Available online at www.sciencedirect.com
ScienceDirect
Computers and Mathematics with Applications 53 (2007) 1159–1180
An International Journal
computers & mathematics with applications
www.elsevier.com/locate/camwa

Space–time adaptive finite difference method for European multi-asset options

Per Lötstedt[a,*], Jonas Persson[a], Lina von Sydow[a], Johan Tysk[b]

[a] *Division of Scientific Computing, Department of Information Technology, Uppsala University, SE-75105 Uppsala, Sweden*
[b] *Department of Mathematics, Uppsala University, SE-75106 Uppsala, Sweden*

Received 7 July 2006; received in revised form 5 September 2006; accepted 14 September 2006

Abstract

The multi-dimensional Black–Scholes equation is solved numerically for a European call basket option using *a priori–a posteriori* error estimates. The equation is discretized by a finite difference method on a Cartesian grid. The grid is adjusted dynamically in space and time to satisfy a bound on the global error. The discretization errors in each time step are estimated and weighted by the solution of the adjoint problem. Bounds on the local errors and the adjoint solution are obtained by the maximum principle for parabolic equations. Comparisons are made with Monte Carlo and quasi-Monte Carlo methods in one dimension, and the performance of the method is illustrated by examples in one, two, and three dimensions.

Keywords: Black–Scholes equation; Finite difference method; Space adaptation; Time adaptation; Maximum principle

1. Introduction

We are interested in the numerical solution of the multi-dimensional Black–Scholes equation

$$\frac{\partial F}{\partial \hat{t}} + \sum_{i=1}^{d} \bar{r} s_i \frac{\partial F}{\partial s_i} + \frac{1}{2} \sum_{i,j=1}^{d} [\bar{\sigma}\bar{\sigma}^*]_{ij} s_i s_j \frac{\partial^2 F}{\partial s_i \partial s_j} - \bar{r} F = 0, \tag{1}$$

$$F(\hat{T}, s) = \Phi(s),$$

to determine the arbitrage free price F of a European option expiring at $\hat{T}$ with contract function $\Phi(s)$. Here, $\bar{r} \in \mathbb{R}_+ = \{x | x \geq 0\}$ is the short rate of interest and $\bar{\sigma} \in \mathbb{R}^{d\times d}$ is the volatility matrix. Our numerical method allows $\bar{r}$ and $\bar{\sigma}$ to be both level and time dependent but some of the theoretical estimates are restricted to time-independent interest and volatility.

* Corresponding author. Tel.: +46 18 471 2972; fax: +46 18 523049.
E-mail addresses: perl@it.uu.se (P. Lötstedt), jonasp@it.uu.se (J. Persson), lina@it.uu.se (L. von Sydow), Johan.Tysk@math.uu.se (J. Tysk).

0898-1221/$ - see front matter
doi:10.1016/j.camwa.2006.09.014

We will consider a European call basket option where the contract function is defined by

$$\Phi(s) = \left(\frac{1}{d} \sum_{i=1}^{d} s_i - K \right)^+, \tag{2}$$

where $(x)^+ = \max(x, 0)$ and K is the so-called strike price. Our method will work just as well for any contract function with vanishing second derivative across the boundary at $s_i = 0$. This way of determining the arbitrage free price was introduced by Black and Scholes in [1] and further developed by Merton in [2], both in 1973.

Another way to determine this price is to solve a stochastic differential equation with a Monte Carlo method and use the Feynman–Kač formula, see e.g. [3]. This method is well known to converge slowly in the statistical error. If we denote the number of simulations by M, the statistical error is proportional to $M^{-1/2}$. Better convergence rates are obtained with quasi-Monte Carlo methods [4,5]. In [6], an adaptive finite difference method is developed with full control of the local discretization error which is shown to be very efficient. The solution with finite difference approximations on a grid suffers from the "curse of dimensionality", with an exponential growth in dimension d of the number of grid points making it impractical to use in more dimensions than four (or so), and a Monte Carlo algorithm is then the best alternative. However, we believe that the finite difference method is a better method in low dimensions due to the uncertainty in the convergence of Monte Carlo and quasi-Monte Carlo methods. Furthermore, a finite difference solution is sufficiently smooth for a simple calculation of derivatives of the solution such as the hedging parameters $\Delta_i = \partial F/\partial s_i$, $\Gamma_i = \partial^2 F/\partial s_i^2$, $\theta = \partial F/\partial t$ ("the Greeks"). While finite difference methods are easily extended to the pricing of American options, this is not the case with Monte Carlo methods [5].

Finite difference methods for option pricing are found in the books [7,8] and in the papers [6,9–11]. A Fourier method is developed in [12] and an adaptive finite element discretization is devised in [13,14] for American options. Another technique to determine a smooth solution on a grid is to use the sparse grid method [15]. For a limited number of underlying assets, sparse grids have been applied to the pricing of options in [16]. The purpose of this paper is to develop an accurate algorithm suitable for European options based on finite difference approximations utilizing their regular error behavior to estimate and control the solution errors.

The partial differential equation (PDE) (1) is here discretized by second-order accurate finite difference stencils on a Cartesian grid. The time steps and the grid sizes are determined adaptively. Adaptive methods have the advantages of providing estimates of the numerical errors and savings in computing time and storage for the same accuracy in the solution. Moreover, there is no need for a user to initially specify a constant time step and a constant grid size for the whole solution domain.

Examples of algorithms for adaptivity in space and time are found in [6,17,18]. The grid and the time step may change at every discrete time point in [17]. In [6], a provisional solution is computed for estimation of the errors and then the fixed grid and the variable time step are chosen so that the local errors satisfy given tolerances in the final solution. The grid has a fixed number of points, but the points move in every time step for optimal distribution of them in moving grid methods; see e.g. [18]. In this paper, the time step varies in every step but the grid is changed only at certain time instants so that a maximal number of points are located optimally or a requirement on the error is fulfilled.

For the adaptive procedure, an error equation is derived for the global error $E(\hat{t}, s)$ in the solution. The driving right-hand side in this equation is the local discretization error. This error is estimated and the grid is adapted at selected time points so that the Cartesian structure of the grid is maintained and the time step is adjusted in every time step. The step sizes are chosen so that a linear functional of the solution error at $\hat{t} = 0$ satisfies an accuracy constraint ϵ

$$\left| \int g(s) E(0, s) \mathrm{d}s \right| \leq \epsilon \tag{3}$$

for a non-negative g chosen to be compactly supported where the accuracy of the solution is most relevant. The weights for the local error bounds in each time interval are solutions of the adjoint equation of (1). The growth of the error in the intervals between the grid adaptations is estimated *a priori* by the maximum principle for parabolic equations. In the same manner, the solution of the adjoint equation is bounded. Furthermore, our algorithm automatically chooses the discretization so that bounds on the errors of the type (3) above are satisfied also for multi-dimensional equations. The emphasis is on error control and reduction of the number of grid points. Efficiency and low CPU times are

also important, but these issues are very much dependent on the implementation and the computer system. The adaptation algorithm is first applied to a one-dimensional problem for comparison between the computed solution and the analytical solution. Two- and three-dimensional problems are then successfully solved with the adaptive algorithm.

An adaptive method for binomial and trinomial tree models on lattices for option pricing is found in [19]. The advantages of our method compared to that method are that there is no restriction on the variation of the spatial and temporal steps due to the method, the discretization errors are estimated and controlled, their propagation to the final solution is controlled, and the error there is below a tolerance given by the user.

The paper is organized as follows. We start by presenting a comparison between Monte Carlo methods, quasi-Monte Carlo methods and a finite difference method to motivate the development in the rest of the paper. Then, Eq. (1) is transformed by a change of variables and scaling in Section 3. The discretization in space and time is described in the following section. The adjoint equation and its relation to the discretization errors is the subject of Section 5. The adjoint solution is estimated with the maximum principle in Section 6. In Section 7, the local discretization errors are estimated and a simplification is derived based on the maximum principle. The algorithms for the space and time adaptivity are discussed in Sections 8 and 9. In Section 10, the adaptive algorithm is applied to the pricing of European call basket options with one, two, and three underlying assets. Conclusions are drawn in the final section.

2. Monte Carlo methods

In this section we are going to make a simple comparison in one dimension, $d = 1$, between a Monte Carlo (MC) method, a quasi-Monte Carlo (QMC) method and a finite difference (FD) method to determine the arbitrage free price of a European call option with one underlying asset. For a description of the Monte Carlo and quasi-Monte Carlo methods, see e.g. [5,20,21]. The finite difference method of second order is described in detail later in this paper.

Let M be the number of simulation paths. The error in the MC method decays as $M^{-1/2}$ independently of the dimension d and in an optimal QMC method as $M^{-1}(\log M)^d$ [4,5]. Time integration with an Euler method with a weak order of convergence of one introduces an error proportional to Δt, the length of the time step. The computational work grows linearly with M and is inversely proportional to Δt. Hence, the work W and error E fulfill

$$W_{\mathrm{MC}} = \mathcal{O}(M\Delta t^{-1}), \qquad E_{\mathrm{MC}} = \mathcal{O}(\Delta t) + \mathcal{O}(M^{-\gamma}), \tag{4}$$

where $\gamma = 1/2$ (MC) or $\gamma = 1$ (QMC, ignoring the logarithmic factor).

The grid size h in a finite difference method is of the order $N^{-1/d}$, where N is the total number of grid points. The error due to the space and time discretizations is proportional to h^2 and Δt^2 in a second-order method. Ideally, the work depends linearly on N in every time step. Thus, we have

$$W_{\mathrm{FD}} = \mathcal{O}(N\Delta t^{-1}), \qquad E_{\mathrm{FD}} = \mathcal{O}(\Delta t^2) + \mathcal{O}(N^{-2/d}). \tag{5}$$

Suppose that the error tolerance is ϵ for the spatial and temporal errors separately. Then it follows from (4) and (5) that

$$W_{\mathrm{MC}} \sim \epsilon^{-1-1/\gamma}, \qquad W_{\mathrm{FD}} \sim \epsilon^{-(d+1)/2}. \tag{6}$$

The work depends on a decreasing ϵ in the same way for a second-order FD method in five dimensions compared to an MC method and in three dimensions compared to a QMC method. With a smaller d, the preferred method is the FD scheme for ϵ sufficiently small. Otherwise, choose the stochastic algorithm.

A problem with constant volatility $\sigma = 0.3$ and strike price $K = 30$ is considered in the numerical experiments. For such problems, time integration in steps with MC and QMC is not necessary to solve a European option problem. We have considered pure MC, MC with antithetic variates (MC-anti) [5] and QMC with a Sobol sequence [5,22,23] generated by the code at [24]. The space and time steps in the FD method were such that the contribution to the error, obtained by comparison with the exact solution [3,25], was equal in space and time. The methods are implemented in a straightforward manner in Matlab without any attempt to optimize the codes.

In Fig. 1, the errors at $s = K$ for MC, MC-anti and QMC are displayed as a function of the number of simulation paths M. We also show the error as a function of computational time for MC, MC-anti and QMC as well as FD. The FD solution is available in an interval and the maximum error in the interval is recorded. The random numbers are

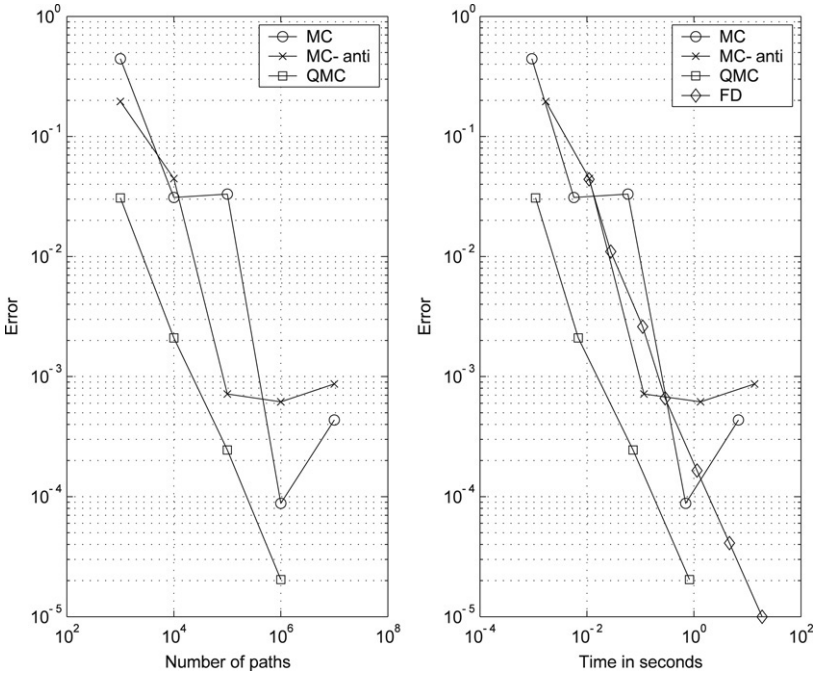

Fig. 1. To the left: Error as a function of simulation paths M. To the right: Error as a function of computational time.

generated separately. The CPU time to obtain 3.2×10^6 pseudo-random numbers was 0.78 s and for the same number of Sobol quasi-random numbers 51 631 s. The cost of calculating the QMC numbers is several orders of magnitude higher than only computing the solution with given random numbers.

From Fig. 1, the conclusion is that QMC is superior compared to MC and FD in this case. The slopes of QMC and FD are as expected from (6). Since W_{MC} is independent of Δt, we have $W_{\mathrm{QMC}} \sim \epsilon^{-1}$ and $W_{\mathrm{FD}} \sim \epsilon^{-1}$ in the figure and in (6). A least squares fit to the data for MC yields an exponent between -1 and -2 while (6) predicts -2.

Next, we are going to consider time stepping for MC and QMC. This is needed when we have to follow the simulation paths. This is the case for example for the constant elasticity of variance model [26]. Also, for other types of options, such as barrier options, it is necessary to resolve the simulation path in order to determine whether the barrier has been hit or not. In our comparisons, we have still used the standard Black–Scholes model with constant volatility in order to be able to accurately compute the error in the solution from the exact solution.

When using time stepping in QMC, each time step corresponds to one dimension. It is well known that QMC does not perform as well when multi-dimensional quasi-random sequences are needed. To enhance the performance of QMC applied to these problems, a so-called Brownian bridge construction is often used, see e.g. [27–29], henceforth referred to as QMC-BB.

In Figs. 2–4, the error is plotted as a function of the number of simulation paths M as well as a function of computational time. We have used 8, 16, and 32 time steps for the different MC and QMC methods in the figures. The time for computing the quasi-random numbers, the pseudo-random numbers, and the construction of the Brownian bridge is not included in the time measurements. Again, we have to bear in mind that the computation of the quasi-random numbers is expensive and is by far the predominant part of the computing time for QMC.

The error in the stochastic methods in the Figs. 2–4, with different Δt, has a more erratic behavior compared to the deterministic FD error. The FD error converges smoothly when the computational work increases. The error in the MC and QMC solutions decreases until the error in the time discretization dominates. This is best illustrated in Figs. 2 and 4 for QMC-BB where a plateau is reached for $M \geq 10^4$. The level of this plateau is about four times higher in Fig. 2 where Δt is four times longer. With more time steps and an improved resolution in time, FD eventually becomes the most efficient method.

In Fig. 5, M is increased in the MC and QMC computations when Δt is reduced using 8, 16, and 32 steps for the whole interval. The values of M are 625, 2500, and 10^4 for MC and 2500, 5000, and 10^4 for QMC to avoid an

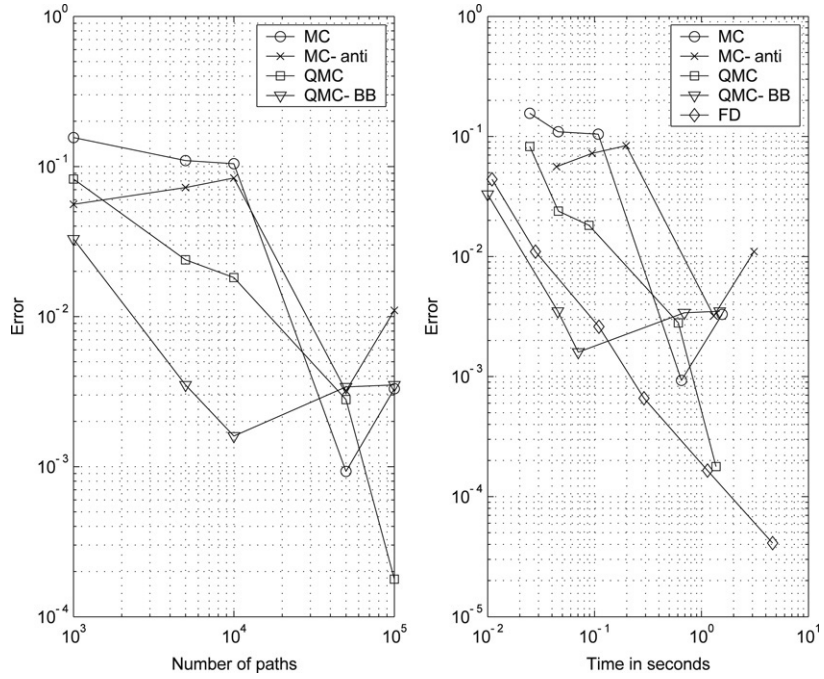

Fig. 2. MC, MC-anti, QMC, and QMC-BB use 8 time steps. To the left: Error as a function of simulation paths M. To the right: Error as a function of computational time.

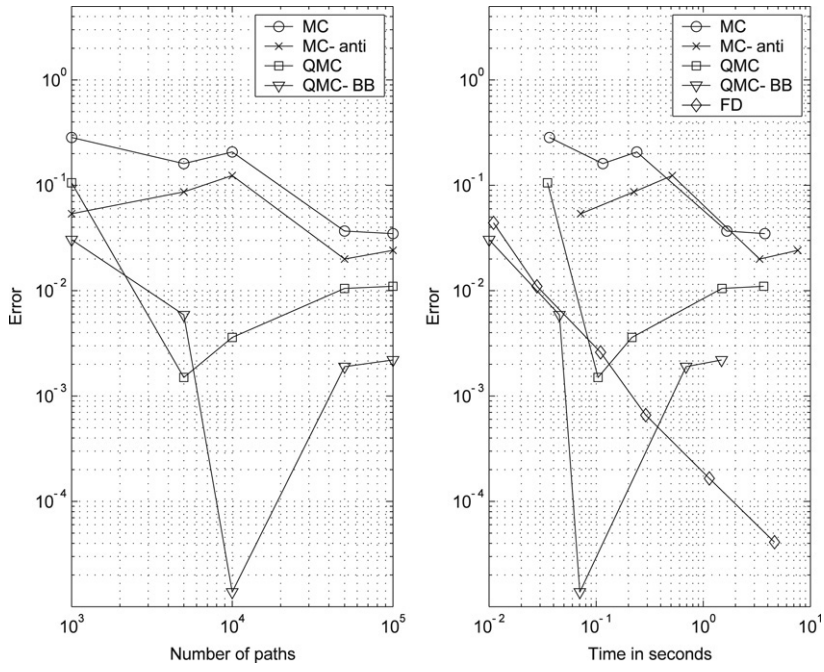

Fig. 3. MC, MC-anti, QMC, and QMC-BB use 16 time steps. To the left: Error as a function of simulation paths M. To the right: Error as a function of computational time.

imbalance between the errors in time and space. The errors in the QMC-BB and the FD methods decay regularly with a steeper slope for the FD algorithm. From the derivations in (6) we expect the exponents for QMC and FD to be -2 and -1, while in Fig. 5 they are -1.3 and -0.8.

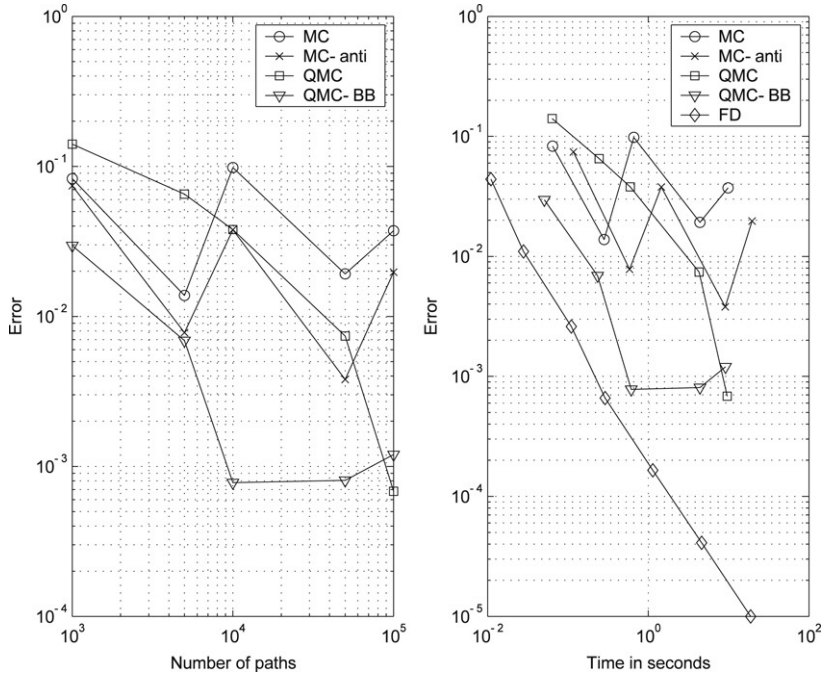

Fig. 4. MC, MC-anti, QMC, and QMC-BB use 32 time steps. To the left: Error as a function of simulation paths. To the right: Error as a function of computational time.

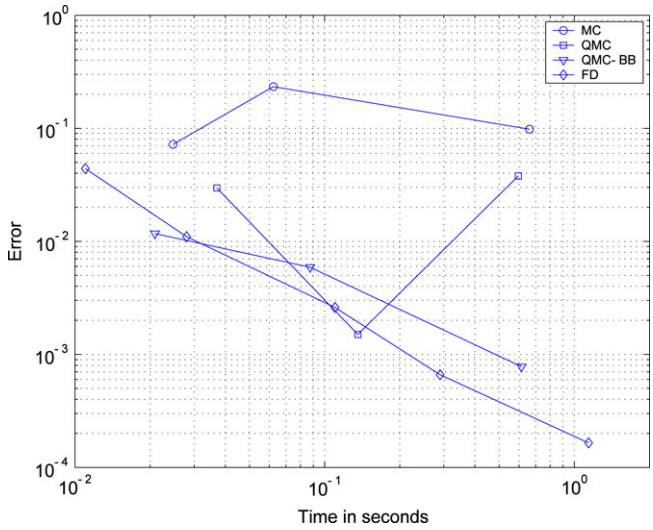

Fig. 5. MC, QMC, QMC-BB use 8, 16, and 32 time steps and M is increased to keep the balance between the errors in the stochastic methods in the same way as in the FD method.

The conclusion of this section is that FD is the preferred method in low dimensions when pricing options accurately and rapidly with error control and time stepping is needed. For most types of options, time integration is necessary in order to resolve the simulation paths and high accuracy is of utmost importance for compound options or for computing the Greeks. Furthermore, FD methods for European options can be extended to American options as in [30–33]. Efficient MC methods for low-dimensional American options are not known.

Error control is possible also for MC and QMC algorithms. Confidence intervals for the error due to the stochastic part of the MC method can be obtained by computing the variance of the solution for different realizations. This is

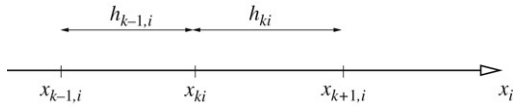

Fig. 6. The x_i-axis. Here, x_{ki}, $k = 1 \ldots n_i$, denotes the k:th node of dimension i.

more difficult for QMC methods [5]. The temporal error can also be estimated, e.g. by comparing the solutions using Δt and $\Delta t/2$ as in Richardson extrapolation [5].

3. Model problem

We start by transforming (1) from a final value problem to an initial value problem with dimensionless parameters. The transformation of the timescale has the advantage that standard texts on time integrators are applicable. The following transformations give the desired properties:

$$\begin{aligned} &Kx = s, \qquad r = \bar{r}/\hat{\sigma}^2, \qquad KP(t,x) = F(\hat{t},s), \\ &\sigma = \bar{\sigma}/\hat{\sigma}, \qquad t = \hat{\sigma}^2(\hat{T}-\hat{t}), \qquad K\Psi(x) = \Phi(s), \end{aligned} \tag{7}$$

where $\hat{\sigma}$ is a constant chosen as $\max_{ij}\sigma_{ij}$ in the solution domain. These transformations result in the following linear partial differential equation:

$$\begin{aligned} &P_t - r\sum_{i=1}^{d} x_i P_i - \sum_{i,j=1}^{d} a_{ij}x_i x_j P_{ij} + rP = 0, \\ &P(0,x) = \Psi(x) = \left(\frac{1}{d}\sum_{i=1}^{d} x_i - 1\right)^+, \end{aligned} \tag{8}$$

where $a_{ij} = \frac{1}{2}[\sigma\sigma^*]_{ij}$. The coordinates of $\mathbb{R}^d$ are called the spatial variables and are denoted by $x_1, \ldots, x_d$. The subscripts i, j, and later also k, l, m, on a dependent variable denote differentiation with respect to x_i and x_j, e.g. P_{ij}. Subscripts on an independent variable denote components of a vector such as x_i, or entries of a matrix such as a_{ij}. The matrix $[a_{ij}]$ is assumed to be positive definite. Thus, (8) is a parabolic equation. The subscript t denotes differentiation with respect to normalized time.

We will solve (8) in a cylinder

$$C = D \times [0,T], \tag{9}$$

where D is a bounded computational domain in $\mathbb{R}^d_+$ with boundary ∂D.

4. Discretization

Let $\mathcal{L}$ be the operator

$$\mathcal{L} = r\sum_{i=1}^{d} x_i\frac{\partial}{\partial x_i} + \sum_{i,j=1}^{d} a_{ij}x_i x_j\frac{\partial^2}{\partial x_i \partial x_j} - r. \tag{10}$$

The partial differential equation (8) can then be written as

$$P_t = \mathcal{L}P. \tag{11}$$

We introduce a semi-discretization of (11) in space by using centered second-order finite differences (FD) on a structured but non-equidistant grid; see Fig. 6.

The number of grid points in the i:th dimension is n_i, $i = 1, \ldots, d$. If we let P_h be a vector of the lexicographically ordered unknowns of length $\prod_{i=1}^{d} n_i$, then

$$\frac{\mathrm{d}P_h}{\mathrm{d}t} = A_h P_h, \tag{12}$$

where A_h is a matrix with the second-order finite difference discretization of $\mathcal{L}$. The matrix A_h in (12) is a very large, sparse matrix with the number of non-zeros of each row depending on the number of space dimensions, i.e. the number of underlying assets.

The first derivative in the i-direction is approximated as in [6,34] by

$$\frac{\partial P(x_{ki})}{\partial x_i} = P_i(x_{ki}) \approx a_{x_{ki}} P(x_{k+1,i}) + b_{x_{ki}} P(x_{ki}) + c_{x_{ki}} P(x_{k-1,i}), \tag{13}$$

where

$$a_{x_{ki}} = \frac{h_{k-1,i}}{h_{ki}(h_{ki} + h_{k-1,i})}, \qquad b_{x_{ki}} = \frac{h_{ki} - h_{k-1,i}}{h_{ki} h_{k-1,i}}, \qquad c_{x_{ki}} = -\frac{h_{ki}}{h_{k-1,i}(h_{k-1,i} + h_{ki})}.$$

and for the second derivative

$$\frac{\partial^2 P(x_{ki})}{\partial x_i^2} = P_{ii}(x_{ki}) \approx a_{x_{ki}x_{ki}} P(x_{k+1,i}) + b_{x_{ki}x_{ki}} P(x_{ki}) + c_{x_{ki}x_{ki}} P(x_{k-1,i}), \tag{14}$$

where

$$a_{x_{ki}x_{ki}} = \frac{2}{h_{ki}(h_{k-1,i} + h_{ki})}, \qquad b_{x_{ki}x_{ki}} = -\frac{2}{h_{k-1,i} h_{ki}}, \qquad c_{x_{ki}x_{ki}} = \frac{2}{h_{k-1,i}(h_{k-1,i} + h_{ki})}.$$

The cross-derivatives with respect to x_i and x_j are obtained by applying (13) once in the i-direction and once in the j-direction.

The leading terms in the discretization errors in (13) and (14) at x_{ki} are as in [34]:

$$\begin{aligned} \tau_{1i} &= -\frac{1}{6} h_{k-1,i} h_{ki} P_{iii}(x_{ki}) + \mathcal{O}(h_i^3), \\ \tau_{2i} &= -\frac{1}{3}(h_{ki} - h_{k-1,i}) P_{iii}(x_{ki}) - \frac{1}{12}(h_{ki}^2 - h_{ki} h_{k-1,i} + h_{k-1,i}^2) P_{iiii}(x_{ki}) + \mathcal{O}(h_i^3). \end{aligned} \tag{15}$$

For a smooth variation of the grid such that $h_{k-1,i} = h_{ki}(1 + \mathcal{O}(h_{ki}))$, the approximations (13) and (14) are both of second order.

There are several possible numerical boundary conditions that can be used for these problems. Here, the condition on a boundary where x_i is constant is that the numerical approximation of the second derivative P_{ii} is set to zero, which implies that the option price is nearly linear with respect to the spot price at the boundaries. These and other boundary conditions are discussed in [7].

For integration in time we use the backward differentiation formula of order two (BDF-2) [35], which is unconditionally stable for constant time steps. This method can be written as

$$\begin{aligned} &\alpha_0^n P_h^n = \Delta t^n \mathcal{L}(P_h^n) - \alpha_1^n P_h^{n-1} - \alpha_2^n P_h^{n-2}, \\ &\alpha_0^n = (1 + 2\theta^n)/(1 + \theta^n), \qquad \alpha_1^n = -(1 + \theta^n), \qquad \alpha_2^n = (\theta^n)^2/(1 + \theta^n), \end{aligned} \tag{16}$$

for variable time steps, where $\theta^n = \Delta t^n / \Delta t^{n-1}$, and $\Delta t^n = t^n - t^{n-1}$, see [17,35].

5. Discretization errors and the adjoint equation

Let $\tilde{P}$ denote a smooth reconstruction of the discrete data in P_h^n so that they agree at $t = t^n$ and at the grid points. The solution error $E = \tilde{P} - P$ approximately satisfies the following boundary value problem ("the error equation")

$$\begin{aligned} &E_t - r \sum_{i=1}^{d} x_i E_i - \sum_{i,j=1}^{d} a_{ij} x_i x_j E_{ij} + rE = E_t - \mathcal{L}E = \tau, \\ &E(0, x) = 0, \quad x \in D, \qquad E(t, x) = 0, \quad x \in \partial D, \end{aligned} \tag{17}$$

where τ is the local discretization or truncation error. By solving (17) we obtain the approximate global error $E_h^n = P_h^n - P$ at t^n at the grid points in $D \times [0, T]$.

The local discretization error consists of two parts, the temporal discretization error τ_k and the spatial discretization error τ_h:

$$\tau = \tau_k + \tau_h. \tag{18}$$

The aim is to develop a method that estimates τ *a posteriori* at t^n and then estimates the evolution of τ_h *a priori* for $t > t^n$. Then we determine computational grids to control τ_h and the time steps are selected to control τ_k in order to obtain a final solution fulfilling predescribed error conditions on a functional of the global solution error E. Such methods have been developed for finite element discretizations of different PDEs; see e.g. [36,37]. For this reason we introduce the adjoint equation to (17):

$$\begin{aligned} &u_t + \mathcal{L}^* u = 0, \\ &\mathcal{L}^* u = -r\sum_{i=1}^{d}(x_i u)_i + \sum_{i,j=1}^{d} a_{ij}(x_i x_j u)_{ij} - ru, \\ &u(T,x) = g(x). \end{aligned} \tag{19}$$

The boundary condition for the adjoint equation is $u = 0$ on ∂D. Note that the adjoint problem is a final value problem.

Using (17) and (19) we obtain

$$\begin{aligned} \int_0^T \int_D u\tau \mathrm{d}x\mathrm{d}t &= \int_0^T \int_D uE_t \mathrm{d}x\mathrm{d}t - \int_0^T \int_D u\mathcal{L}E \mathrm{d}x\mathrm{d}t \\ &= \int_D g(x)E(T,x)\mathrm{d}x - \int_0^T \int_D u_t E \mathrm{d}x\mathrm{d}t - \int_0^T \int_D (\mathcal{L}^* u)E \mathrm{d}x\mathrm{d}t \\ &= \int_D g(x)E(T,x)\mathrm{d}x. \end{aligned} \tag{20}$$

The function $g(x)$ should be chosen such that it is non-negative and has compact support in the domain where one is most interested in having an accurate solution. It is normalized such that

$$\int_D g(x)\mathrm{d}x = 1. \tag{21}$$

Partition the interval $[0, T]$ into L subintervals $\mathcal{I}_\ell = [t_\ell, t_{\ell+1})$ and take the absolute value of the left-hand side in (20) to arrive at

$$\begin{aligned} \left| \int_0^T \int_D u(t,x)\tau(t,x)\mathrm{d}x\mathrm{d}t \right| &\le \sum_{\ell=0}^{L-1} \left| \int_{t_\ell}^{t_{\ell+1}} \int_D u(t,x)\tau(t,x)\mathrm{d}x\mathrm{d}t \right| \\ &\le \sum_{\ell=0}^{L-1} \sup_{\substack{x\in D \\ t_\ell \le t \le t_{\ell+1}}} |\tau(t,x)| \int_{t_\ell}^{t_{\ell+1}} \int_D |u(t,x)|\mathrm{d}x\mathrm{d}t \\ &= \sum_{\ell=0}^{L-1} \|u\|_\ell \sup_{\substack{x\in D \\ t_\ell \le t \le t_{\ell+1}}} |\tau(t,x)|, \end{aligned} \tag{22}$$

with the definition

$$\|u\|_\ell = \int_{t_\ell}^{t_{\ell+1}} \int_D |u(t,x)|\mathrm{d}x\mathrm{d}t. \tag{23}$$

Our goal now is to generate a discretization of D and $[0, T]$ adaptively so that

$$\left| \int_D g(x)E(T,x)\mathrm{d}x \right| \le \epsilon, \tag{24}$$

where ϵ is a prescribed error tolerance. From (20) and (22), it is clear that we can bound the integral from above by estimating $\sup|\tau|$ and $\|u\|_\ell$.

The unknown u is the solution to the adjoint problem (19) and thus $\|u\|_\ell$ cannot be adjusted in order to fulfill (24). However, we are able to adjust the discretization error τ by controlling h and Δt in the spatial and temporal discretization. Thus, we will require in each interval $\mathcal{I}_\ell$ that

$$\sup_{\substack{x\in D\\ t_\ell\le t\le t_{\ell+1}}} |\tau(t,x)| \le \epsilon_\ell. \tag{25}$$

We choose to equidistribute the errors in the intervals, yielding

$$\epsilon_\ell = \epsilon/(L\|u\|_\ell). \tag{26}$$

Then from (22), (25) and (26) we find that

$$\left|\int_D g(x)E(T)\mathrm{d}x\right| \le \sum_{\ell=0}^{L-1}\|u\|_\ell \sup_{\substack{x\in D\\ t_\ell\le t\le t_{\ell+1}}} |\tau(t,x)| \le \epsilon. \tag{27}$$

To summarize this section, we have a strategy to obtain the prescribed tolerance ϵ in (24):

(i) Compute $\|u\|_\ell$, $\ell = 0,\dots,L-1$ in (23).
(ii) Compute ϵ_ℓ, $\ell = 0,\dots,L-1$ using (26).
(iii) Generate computational grids Γ_ℓ, $\ell = 0,\dots,L-1$, and choose time steps Δt^n for all n such that (25) is satisfied.

The time steps are adjusted in every step but the grids are changed only at L prespecified locations. The spatial error is estimated in the beginning of each interval with a constant grid and its growth in the interval is estimated (see Section 7). In this way, the expensive redistribution of the grid points and interpolation of the solution to a new grid are limited to $t = t_\ell$, $\ell = 0,1,\dots,L-1$. When passing from grid Γ_ℓ to $\Gamma_{\ell+1}$, the solution is transferred by cubic interpolation. In Section 6 we will estimate $\|u\|_\ell$ *a priori* and in Sections 7–9 we will demonstrate how to estimate τ *a priori* and *a posteriori* and derive new computational grids and vary the time step.

6. Maximum principle for the solution of the adjoint equation

A bound on the solution of the adjoint equation (19) is derived assuming constant r and a_{ij} using the maximum principle for parabolic equations; see [38]. Performing the differentiation in (19) and transforming the adjoint equation to an initial value problem by substituting $\tilde t = T - t$ yields

$$\begin{aligned} &u_{\tilde t} - \sum_{i,j=1}^{d}\left(2a_{ij}(1+\delta_{ij}) - \delta_{ij}r\right)x_j u_j - \sum_{i,j=1}^{d} a_{ij}x_i x_j u_{ij} - \left(\sum_{i,j=1}^{d} a_{ij}(1+\delta_{ij}) - (d+1)r\right)u = 0,\\ &u(\tilde t,x) = 0, \quad x\in\partial D, \qquad u(0,x) = g(x), \quad x\in D. \end{aligned} \tag{28}$$

The Kronecker delta function is denoted by δ_{ij}. We also have $t_\ell = \tilde t_{L-\ell}$, $\ell = 0,\dots,L-1$.

We introduce the standard notion of parabolic boundary of the cylinder,

$$\tilde C_\ell = D\times[\tilde t_\ell, \tilde t_{\ell+1}), \tag{29}$$

denoting it by $\partial\tilde C_\ell$ as the topological boundary of $\tilde C_\ell$ except $D\times\tilde t_\ell$. The standard maximum principle, see [38], says that in an equation of the type (28), in the absence of zero-order terms, the maximum and minimum of u over $\tilde C_\ell$ are attained on $\partial\tilde C_\ell$. In our case there is a zero-order term Ru where

$$R = \sum_{i,j=1}^{d} a_{ij}(1+\delta_{ij}) - (d+1)r.$$

However, the function $\mathrm{e}^{-R\tilde t}u$ satisfies (28) without zero-order terms. Thus, by the maximum principle

$$\inf_{\partial\tilde C_\ell} u\mathrm{e}^{-R\tilde t} \le u(\tilde t,x)\mathrm{e}^{-R\tilde t} \le \sup_{\partial\tilde C_\ell} u\mathrm{e}^{-R\tilde t}. \tag{30}$$

Using that $g \geq 0$ and the boundary condition on ∂D, the estimate

$$0 \leq u(\tilde{t}, x) \leq \mathrm{e}^{R\tilde{t}} \sup_{\partial \tilde{C}_\ell} u \mathrm{e}^{-R\tilde{t}} \leq \begin{cases} \mathrm{e}^{R(\tilde{t}-\tilde{t}_\ell)} \sup_{\partial \tilde{C}_\ell} u, & R \geq 0, \\ \mathrm{e}^{R(\tilde{t}-\tilde{t}_{\ell+1})} \sup_{\partial \tilde{C}_\ell} u, & R < 0, \end{cases} \tag{31}$$

holds for all $(\tilde{t}, x)$ in $\tilde{C}_\ell$.

Let $\Delta_\ell = t_{\ell+1} - t_\ell$. From the previous section we are interested in estimating

$$\sup_{\substack{x \in D \\ t_\ell \leq t \leq t_{\ell+1}}} |u(t,x)| = \sup_{\substack{x \in D \\ \tilde{t}_{L-\ell-1} \leq \tilde{t} \leq \tilde{t}_{L-\ell}}} |u(\tilde{t},x)| \leq \mathrm{e}^{|R|\Delta_\ell} \sup_{\partial \tilde{C}_\ell} |u(\tilde{t},x)|, \quad \ell = 0, \ldots, L-1. \tag{32}$$

Since $u(\tilde{t}, x) = 0$ on ∂D, $\sup_{\partial \tilde{C}_\ell} |u(\tilde{t}, x)|$ is reached at $\tilde{t} = \tilde{t}_\ell$. In particular, with the initial data $g(x)$,

$$\sup_{\substack{x \in D \\ t_\ell \leq t \leq t_{\ell+1}}} |u(t,x)| \leq \sup_{\substack{x \in D \\ 0 \leq t \leq t_L}} |u(t,x)| \leq \mathrm{e}^{|R|t_L} \sup_{x \in D} g(x). \tag{33}$$

Finally, by (33) we have a bound on $\|u\|_\ell$ in (23):

$$\|u\|_\ell \leq |D| \Delta_\ell \sup_{\substack{x \in D \\ t_\ell \leq t \leq t_{\ell+1}}} |u| \leq |D| \Delta_\ell \mathrm{e}^{|R|t_L} \sup_{x \in D} g(x). \tag{34}$$

From this upper bound, ϵ_ℓ, $\ell = 0, \ldots, L-1$, can be computed using (26). The adjoint solution is bounded by the given data and there is a non-vanishing lower bound on τ to satisfy the tolerance in (27).

These *a priori* estimates are in general not sufficiently sharp for the selection of ϵ_ℓ and an efficient adaptive procedure. Instead, (28) is solved numerically on a coarse grid in order to find $\|u\|_\ell$.

7. Estimating the spatial discretization error

The spatial error is estimated *a priori* in this section by applying the maximum principle to equations satisfied by terms in the discretization error. A simplifying assumption concerning the spatial error in the analysis here and in the implementation of the adaptive scheme is

Assumption 1. The dominant error terms in the approximations of the second derivatives in (8) are due to the diagonal terms $a_{ii} x_i^2 P_{ii}$.

The assumption is valid if $a_{ij} \ll a_{kk}$ for $i \neq j$ and all k, i.e. the correlations between the assets are small.

The following assumption is necessary for the analysis below to be valid. The adaptive procedure works well for an x-dependent interest rate and volatility but the *a priori* analysis is much more complicated.

Assumption 2. The interest rate r and the volatility matrix $[a_{ij}]$ are level (i.e. space) independent.

If Assumption 1 is valid, then the dominant terms in the discretization error in space of the operator (8) is

$$\tau_h = \sum_{k=1}^{d} \tau_{hk} = r \sum_{k=1}^{d} x_k \tau_{1k} + \sum_{k=1}^{d} a_{kk} x_k^2 \tau_{2k}, \tag{35}$$

where τ_{1k} is the error in the approximation of P_k and τ_{2k} is the error in P_{kk}.

Let $\partial_k h$ denote the derivative $\partial h / \partial x_k$. The grid is assumed to have a smooth variation such that

$$|\partial_k h| \leq ch, \tag{36}$$

for some constant c (cf. the discussion following (15)). With the centered difference schemes in Section 4 and the assumption (36), the leading terms in τ_{1k} and τ_{2k} in the step size h_k in the k-direction in (15) are

$$\tau_{1k} = -\frac{1}{6} h_k^2 P_{kkk} + O(h_k^3), \qquad \tau_{2k} = -\frac{1}{3} h_k \partial_k h P_{kkk} - \frac{1}{12} h_k^2 P_{kkkk} + O(h_k^3). \tag{37}$$

The derivatives of P satisfy parabolic equations similar to the equation for P (8). These equations are derived in the following lemma.

Lemma 1. *Let $G = \partial^K P/(\partial x_k)^K$ and let Assumption 2 be valid. Then G fulfills*

$$G_t = \sum_{i,j=1}^{d} a_{ij}x_i x_j G_{ij} + \sum_{j=1}^{d}(\alpha_K a_{jk}x_j + r)G_j + (\beta_K a_{kk} + r\gamma_K)G, \tag{38}$$

where $\alpha_K = 2K$, $\beta_K = \sum_{j=1}^{K-1}\alpha_j$, $\gamma_K = K - 1$.

Proof. The result follows from induction starting with (8) for $K = 0$. □

In order to estimate the error terms in each separate coordinate direction in (35) and (37), a parabolic equation is derived for fG, where f depends on only one coordinate.

Lemma 2. *Let $G = \partial^K P/(\partial x_k)^K$, $f = f(x_k)$, and let $[a_{ij}]$ be symmetric and let Assumption 2 be satisfied. Then*

$$\begin{aligned}(fG)_t = &\sum_{i,j=1}^{d} a_{ij}x_i x_j (fG)_{ij} + \sum_{j=1}^{d}((\alpha_K a_{jk} + r)x_j - 2a_{jk}x_j x_k(f_k/f))(fG)_j \\ &+ (\beta_K a_{kk} + r\gamma_K - a_{kk}x_k^2(f_{kk}/f) - (\alpha_K a_{kk} + r)x_k(f_k/f) + 2a_{kk}(x_k f_k/f)^2)(fG).\end{aligned} \tag{39}$$

Proof. Multiply (38) by f, replace fG_j and fG_{ij} by

$$\begin{aligned}fG_j &= (fG)_j - \delta_{jk}(f_k/f)(fG),\\ fG_{ij} &= (fG)_{ij} - \delta_{ik}\delta_{jk}(f_{kk}/f)(fG) + 2\delta_{ik}\delta_{jk}(f_k/f)^2(fG) - \delta_{ik}(f_k/f)(fG)_j - \delta_{jk}(f_k/f)(fG)_i,\end{aligned}$$

and we have the Eq. (39). □

We are now able to obtain a bound on the spatial discretization error in (35) by letting $f^1 = x_k h(x_k)^2$, $f^2 = x_k^2 h(x_k)\partial_k h(x_k)$, $f^3 = x_k^2 h(x_k)^2$, and $G = P_{kkk}$ and P_{kkkk} in Lemma 2.

Theorem 1. *Let $[a_{ij}]$ be symmetric and let Assumption 2 be satisfied. Then the spatial error τ_h in (35) in $C_\ell = D \times [t_\ell, t_{\ell+1})$ is bounded by*

$$\begin{aligned}|\tau_h| \le &\sum_{k=1}^{d}\frac{1}{6}r\exp(z_{1k}\Delta_\ell)\sup_{\partial C_\ell}x_k h^2(x_k)|P_{kkk}| + \sum_{k=1}^{d}\frac{1}{3}a_{kk}\exp(z_{2k}\Delta_\ell)\sup_{\partial C_\ell}x_k^2 h(x_k)\partial_k h(x_k)|P_{kkk}| \\ &+ \sum_{k=1}^{d}\frac{1}{12}a_{kk}\exp(z_{3k}\Delta_\ell)\sup_{\partial C_\ell}x_k^2 h^2(x_k)|P_{kkkk}|.\end{aligned} \tag{40}$$

The constants z_{pk} are the upper bounds

$$\begin{aligned}&\beta_K a_{kk} + \gamma_K r - (\alpha_K a_{kk} + r)x_k f_k^p/f^p + 2a_{kk}(x_k f_k^p/f^p)^2 - a_{kk}x_k^2 f_{kk}^p/f^p \le z_{pk},\\ &p = 1, 2, 3,\ K = 3, 3, 4.\end{aligned} \tag{41}$$

Proof. With $f = f^1 = x_k h^2(x_k)$ and $G = P_{kkk}$, the non-constant part of the leading term of τ_{1k} in (35) and (37) satisfies (39). By the maximum principle, see [38], applied to (39) using the same type of argument as in Section 6, we obtain

$$|x_k h^2(x_k)P_{kkk}| \le \exp(z_{1k}(t - t_\ell))\sup_{\partial C_\ell}x_k h^2(x_k)|P_{kkk}|.$$

Then the error due to the first derivatives is inferred from (35). The error τ_{2k} caused by the second derivatives is derived in the same manner. □

The upper bounds z_{pk} in the theorem depend on the smoothness of the step sizes. The factors depending on $f^p = x_k^q h_k^2$ with $(p, q) = (1, 1)$ and $(3, 2)$ in (41) are

$$\frac{x_k f_k^p}{f^p} = q + 2\frac{x_k \partial_k h}{h},$$
$$\frac{x_k^2 f_{kk}^p}{f^p} = q(q-1) + 4q\frac{x_k \partial_k h}{h} + 2\frac{x_k^2 \partial_k^2 h}{h} + 2\left(\frac{x_k \partial_k h}{h}\right)^2.$$

For f^2 we have

$$\frac{x_k f_k^2}{f} = 2 + \frac{x_k \partial_k h}{h} + \frac{x_k \partial_k^2 h}{\partial_k h},$$
$$\frac{x_k^2 f_{kk}^2}{f^2} = 2 + 4\left(\frac{x_k \partial_k h}{h} + \frac{x_k \partial_k^2 h}{\partial_k h}\right) + 3\frac{x_k^2 \partial_k^2 h}{h} + \frac{x_k^2 \partial_k^3 h}{\partial_k h}.$$

If the successive steps vary so that $h(x_k) = h_{0k} \exp(c x_k)$ for some constant c, then

$$\frac{\partial_k h}{h} = c, \qquad \frac{\partial_k^2 h}{h} = c^2, \qquad \frac{\partial_k^2 h}{\partial_k h} = c, \qquad \frac{\partial_k^3 h}{\partial_k h} = c^2,$$

(cf. the assumption in (36)) and with a small c, $x_k f_k^p / f^p$ and $x_k^2 f_{kk}^p / f^p$ are small in (41).

8. Space adaptivity

The computational domain D is a d-dimensional cube $[0, x_{\max}]^d$ covered by a Cartesian grid with the step sizes $h_{ij}, i = 1, \ldots, n_j, j = 1, \ldots, d$. The grid points, the outer boundary $x_{\max}$ and the step sizes are related by (cf. Fig. 6)

$$x_{ij} = x_{i-1,j} + h_{ij}, \quad i = 2, \ldots, n_j,$$
$$\sum_{i=1}^{n_j} h_{ij} = x_{\max}, \quad j = 1, \ldots, d.$$

Suppose that the time step Δt is constant in $[t_\ell, t_{\ell+1})$ and that the spatial step h_j is constant in the j:th dimension of D. If w_0 is the computational work per grid point and time step, then the total computational work in C_ℓ is

$$w = w_0 \frac{\Delta_\ell}{\Delta t} \prod_{j=1}^{d} \frac{x_{\max}}{h_j}. \tag{42}$$

The discretization error according to (25), (35) and (37) satisfies

$$|\tau| \le |\tau_k| + |\tau_h| \le |\tau_k| + \sum_{j=1}^{d} |\tau_{hj}| \le c_t \Delta t^2 + \sum_{j=1}^{d} c_j h_j^2 \le \epsilon_\ell, \tag{43}$$

for all t and x for some positive constants c_t and c_j in a second-order method. The step sizes Δt and h_j should be chosen such that w in (42) is minimized subject to the accuracy constraint (43). Since c_t and c_j are positive, the minimum of w is attained when the right part of (43) is satisfied as an equality. Then w is

$$w = w_0 \frac{\sqrt{c_t}\,\Delta_\ell}{\sqrt{\epsilon_\ell - \sum_{j=1}^{d} c_j h_j^2}} \prod_{j=1}^{d} \frac{x_{\max}}{h_j},$$

and a stationary point with respect to h_i is at

$$\frac{\partial w}{\partial h_i} = w\left(\frac{c_i h_i}{\epsilon_\ell - \sum_{j=1}^{d} c_j h_j^2} - \frac{1}{h_i}\right) = 0.$$

Hence,

$$c_i h_i^2 = \epsilon_\ell - \sum_{j=1}^{d} c_j h_j^2, \quad i = 1, \ldots, d,$$

with the solution

$$c_i h_i^2 = \epsilon_\ell/(d+1), \quad i = 1, \ldots, d. \tag{44}$$

The optimal bound on the time steps is obtained from (43) and (44)

$$c_t \Delta t^2 = \epsilon_\ell/(d+1). \tag{45}$$

Thus, it is optimal under these conditions to equidistribute the discretization errors in time and the dimensions. Ideally, w_0 is constant but for example the number of iterations in the iterative solver in each time step often depends on h_j and Δt in a complicated manner such that w_0 grows with decreasing h_j and decreases with smaller Δt.

As in [6], the spatial error τ_h is estimated *a posteriori* from the numerical solution by comparing the result of the fine grid space operator B_h with a coarse grid operator B_{2h} using every second grid point. Both B_h and B_{2h} approximate B to second order. Suppose that P_{hi} approximates the analytical solution $P(x)$ at x_i to second order in one dimension so that

$$P_{hi} = P(x_i) + c(x_i)h_i^2 + O(h_i^3),$$

where $c(x)$ is a smooth function and h_i has a slow variation. Then

$$\begin{aligned}(B_h P_h)_i &= (B_h P)(x_i) + (B_h c)(x_i)h_i^2 + O(h_i^3)\\ &= (BP)(x_i) + (Bc)(x_i)h_i^2 + \eta_{hi} + O(h_i^3),\\ (B_{2h} P_h)_i &= (B_{2h} P)(x_i) + (B_{2h} c)(x_i)h_i^2 + O(h_i^3)\\ &= (BP)(x_i) + (Bc)(x_i)h_i^2 + \eta_{2hi} + O(h_i^3).\end{aligned}$$

Subtract $B_h P_h$ from $B_{2h} P_h$ at every second grid point and use the second-order accuracy in the discretization error to obtain $\eta_{2hi} = 4\eta_{hi} + O(h_i^3)$ and

$$\eta_{hi} = \frac{1}{3}((B_{2h} P_h)_i - (B_h P_h)_i) + O(h_i^3). \tag{46}$$

The leading term in the spatial error is given by the first term in the right-hand side of (46).

The sequence h_{ij} in each dimension j is determined according to Theorem 1 and (40). Assuming that $c \ll 1$ in (36), the second term in the estimate in (40) is negligible and h_{ij} is chosen such that

$$\max_i h_{ij}^2 \left(\frac{1}{6} r \exp(z_{1j}\Delta_\ell) x_{ij} |P_{jjj}| + \frac{1}{12} a_{jj} \exp(z_{3j}\Delta_\ell) x_{ij}^2 |P_{jjjj}|\right) \le \epsilon_\ell/(d+1) \tag{47}$$

in each coordinate direction j where the maximum for i is taken over all the other dimensions. By changing the step size in each dimension separately, the Cartesian grid structure is maintained. The derivative P_{jjj} is estimated by computing η_{hi} in (46) with B_h being the centered difference approximation of the first derivative of P. Then $\eta_{hi} = h_i^2 P_{jjj}/6$. With B_h approximating the second derivative of P to second order, we have $\eta_{hi} = h_i^2 P_{jjjj}/12$.

The spatial error τ_h at t_ℓ is estimated as in (47) with the solution P^ℓ at t_ℓ and the step size sequences $h_{ij}, i = 1, \ldots, n_j, j = 1, \ldots, d$. The new sequence $\bar{h}_{ij}$ for $t > t_\ell$ is chosen locally at x_{ij} such that

$$\bar{h}_{ij} = h_{ij}\sqrt{\frac{\epsilon_\ell}{(d+1)(\epsilon_\ell \chi_h + |\tau_{hj}(x_{ij})|)}}. \tag{48}$$

Then the new error $\tau_{\bar{h}}$ is expected to be

$$|\tau_{\bar{h}j}(x_{ij})| = \bar{h}_{ij}^2 |\tau_{hj}(x_{ij})|/h_{ij}^2 = \epsilon_\ell/(d+1). \tag{49}$$

The small parameter χ_h in (49) ensures that h_{ij} is not too large when τ_h is very small. Since τ_h occasionally is non-smooth we apply a filter on these approximations of the local discretization errors to avoid an oscillatory sequence h_{ij}.

For multi-dimensional problems, the storage requirements may be the limiting factor and as an option the number of grid points can be restricted to a predefined level. The grid will be optimized for a small error within the limits of the available memory. By choosing a maximum number of grid points $N_{\max}$ in each direction j the method will still distribute the points so that $|\tau_{hj}(x_{ij})|$ is minimized. Suppose that the numerically computed discrete distribution of the grid points is $\bar{h}(x)$ determined by τ_h and that this distribution induces that $\bar{N}$ grid points are used. The new distribution will then place the grid points according to the scaled function

$$h_{\text{new}} = \frac{\bar{N}}{N_{\max}}\bar{h}(x). \tag{50}$$

In several dimensions this simple technique can reduce the number of grid points in each interval so that larger problems can be solved, but it can also be used to ensure that not too many points are used in the first interval. Experiments have shown that limiting the number of grid points, especially in the first interval, does not destroy the end-time accuracy in (24).

9. Time adaptivity

The discretization error in space is estimated by comparing a fine grid operator with a coarse grid operator. For the adaption of the time steps we compare an explicit predictor and an implicit corrector (BDF-2), both of second-order accuracy, to find an approximation of the local error in BDF-2 in the same way as in [17]. The predictor is the explicit method

$$\begin{aligned} &\tilde{\alpha}_0^n \tilde{P}^n = \Delta t^n \mathcal{L}(P^{n-1}) - \tilde{\alpha}_1^n P^{n-1} - \tilde{\alpha}_2^n P^{n-2}, \\ &\tilde{\alpha}_0^n = 1/(1+\theta^n), \qquad \tilde{\alpha}_1^n = \theta^n - 1, \qquad \tilde{\alpha}_2^n = -(\theta^n)^2/(1+\theta^n), \end{aligned} \tag{51}$$

with the local discretization error

$$\begin{aligned} &P(t^n) - \tilde{P}^n = C_p(\theta^n)(\Delta t^n)^3 P_{ttt} + O(\Delta t^4), \\ &C_p(\theta^n) = (1 + 1/\theta^n)/6, \end{aligned} \tag{52}$$

and θ^n defined by $\theta^n = \Delta t^n/\Delta t^{n-1}$ as in (16). The solution at t^n is determined by the implicit method BDF-2 defined in (16) with the predicted value $\tilde{P}^n$ from (51) as initial guess in an iterative solver. The local error of BDF-2 is

$$\begin{aligned} &P(t^n) - P^n = C_i(\theta^n)(\Delta t^n)^3 P_{ttt} + O(\Delta t^4), \\ &C_i(\theta^n) = -(1+\theta^n)^2/(6\theta^n(1+2\theta^n)). \end{aligned} \tag{53}$$

The integration is initialized at $t = 0$ with the Euler backward method with $\alpha_0^1 = 1, \alpha_1^1 = -1$, and $\alpha_2 = 0$ in (16).

The leading term $C_i(\theta^n)(\Delta t^n)^3 P_{ttt}$ in the local error in time in (53) is estimated by computing the difference between the numerical solution P^n in (16) and $\tilde{P}^n$ in (51):

$$\tau_k(t^n) = -\alpha_0^n C_i(\Delta t^n)^2 P_{ttt} \approx \alpha_0 C_i(P^n - \tilde{P}^n)/(\Delta t^n(C_i - C_p)). \tag{54}$$

The maximum $|\tau_k|$ of the estimate $\tau_k(t^n)$ in (54) over all grid points in D is compared to the accuracy requirement $\epsilon_\ell/(d+1)$ by computing

$$\zeta^n = \sqrt{\frac{\epsilon_\ell}{(d+1)(\epsilon_\ell \chi_k + |\tau_k|)}}, \tag{55}$$

where χ_k is a small parameter to avoid large time steps when τ_k is small (cf. (48)). If ζ^n is too large, then the time step is rejected and P^n is recomputed with a smaller Δt. Otherwise, P^n is accepted and a new Δt^{n+1} is determined. If $0.8 \le \zeta^n \le 1.15$, then we accept the time step and let $\Delta t^{n+1} = \Delta t^n$. If $\zeta^n < 0.8$, then the time step to t^n is rejected and P^n is recomputed with $\Delta t^n := 0.9\zeta^n \Delta t^n$. If $\zeta^n > 1.15$, then the step is accepted and the next time step is increased to $\Delta t^{n+1} = \min(0.9\zeta^n \Delta t^n, 2\Delta t^n)$ with the upper bound $2\Delta t^n$ introduced to avoid instabilities.

Since BDF-2 is an implicit method in time, we must solve large, linear, sparse systems of equations in each time step. These systems are solved with the `GMRES` method [39]. The `GMRES` iterations are terminated when the relative residual norm is sufficiently small. To be efficient and memory lean, the iterative method is restarted after six iterations. The system of equations is preconditioned by the incomplete LU factorization [40] with zero fill-in. The same factorization computed in the first time step is applied in all time steps after t_ℓ in each interval.

10. Numerical results

The transformed Black–Scholes equation (8) is solved in one, two, and three space dimensions with our adaptive method. Several different tests have been performed examining the method and its performance. Our method is compared to the standard method with a uniform grid in space and adaptivity in time and we also study how the memory can be used efficiently by restricting the number of grid points.

Since the precision of the estimates of the derivatives was investigated in [6] we mainly focus on the estimates of the linear functional (24) in this paper. In one space dimension the true numerical error can be calculated so that the functional (24) can be determined. In higher dimensions this is not possible. However, in all tests the upper bound (22) of the leftmost integral in (20) is computed. This estimate will be denoted by Υ^ϵ, and the adaptive process controls this value.

As a standard setup we have used the following parameters: the local mean rate of return r has been set to 0.05 and the volatility matrix σ has the value 0.3 on the diagonal and 0.05 in the sub- and super-diagonals. All other entries are zero. In the examples that follow, the volatility matrix is neither level nor time dependent but it could be chosen to be so without causing any difficulty in the adaptive method. In all computations we solve the transformed PDE (8) in forward time from 0 to $T = 0.1$. The computational domain D is a d-dimensional cube truncated at $x_{\max} = 4dK$ in every dimension, using a generalization of the common 'rule of thumb'. The reason for multiplying by d is to have the far-field boundary at four times the location of the discontinuity of the derivative of the initial function $\Phi(s)$ in each dimension. The location of the outer boundary is not critical for the efficiency of the method. Few grid points are placed there by the adaptive scheme.

10.1. Estimating the functional

To evaluate the method, the functional (24) is estimated in numerical experiments. In one space dimension, the exact solution for the European call option is found in [25,3] and is used to calculate the true error $E(x,T)$. The product $g(x)E(x,T)$ is integrated numerically with the second-order trapezoidal method. The integral is denoted by $\int_D^* g(x)E(x,T)\mathrm{d}x$.

The estimate Υ^ϵ defined by

$$\Upsilon^\epsilon = \sum_{\ell=0}^{L-1} \|u\|_\ell \sup_{\substack{x\in D \\ t_\ell \le t \le t_{\ell+1}}}{}^{*} |\tau(t,x)| \tag{56}$$

has been used in one and multiple space dimensions. This is the most interesting quantity since it is used to generate the grids in space and to select the time steps; see Section 5. The supremum of τ in (56) is denoted by a $*$ since it is not truly the supremum but has been estimated as follows.

Table 1
The estimate Υ^ϵ, the error functional in (24) and the number of grid points used in each interval for two different tolerances

ϵ	Υ^ϵ	$\left\lvert\int_D^* gE\mathrm{d}x\right\rvert$	# Grid points
10^{-3}	0.001088	0.000083	[81 61 49 45 41 37 33 53]
10^{-4}	0.000152	0.000009	[233 173 137 121 109 101 93 157]

Table 2
The bound on the number of points, the upper bound on the error functional, and number of grid points in the eight intervals

$N_{\max}$	Υ^ϵ	# Grid points
–	0.001088	[81 61 49 45 41 37 33 53]
65	0.001161	[65 61 49 45 41 37 33 53]
57	0.001246	[57 53 49 45 41 37 33 53]

The adjoint solution (19) is computed on a coarse equidistant grid with only a few time steps. Then $\|u\|_\ell$ in (23) is computed numerically. Theoretically the supremum of τ_h should be measured on the parabolic cylinder C_ℓ, see Theorem 1, but the errors are small on ∂D and we measure only τ on D after a few time steps from the start t_ℓ of each interval. The reason is that, when interpolating the solution from one grid to the next, additional errors are introduced, making the estimates of τ_h at t_ℓ unreliable. The initial condition is not sufficiently smooth for the adaptive procedure to work properly. Hence, in the first interval, we measure τ towards the end of the interval instead since the approximations of the derivatives P_{kkk} and P_{kkkk} blow up close to $t = 0$ and the algorithm would then use an excessive amount of grid points and very small time steps in the vicinity of $t = 0$. In Section 10.3 we show that the method actually can produce good results even with a restricted number of points in the first interval.

The *a priori* spatial error estimate in Theorem 1 contains the two factors $\exp(z_{1k}\Delta_\ell)$ and $\exp(z_{3k}\Delta_\ell)$. These coefficients in front of the third and fourth derivatives of P are typically of the size 1–3, indicating that the local discretization errors can grow that much in each interval. However, all our results show that these are really overestimates of the growth. The discretization errors do not increase with time in the intervals. On the contrary, they decay. This implies that Υ^ϵ will be overestimated in each interval.

The d-dimensional function $g(x)$ has been chosen as the product of Gaussian functions

$$g(x) = \kappa \prod_{i=1}^{d} \exp(-5(x_i - 1)^2), \tag{57}$$

scaled by κ to satisfy (21).

10.2. A one-dimensional numerical example

In the first one-dimensional example we have studied two different levels of ϵ. The estimate Υ^ϵ is compared with the numerically integrated (24) and the desired tolerance level ϵ for $L = 8$. The results are presented in Table 1.

We see that the algorithm produces a solution with a bound on the error close to the desired tolerance. As expected, the estimate Υ^ϵ is larger than $\int_D^* gE\mathrm{d}x$. A sharper estimate is obtained by increasing the number of intervals implying more frequent changes of the grid. We seek a balance between accurate estimates and many regridding operations (as in moving grid methods [18]) and coarser estimates with fewer changes of the grid (as we prefer here).

10.3. Restricting the number of grid points

An upper bound on the number of grid points is introduced in this one-dimensional example. Either this bound or the error tolerance determines the number of points. The distribution of points still depends on the spatial error estimate; see Section 8 and (50).

The limit has been set to unlimited, 65 and 57 grid points in Table 2 and $\epsilon = 0.001$. By restricting the number of grid points we can still achieve quite accurate results. The method sometimes has to add a few extra points (maximum of 4) since the number of points n_j must satisfy $n_j \bmod (4) = 1$ to be suitable for the error estimates.

Table 3
Estimates of the functionals are compared for an adaptive grid and two uniform grids

	Υ^ϵ	$\int_D^* g E \mathrm{d}x$	# Grid points
Adap. grid	0.001088	−0.000083	[81 61 49 45 41 37 33 53]
Equi. grid	0.002229	−0.000178	[81]
Equi. grid	0.001086	−0.000075	[121]

Table 4
The error tolerances, the estimate of the functional (24), and the number of grid points in two dimensions

ϵ	Υ^ϵ	# Grid points
10^{-2}	0.007710	$[61^2\ 45^2\ 33^2\ 29^2\ 29^2\ 25^2\ 25^2\ 29^2]$
10^{-3}	0.001417	$[193^2\ 113^2\ 81^2\ 65^2\ 57^2\ 53^2\ 45^2\ 73^2]$

Table 5
Estimates of the functional for two uniform grids and the adaptive grid

	Υ^ϵ	# Grid points
Adap. grid	0.007710	$[61^2\ 45^2\ 33^2\ 29^2\ 29^2\ 25^2\ 25^2\ 29^2]$
Equi. grid	0.014202	$[61^2]$
Equi. grid	0.007874	$[81^2]$

10.4. Comparison with uniform grids in one dimension

A solution on an equidistant grid in space is compared to a solution with our adaptive method in Table 3. The maximal number of grid points used by the adaptive algorithm with tolerance 0.001 is distributed equidistantly.

The results show that by redistributing the grid points adaptively, the error functional can be reduced significantly with fewer points. Counting the total number of points in the intervals, more than twice as many points in one dimension are needed to reduce the error with an equidistant grid to the same level as the adapted grid (400 vs. 968). The price is more administration for the adaptivity but this overhead cost drops quickly with increasing number of dimensions.

10.5. Two-dimensional numerical example

In the first two-dimensional example, two tolerance levels, $\epsilon = 0.01$ and 0.001, are tested. In this case, an exact solution is not available. Therefore, only the estimate Υ^ϵ is presented together with the number of grid points used in each dimension in Table 4.

As in the one-dimensional case in Table 1 we find that our method produces a result that almost fulfills the desired accuracy.

10.6. A second two-dimensional example

The one-dimensional numerical example from Section 10.4 is repeated here in two dimensions. The result on an adapted grid with $\epsilon = 0.001$ is compared to the results on two equidistant grids in space in Table 5. The same number of points in space is used in one uniform grid as the largest number in an interval of the adapted grid. The other uniform grid is chosen so that Υ^ϵ is approximately the same.

From the table we observe that the equidistant grid results in a lower bound on the error even though 61^2 grid points were used in all time steps. The equidistant grid uses 81^2 grid points to achieve the same level of accuracy as our adaptive method. However, as remarked in Section 10.4, the adaptive method introduces a certain overhead and the computation time is sometimes longer.

The variation of h along a coordinate is plotted in Fig. 7 in three consecutive intervals $[t_\ell, t_{\ell+1})$. The maximum h permitted in the algorithm is 0.5. The initial singularity in the solution influences the choice of step size in the first interval.

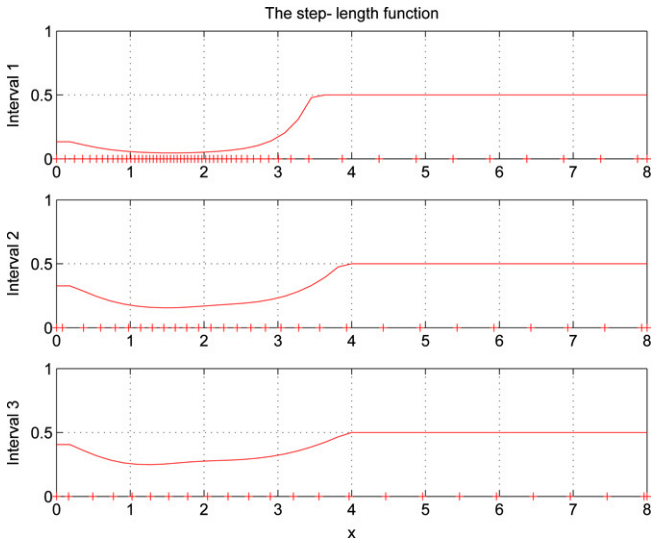

Fig. 7. The space steps and the grid points in three different time intervals in one coordinate direction.

Table 6
The estimate of the functional (24) Υ^ϵ with two adaptive grids, an equidistant grid and an adaptive grid with a maximal number of grid points

	ϵ	Υ^ϵ	# Grid points
Adap. grid	0.1	0.055996	$[41^3\ 29^3\ 29^3\ 25^3\ 29^3\ 25^3\ 29^3\ 29^2 \times 25]$
Equi. grid	–	0.082717	$[41^3]$
Adap. grid-$N_{\max}$	0.1	0.105518	$[29^2 \times 25\ 25^3\ 25^3\ 25^3\ 25^3\ 25^3\ 25^3\ 25^3]$
Adap. grid-$N_{\max}$	0.1	0.084784	$[33^3\ 29^3\ 29^3\ 29^3\ 29^3\ 29^3\ 29^3\ 29^3]$
Adap. grid	0.05	0.039529	$[53^3\ 37^3\ 29^3\ 29^3\ 29^3\ 29^3\ 29^3\ 29^3]$

10.7. A three-dimensional example

In this three-dimensional example we combine two of the experiments in the previous examples. First we solve with our adaptive method and the error tolerance $\epsilon = 0.1$. Then we solve with an equidistant grid with the same number of grid points as the maximal number used by the adaptive method. In the next two experiments, the number of grid points is restricted as in (50) ($N_{\max} = 29$ and $N_{\max} = 33$). Finally, the solution is computed with a halved ϵ. The results are displayed in Table 6. The conclusion is also here that adaptive distribution achieves a lower error bound for the same number of points compared to a uniform distribution or the same error with fewer points. As an example, the CPU time for the second case is about three times longer than for the adaptive, third case.

10.8. Time-stepping and iterations

The time steps are selected at every t^n following Section 9 such that the estimated τ_k satisfies $\max_D |\tau_k| \leq \epsilon_\ell/(d+1)$.

The time history of the time steps in the one-dimensional example with $\epsilon = 0.0001$ is displayed in Fig. 8. The vertical lines indicate the interval boundaries t_ℓ where a new grid is determined. At t_ℓ the estimate of the time discretization error is not always reliable and three steps with a constant Δt are taken there. The time step increases rapidly after $t = 0$ where higher derivatives of P are large due to the discontinuous initial data in (8).

The two-dimensional problem is solved in four intervals with $\epsilon = 0.005$. The variation of $|\tau_k|$ in the intervals is smooth in Fig. 9. The error tolerance $\epsilon_\ell/(d+1)$ is not satisfied in the first steps after $t = 0$ where the integration is advanced with a minimal time step $\Delta t_{\min}$. The number of GMRES iterations in each time step is found in Fig. 10. It is about 10 in the whole interval with a small increase at the end when Δt is longer.

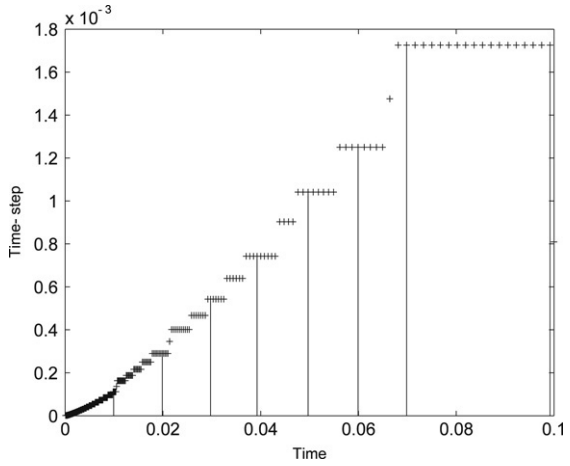

Fig. 8. The time steps as a function of time. The vertical lines are the boundaries between the eight time intervals.

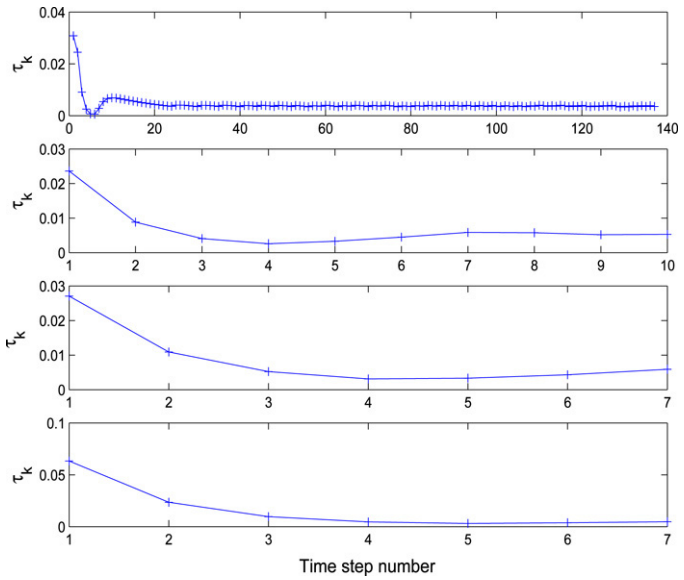

Fig. 9. The measured local discretization error in time $|\tau_k|$ in four intervals.

11. Conclusions

An analysis of the computational work and numerical experiments in one dimension confirm that a finite difference method is more efficient compared to Monte Carlo and quasi-Monte Carlo methods for the accurate solution of the Black–Scholes equation for European multi-asset call options. Therefore, an adaptive method with full error control has been developed for solution of this equation. The multi-dimensional computational grid and the time step are chosen such that a tolerance on a functional of the final global error is satisfied by the solution. The temporal discretization error is estimated *a posteriori* in every step but the spatial grid is constant in intervals of the time domain. In each interval, the error due to the space discretization is first determined *a posteriori* based on the solution and then its growth is estimated *a priori*.

The grid is adjusted in each dimension separately so that its Cartesian structure is maintained. The user has to supply the error tolerance and a maximal number of grid points in each dimension. The algorithm automatically selects the grid and the time steps and provides an upper bound on the numerical error at the final time. The method has been tested successfully for problems with up to three dimensions corresponding to three underlying assets. Comparisons between adapted and equidistant grids with time step control show that lower bounds on the solution error are obtained

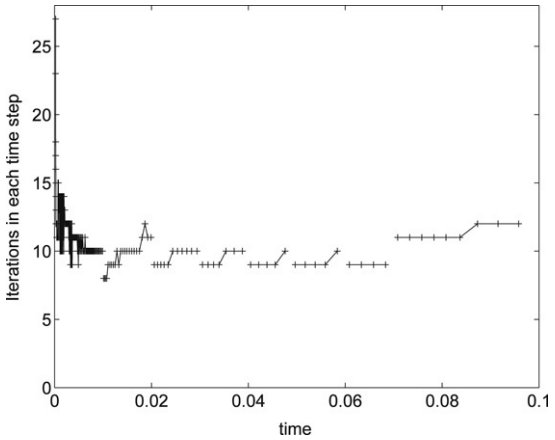

Fig. 10. The number of GMRES iterations in each time step.

with the same number of grid points with adaptation or we satisfy the same bounds with fewer grid points. Since the time step increases rapidly from a low level, important gains in efficiency are achieved with a variable, adapted time step compared to a fixed, small time step.

Acknowledgements

The second author was supported by FMB, the Swedish Graduate School in Mathematics and Computing Science. The third author was partially supported by the Swedish Research Council (VR) under contract 621-2003-5280. The fourth author was partially supported by the Swedish Research Council (VR) under contract 621-2003-5267.

References

[1] F. Black, M. Scholes, The pricing of options and corporate liabilities, Journal of Political Economy 81 (1973) 637–659.
[2] R.C. Merton, Theory of rational option pricing, Bell Journal of Economical Management Science 4 (1973) 141–183.
[3] M. Musiela, M. Rutkowski, Martingale Methods in Financial Modelling, Springer-Verlag, Berlin, 1997.
[4] R.E. Caflisch, Monte Carlo and quasi-Monte Carlo methods, Acta Numerica 7 (1998) 1–49.
[5] P. Glasserman, Monte Carlo Methods in Financial Engineering, Springer-Verlag, New York, NY, 2004.
[6] J. Persson, L. von Sydow, Pricing European multi-asset options using a space–time adaptive FD-method, Technical report 2003-059, Department of Information Technology, Uppsala University, Uppsala, Sweden, 2003, Computing and Visualization in Science (in press). Available at http://www.it.uu.se/research/reports/.
[7] D. Tavella, C. Randall, Pricing Financial Instruments — The Finite Difference Method, John Wiley & Sons, Chichester, 2000.
[8] P. Wilmott, Option Pricing: Mathematical Models and Computation, Oxford Financial Press, Oxford, 1993.
[9] M. Gilli, E. Këllezi, G. Pauletto, Solving finite difference schemes arising in trivariate option pricing, Journal of Economic Dynamics & Control 26 (2002) 1499–1515.
[10] B.J. McCartin, S.M. Labadie, Accurate and efficient pricing of vanilla stock options via the Crandall–Douglas scheme, Applied Mathematics and Computation 143 (2003) 39–60.
[11] E.S. Schwartz, The valuation of warrants: Implementing a new approach, Journal of Financial Economy 4 (1977) 79–93.
[12] B. Engelmann, P. Schwender, The pricing of multi-asset options using a Fourier grid method, Journal of Computational Finance 1 (1998) 53–61.
[13] O. Pironneau, Y. Achdou, Computational Methods for Option Pricing, in: Frontiers in Applied Mathematics, vol. 30, SIAM, Philadelphia, PA, 2005.
[14] O. Pironneau, F. Hecht, Mesh adaption for the Black & Scholes equations, East–West Journal of Numerical Mathematics 8 (2000) 25–35.
[15] H.-J. Bungartz, M. Griebel, Sparse grids, Acta Numerica 13 (2004) 147–269.
[16] C. Reisinger, Numerische Methoden für hochdimensionale parabolische Gleichungen am Beispiel von Optionspreisaufgaben, Ph.D. Thesis, Universität Heidelberg, Heidelberg, Germany, 2004.
[17] P. Lötstedt, S. Söderberg, A. Ramage, L. Hemmingsson-Frändén, Implicit solution of hyperbolic equations with space–time adaptivity, BIT 42 (2002) 128–153.
[18] A. Vande Wouwer, P. Saucez, W.E. Schiesser, Adaptive Method of Lines, Chapman & Hall, Boca Raton, FL, 2001.
[19] S. Figlewski, B. Gao, The adaptive mesh model: A new approach to efficient option pricing, Journal of Financial Economics 53 (1999) 313–351.
[20] P.P. Boyle, M. Broadie, P. Glasserman, Monte Carlo methods for security pricing, Journal of Economic Dynamics & Control 21 (1997) 1267–1321.

[21] C. Joy, P.P. Boyle, K.S. Tan, Quasi-Monte Carlo methods in numerical finance, Management Science 42 (1996) 926–938.
[22] P. Bratley, B.L. Fox, Algorithm 659, Implementing Sobol's quasirandom sequence generator, ACM Transactions on Mathematical Software 14 (1988) 88–100.
[23] I.M. Sobol, On the distribution of points in a cube and the approximate evaluation of integrals, USSR Computational Mathematics and Mathematical Physics 7 (1967) 86–112.
[24] J. Burkardt, School of Computational Science, Florida State University, Tallahassee, FL. http://www.scs.fsu.edu/~burkardt/msrc/msrc.html, 2006.
[25] T. Björk, Arbitrage Theory in Continuous Time, Oxford University Press, New York, NY, 1998.
[26] J.C. Cox, S.A. Ross, The valuation of options for alternative stochastic processes, Journal of Financial Economics 3 (1976) 145–166.
[27] R.E. Caflisch, B. Moskowitz, Modified Monte Carlo methods using quasi-random sequences, in: H. Niederreiter, P.J.S. Shiue (Eds.), Monte Carlo and Quasi-Monte Carlo Methods in Scientific Computing, in: Lecture Notes in Statistics, vol. 106, Springer, New York, 1995, pp. 1–16.
[28] W.J. Morokoff, Generating Quasi-Random paths for stochastic processes, SIAM Review 40 (1998) 765–788.
[29] W.J. Morokoff, R.E. Caflisch, Quasi-Monte Carlo simulation of random walks in finance, in: H. Niederreiter, P. Hellekalek, G. Larcher, P. Zinterhof (Eds.), Monte Carlo and Quasi Monte Carlo Methods 1996, in: Lecture Notes in Statistics, vol. 127, Springer, New York, 1998, pp. 340–352.
[30] N. Clarke, K. Parrott, Multigrid for American option pricing with stochastic volatility, Applied Mathematical Finance 6 (1999) 177–195.
[31] S. Ikonen, J. Toivanen, Operator splitting methods for American option pricing, Applied Mathematical Letters 17 (2004) 809–814.
[32] B.F. Nielsen, O. Skavhaug, A. Tveito, Penalty and front-fixing methods for the numerical solution of American option problems, Journal of Computational Finance 5 (4) (2002) 69–97.
[33] C.W. Oosterlee, On multigrid for linear complementarity problems with application to American-style options, Electronic Transactions on Numerical Analysis 15 (2003) 165–185.
[34] T.A. Manteuffel, A.B. White Jr., The numerical solution of second-order boundary value problems on nonuniform meshes, Mathematics of Computation 47 (1986) 511–535.
[35] E. Hairer, S.P. Nørsett, G. Wanner, Solving Ordinary Differential Equations, Nonstiff Problems, 2nd ed., Springer-Verlag, Berlin, 1993.
[36] R. Becker, R. Rannacher, An optimal control approach to a posteriori error estimation in finite element methods, Acta Numerica 10 (2001) 1–102.
[37] K. Eriksson, D. Estep, P. Hansbo, C. Johnson, Introduction to adaptive methods for differential equations, Acta Numerica 4 (1995) 105–158.
[38] A. Friedman, Partial Differential Equations of Parabolic Type, Prentice-Hall, Englewood Cliffs, NJ, 1964.
[39] Y. Saad, M.H. Schultz, GMRES: A generalized minimal residual algorithm for solving nonsymmetric linear systems, SIAM Journal on Scientific Computing 7 (1986) 856–869.
[40] A. Greenbaum, Iterative Methods for Solving Linear Systems, SIAM, Philadelphia, PA, 1997.