0.

The *inner loop*, with iteration index ℓ representing a level in the adaptive mesh hierarchy, generates M_ℓ realization pairs[3], $(\overline{X}_{\ell-1}(T), \overline{X}_\ell(T))$, of (5) approximating (1) to the accuracy tolerances $\mathrm{TOL}_{\ell-1}$ and TOL_ℓ. These pairs are constructed by successive subdivision of an initial grid Δt_{-1}. First, the algorithm determines the grid $\Delta t_{\ell-1}$ from the initial grid Δt_{-1} by starting out with the tolerance $\mathrm{TOL}_0 = 2^L\mathrm{TOL_T}$ for the time discretization error and successively halving that tolerance until it becomes $\mathrm{TOL}_{\ell-1} = 2^{(L-\ell+1)}\mathrm{TOL_T}$ while for each new tolerance constructing the new grid by repeated adaptive subdivision of the previously constructed mesh. This iterative procedure in Algorithm 2.1, with index $\tilde{\ell} = 0, \ldots, \ell-1$, ensures that a grid $\Delta t_{\ell-1}$ on level ℓ is generated in the same way as a grid $\Delta t_{\ell-1}$ on level $\ell-1$ and consequently that $E[\overline{X}_\ell(T)]$ when computed as the coarser approximation in a pair $(\overline{X}_\ell(T), \overline{X}_{\ell+1}(T))$ is the same as when computed as the finer approximation in a pair $(\overline{X}_{\ell-1}(T), \overline{X}_\ell(T))$. The above mentioned property is necessary for the telescopic expansion of the time discretization error introduced by Giles in [3]. Second, the algorithm determines the grid Δt_ℓ by successively subdividing the recently determined $\Delta t_{\ell-1}$ according to the refinement criterion (34) until the stopping criterion (33) is satisfied.

Due to the stochastic nature of SDEs, each realization pair of $(\overline{X}_{\ell-1}(T), \overline{X}_\ell(T))$ may refine the initial grid Δt_{-1} differently. In particular, grids corresponding to different realizations on the same level ℓ may be different. To take this feature into account in the grid refinement, we introduce some notation. Let N_ℓ and $\overline{\mathscr{N}}_\ell$ denote the number of time steps and the approximate average number of time steps for realizations at level ℓ, respectively; see Algorithm 2.2 for details on the approximation technique and its update through the iteration. Further, denote the grid corresponding to one realization at level ℓ by

$$\Delta t_\ell = [\Delta t_\ell(0), \ldots, \Delta t_\ell(N_\ell - 1)]\,, \tag{32}$$

and its corresponding Wiener increments by

$$\Delta W_\ell = [\Delta W_\ell(0), \ldots, \Delta W_\ell(N_\ell - 1)]\,.$$

[3] Observe that for the level $\ell = 0$ only the realisation of $\overline{X}_0(T)$ is generated.

The refinement condition is based on the error indicator $r_{[\ell]}$, defined in (26), and uses similar refinements to those defined for the single level method. The stopping condition for refinement of Δt_ℓ is

$$\max_{1 \leq n \leq N_\ell} r_{[\ell]}(n) < C_{\mathrm{S}} \frac{\mathrm{TOL}_\ell}{\overline{\mathscr{N}}_\ell}. \tag{33}$$

When inequality (33) is violated, the n^{th} time step of Δt_ℓ is refined if

$$r_{[\ell]}(n) \geq C_{\mathrm{R}} \frac{\mathrm{TOL}_\ell}{\overline{\mathscr{N}}_\ell}. \tag{34}$$

Normally, the value for C_{R} would be around 2, and $C_{\mathrm{S}} > C_{\mathrm{R}}$ following the theory developed in [10, 9].

The inputs in Algorithm 2.1 are: $\mathrm{TOL_S}$, $\mathrm{TOL_T}$, initial number of sample realisations M_0, L, Δt_{-1}, initial guesses for the mean number of time steps $(\overline{\mathscr{N}}_\ell)_{\ell=0}^{L}$ needed for fulfillment of (33), and the three parameters C_{R}, C_{C}, and C_{S} used in the refinement condition (34) and stopping conditions (30) and (33), respectively. In this algorithm the mean number of initial time steps are chosen as $\overline{\mathscr{N}}_\ell = c\mathrm{TOL}_\ell^{-1}$, for $\ell = 0, \ldots, L$ and a small constant c.

Algorithm 2.1: Multilevel Monte Carlo with stochastic time stepping

```
Input  : $\mathrm{TOL_S}$, $\mathrm{TOL_T}$, $M_0$, $\Delta t_{-1}$, $\{\overline{\mathcal{N}}_\ell\}_{\ell=0}^L$, $C_\mathrm{R}$, $C_\mathrm{S}$, $C_\mathrm{C}$
Output: $\mu \approx E\left[g(X(T))\right]$

Set $k = 0$.
while $k < 1$ or (30) is violated do
    Compute $M_0$ new realizations of $g\left(\overline{X}_0(T)\right)$
    and their corresponding number of time steps, $\{N_0\}_1^{M_0}$,
    by generating Wiener increments $\Delta W_{-1}$ on the mesh $\Delta t_{-1}$ (independently
    for each realization) and calling Algorithm 2.3:
    ATSSE$(\Delta t_{-1}, \Delta W_{-1}, \mathrm{TOL_T} 2^L, \overline{\mathcal{N}}_0)$.
    Set $\mu = \mathcal{A}_{\mathcal{S}_0}\left[g\left(\overline{X}_0(T)\right)\right]$ and $\sigma^2 = \frac{\mathcal{V}_{\mathcal{S}_0}\left[g(\overline{X}_0(T))\right]}{M_0}$.
    Compute the average number of time steps $\mathcal{A}_{\mathcal{S}_0}[N_0]$.
    for $\ell = 1, \ldots, L$ do
        Set $M_\ell$ as in (27)
        Compute $M_\ell$ new realizations of $g\left(\overline{X}_{\ell-1}(T)\right)$,
        their corresponding number of time steps, $\{N_{\ell-1}\}_1^{M_\ell}$, and Wiener
        increments, $\Delta W_{\ell-1}$, by generating Wiener steps $\Delta W_{-1}$ on the mesh
        $\Delta t_{-1}$ (independently for each realization) and using the loop
        for $\hat{\ell} = 0, \ldots, \ell - 1$ do
            compute $\Delta t_{\hat{\ell}}$ and $\Delta W_{\hat{\ell}}$ by calling Algorithm 2.3:
            ATSSE$(\Delta t_{\hat{\ell}-1}, \Delta W_{\hat{\ell}-1}, \mathrm{TOL}_T 2^{L-\hat{\ell}}, \overline{\mathcal{N}}_{\hat{\ell}})$.
        end
        Compute the corresponding $M_\ell$ realizations of $g\left(\overline{X}_\ell(T)\right)$
        and their number of time steps, $N_\ell$, by calling Algorithm 2.3:
        ATSSE$(\Delta t_{\ell-1}, \Delta W_{\ell-1}, \mathrm{TOL}_T 2^{L-\ell}, \overline{\mathcal{N}}_\ell)$.
        Set $\mu = \mu + \mathcal{A}_{\mathcal{S}_\ell}\left[g\left(\overline{X}_\ell(T)\right) - g\left(\overline{X}_{\ell-1}(T)\right)\right]$ and
        $\sigma^2 = \sigma^2 + \frac{\mathcal{V}_{\mathcal{S}_\ell}\left[g\left(\overline{X}_\ell(T)\right) - g\left(\overline{X}_{\ell-1}(T)\right)\right]}{M_\ell}$.
        Compute the average number of time steps $\mathcal{A}_{\mathcal{S}_\ell}[N_{\ell-1}]$ and $\mathcal{A}_{\mathcal{S}_\ell}[N_\ell]$.
    end
    if $\sigma$ violates (30) then
        Update the number of samples by
        $\left\lceil M_0 \max\left\{2, \min\left\{\sigma^2 \left(C_\mathrm{C}/\mathrm{TOL_S}\right)^2, \mathrm{MCH}\right\}\right\}\right\rceil$.
        Update the values of $\{\overline{\mathcal{N}_\ell}\}_{\ell=0}^L$ by calling Algorithm 2.2:
        UMNT $(\{M_\ell\}_{\ell=0}^L, \{\mathcal{A}_{\mathcal{S}_\ell}[N_\ell]\}_{\ell=0}^L, \{\mathcal{A}_{\mathcal{S}_\ell}[N_{\ell-1}]\}_{\ell=1}^L)$.
    end
    Increase $k$ by 1.
end
```

Algorithm 2.2: Update for the mean number of time steps, **(UMNT)**

```
Input  : $\{M_\ell\}_{\ell=0}^{L}$, $\{\mathscr{A}_{\mathscr{S}_\ell}[N_\ell]\}_{\ell=0}^{L}$, $\{\mathscr{A}_{\mathscr{S}_\ell}[N_{\ell-1}]\}_{\ell=1}^{L}$
Output: $\{\overline{\mathscr{N}_\ell}\}_{\ell=0}^{L}$
for $\ell = 0, 1, \ldots, L$ do
    if $\ell < L$ then
        Set $\overline{\mathscr{N}}_\ell = \frac{M_\ell \mathscr{A}_{\mathscr{S}_\ell}[N_\ell] + M_{\ell+1} \mathscr{A}_{\mathscr{S}_{\ell+1}}[N_\ell]}{M_\ell + M_{\ell+1}}$.
    else
        Set $\overline{\mathscr{N}}_L = \mathscr{A}_{\mathscr{S}_L}[N_L]$.
    end
end
```

Algorithm 2.3: Adaptive Time Step Stochastic Euler **(ATSSE)**

```
Input  : $\Delta t_{in}, \Delta W_{in}$, TOL, $\overline{\mathscr{N}}_{in}$
Output: $\Delta t_{out}, \Delta W_{out}, N_{out}, g_{out}$
Set $m = 0$, $\Delta t_{[0]} = \Delta t_{in}$, $\Delta W_{[0]} = \Delta W_{in}$, $N_{[0]} =$ number of steps in $\Delta t_{in}$
while $m < 1$ or $(r_{[m-1]}$; TOL, $\overline{\mathscr{N}}_{in})$ violates (33) do
    Compute the Euler approximation $\overline{X}_{[m]}$ and the error indicators $r_{[m]}$ on
    $\Delta t_{[m]}$ with the known Wiener increments $\Delta W_{[m]}$.
    if $(r_{[m]}$; TOL, $\overline{\mathscr{N}}_{in})$ violates (33) then
        Refine the grid $\Delta t_{[m]}$ by
        forall the intervals $n = 1, 2, \ldots, N_{[m]}$ do
            if $r_{[m]}(n)$ satisfies (34) then
                divide the interval $n$ into two equal parts
            end
        end
        and store the refined grid in $\Delta t_{[m+1]}$.
        Compute $\Delta W_{[m+1]}$ from $\Delta W_{[m]}$ using Brownian bridges on $\Delta t_{[m+1]}$.
        Set $N_{[m+1]} =$ number of steps in $\Delta t_{[m+1]}$.
    end
    Increase $m$ by 1.
end
Set $\Delta t_{out} = \Delta t_{[m-1]}$, $\Delta W_{out} = \Delta W_{[m-1]}$, $N_{out} = N_{[m-1]}$, $g_{out} = g(\overline{X}_{[m-1]})$.
```

3 A Stopped Diffusion Example

This section presents numerical results from an implementation of the algorithm of Sect. 2. We apply the algorithm to a challenging problem where the computational work of multilevel Monte Carlo based on uniform meshes is larger than the optimal $\mathscr{O}\left((\mathrm{TOL}^{-1}\log(\mathrm{TOL}))^2\right)$, which is still attained by the adaptive multilevel Monte

Carlo algorithm. This motivates the use of stochastic time steps that are adaptively refined for each sample path.

The additional difficulty of the problem is that we now wish to compute approximations of an expected value

$$E[g(X(\tau),\tau)], \tag{35}$$

where $X(t)$ solves the SDE (1), but where the function $g : D \times [0,T] \to \mathbb{R}$ is evaluated at the first exit time

$$\tau := \inf\{t > 0 : (X(t),t) \notin D \times (0,T)\}$$

from a given open domain $D \times (0,T) \subset \mathbb{R}^d \times (0,T)$. This kind of stopped (or killed) diffusion problems arise for example in mathematical finance when pricing barrier options and for boundary value problems in physics.

The main difficulty in the approximation of the stopped diffusion on the boundary ∂D is that a continuous sample path may exit the given domain D even though a discrete approximate solution does not cross the boundary of D. Due to this hitting of the boundary the order of weak convergence of the Euler–Maruyama method is reduced from 1 to $1/2$, in terms of the step size of uniform meshes, and the order of strong convergence is less than $1/2$ so that the complexity estimate in Theorem 1 of [3] for uniform multilevel simulations can not be applied.

We combine the adaptive multilevel algorithm of Sect. 2 with an error estimate derived in [2] that takes into account also the hitting error. The hitting error is accounted for by an extra contribution to the error density in (23); this contribution can be expressed in terms of exit probabilities for individual time steps, conditioned on the computed path at the beginning and the end of the time steps, and of the change in the goal function, g, when evaluated at a possible exit point within the time step instead of the actually computed exit $(\overline{X}(\bar{\tau}),\bar{\tau})$. The full expression of the resulting error indicators is given in equation (50) of [2]. Since the differential $\partial_i g(\overline{X}(T),T)$ in the discrete dual backward problem (17) does not exist if T is replaced by $\bar{\tau} < T$ this initial value must be alternatively defined; this can be done using difference quotients with restarted computed trajectories as described, both for the discrete dual and for its first and second variations, in equations (20-25) of [2]. Note that for this modified error density the proof in [10] of almost sure convergence to a limit density does not apply.

The results in this section are on the accuracy and cost of the adaptive multilevel algorithm of Sect. 2, applied to (35)–(36), with the error estimate modified for the barrier problem.

For the numerical example we consider the stopped diffusion problem

$$\begin{aligned} dX(t) &= \frac{11}{36}X(t)\,dt + \frac{1}{6}X(t)\,dW(t), \qquad \text{for } t \in [0,2] \text{ and } X(t) \in (-\infty,2), \\ X(0) &= 1.6. \end{aligned} \tag{36}$$

For $g(x,t) = x^3 e^{-t}$ with $x \in \mathbb{R}$, this problem has the exact solution $E[g(X_\tau, \tau)] = u(X(0),0) = X(0)^3$, where the solution, u, of the Kolmogorov backward equation is $u(x,t) = x^3 e^{-t}$. We chose an example in one space dimension for simplicity, although it is only in high dimension that Monte Carlo methods are more efficient than deterministic finite difference or finite element methods to solve stopped diffusion problems. The comparison here between the standard Monte Carlo and the Multilevel Monte Carlo methods in the simple one dimensional example indicates that the Multilevel Monte Carlo method will also be more efficient in high dimensional stopped diffusion problems, where a Monte Carlo method is a good choice.

In the simulations the tolerance levels were chosen as $\mathrm{TOL_S} = \mathrm{TOL}/2$ and $\mathrm{TOL_T} = \mathrm{TOL}/4$. We used for the stopping and refinement constants the values $C_\mathrm{S} = 5$ and $C_\mathrm{R} = 2$. The computations were performed in `Matlab 7` using the built in pseudo random number generator `randn` for simulating sampling from the normal distribution.

In the numerical complexity results the cost is measured by counting the total number of time steps in all batches and on all levels. The complexity study in Fig. 1 is based on multiple simulations for each tolerance using different initial states in the pseudo random number generator, with more data on the large tolerances than on the smallest ones. A least squares fit of the model[4]

$$cost = c_1 \left(\log\left(\frac{\mathrm{TOL_0}}{\mathrm{TOL}}\right) \frac{1}{\mathrm{TOL}} \right)^{c_2} \tag{37}$$

in the $\log_2$-$\log_2$-scale of the graph using equal weights on all data points gives $c_2 = 1.9$ where the value 2 is predicted by theory. When the least squares fit is made on the mean cost for each tolerance the parameter in the cost model is $c_2 = 2.0$. The corresponding cost using the single level adaptive algorithm with just one data point per tolerance used grows faster than TOL^3 in this example.

In Fig. 2 the data on cost versus tolerance of Fig. 1 is shown together with the corresponding errors. The observed errors are scattered below the corresponding tolerances showing that the algorithm achieves the prescribed accuracy. It was already observed above that the multilevel version of the adaptive algorithm improves on the convergence of the single level version; this figure also shows that the error using a basic single level Monte Carlo method with uniform time steps for the stopped diffusion problem decreases only like $cost^{-0.26}$, which is worse than the convergence rate of the single level version of the adaptive algorithm.

We remark that we present the error versus cost results for the basic Monte Carlo algorithm in a way that slightly favours it over the adaptive methods. To explain this we note that the adaptive algorithms aim to balance the contributions to the total error made by the statistical and by time discretization errors; since the constant time step algorithm was implemented without time discretization error estimates this balancing could not be made in the computations. Instead, for each step size, the cost and error pair displayed in the graph was obtained indirectly by first over-

[4] The number of levels is $1 + L = 1 + \log_2\left(\frac{\mathrm{TOL_0}}{\mathrm{TOL_T}}\right) = \log_2\left(\frac{\mathrm{TOL_0}}{\mathrm{TOL}}\right)$.

killing the statistical error using a large number of samples and then by, knowing that the resulting error was dominated by the time discretization error, using the computed sample variance to get an estimate of the number of samples that would have been sufficient for obtaining a statistical error of the same size as the time discretization error. This procedure favours the constant time step method over the adaptive methods in that it gives an ideal constant factor in the cost, but the order of convergence is not affected. On the other hand the computational overhead in the implementation of the adaptive time stepping algorithm is significantly greater than in the naive Monte Carlo algorithm; again the order of convergence is not changed.

In conclusion the observed convergence of the adaptive multilevel Monte Carlo method applied to the barrier problem (36) is close to the predicted

$$cost = c_1 \left(\log\left(\frac{\mathrm{TOL}_0}{\mathrm{TOL}} \right) \frac{1}{\mathrm{TOL}} \right)^2 .$$

This shows an improved convergence compared to the single level version of the adaptive Monte Carlo algorithm where the cost grows approximately like TOL^{-3}, which in itself is a better order of weak convergence than the one obtained using a single level Monte Carlo method with constant time steps where the cost grows like $error^{-4}$.

References

1. Eric Cancès, Frédéric Legoll, and Gabriel Stoltz. Theoretical and numerical comparison of some sampling methods for molecular dynamics. *M2AN Math. Model. Numer. Anal.*, 41(2):351–389, 2007.
2. Anna Dzougoutov, Kyoung-Sook Moon, Erik von Schwerin, Anders Szepessy, and Raúl Tempone. Adaptive Monte Carlo algorithms for stopped diffusion. In *Multiscale methods in science and engineering*, volume 44 of *Lect. Notes Comput. Sci. Eng.*, pages 59–88. Springer, Berlin, 2005.
3. Michael B. Giles. Multilevel Monte Carlo path simulation. *Oper. Res.*, 56(3):607–617, 2008.
4. Mike Giles. Improved multilevel Monte Carlo convergence using the Milstein scheme. In *Monte Carlo and quasi-Monte Carlo methods 2006*, pages 343–358. Springer, Berlin, 2008.
5. Paul Glasserman. *Monte Carlo methods in financial engineering*, volume 53 of *Applications of Mathematics (New York)*. Springer-Verlag, New York, 2004. Stochastic Modelling and Applied Probability.
6. Elyes Jouini, Jaksa Cvitanić, and Marek Musiela, editors. *Option pricing, interest rates and risk management*. Handbooks in Mathematical Finance. Cambridge University Press, Cambridge, 2001.
7. Ioannis Karatzas and Steven E. Shreve. *Brownian motion and stochastic calculus*, volume 113 of *Graduate Texts in Mathematics*. Springer-Verlag, New York, second edition, 1991.
8. Andrew J. Majda, Ilya Timofeyev, and Eric Vanden Eijnden. A mathematical framework for stochastic climate models. *Comm. Pure Appl. Math.*, 54(8):891–974, 2001.
9. Kyoung-Sook Moon, Anders Szepessy, Raúl Tempone, and Georgios E. Zouraris. Convergence rates for adaptive weak approximation of stochastic differential equations. *Stoch. Anal. Appl.*, 23(3):511–558, 2005.
10. Kyoung-Sook Moon, Erik von Schwerin, Anders Szepessy, and Raúl Tempone. An adaptive algorithm for ordinary, stochastic and partial differential equations. In *Recent advances in*

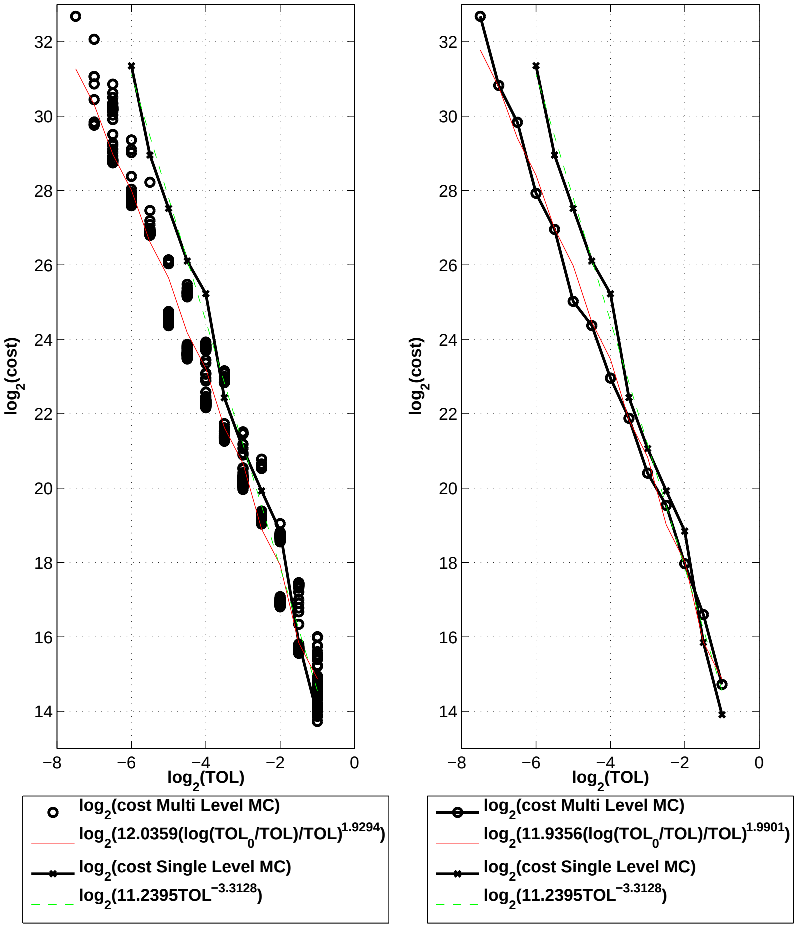

Fig. 1 Experimental complexity for the barrier example. To the left, the computational cost of the multilevel adaptive algorithm is shown for varying tolerances using different initial states in the pseudo random number algorithm. A least squares fit, in $\log_2 - \log_2$-scale, of the model $cost = c_1\left(\log\left(\mathrm{TOL}_0/\mathrm{TOL}\right)/\mathrm{TOL}\right)^{c_2}$ with equal weight on all observations results in $c_1 = 12$ and $c_2 = 1.9$. One realisation of the corresponding cost using a single level implementation of the same adaptive Monte Carlo method is included for reference. To the right is shown the mean computational cost over all observations where the values for large tolerances are based on more observations than those for small tolerances. When the least square fit is performed on the average values the resulting coefficients are $c_1 = 12$ and $c_2 = 2.0$.

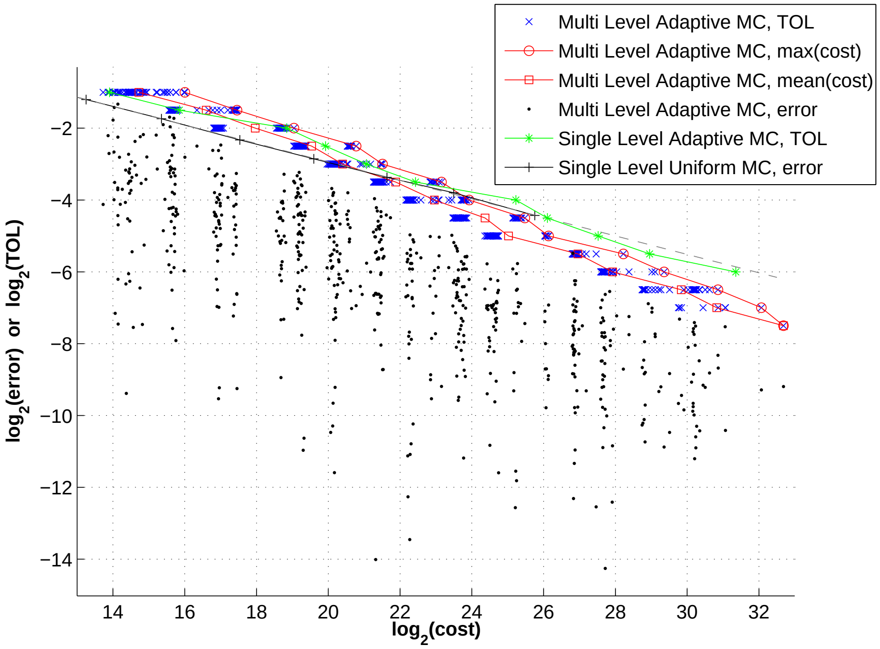

Fig. 2 The multilevel adaptive Monte Carlo algorithm with stochastic time steps has been tested on the barrier problem using a sequence of tolerances and different initial states in the pseudo random number generator. For each tolerance and each sample the computational cost is marked by an $\times$; the maximal cost and the average cost for a given tolerance have been chosen as representative measures. One realisation of the computational cost using a single level implementation of the adaptive algorithm for a sequence of tolerances is included as a reference, showing that the multilevel version is more efficient for small tolerances. A further comparison can be made with a basic single level, constant time step, Monte Carlo algorithm. This algorithm lacks error control; instead the statistical error was balanced against the time discretisation error in two steps: first the statistical error was over killed to reveal the time discretisation error for each time step size, and then the number of samples needed to make the statistical error the same size as the time discretisation error was estimated using variance estimates from the computation. This represents an ideal situation and it explains the very regular decay of the error with increasing cost seen in the graph; the least square fit, shown as a dashed line, has the slope -0.26, consistent with the $\mathscr{O}(\sqrt{\Delta t})$ time discretisation error for the barrier problem, and a $\mathscr{O}(1/\sqrt{N})$ statistical error.

adaptive computation, volume 383 of *Contemp. Math.*, pages 325–343. Amer. Math. Soc., Providence, RI, 2005.

11. Bernt Øksendal. *Stochastic differential equations*. Universitext. Springer-Verlag, Berlin, fifth edition, 1998. An introduction with applications.
12. Anders Szepessy, Raúl Tempone, and Georgios E. Zouraris. Adaptive weak approximation of stochastic differential equations. *Comm. Pure Appl. Math.*, 54(10):1169–1214, 2001.