Journal of Computational and Applied Mathematics 236 (2012) 3741–3750

Contents lists available at SciVerse ScienceDirect

Journal of Computational and Applied Mathematics

journal homepage: www.elsevier.com/locate/cam

Option pricing with a direct adaptive sparse grid approach

Hans-Joachim Bungartz, Alexander Heinecke, Dirk Pflüger, Stefanie Schraufstetter *

Institut für Informatik, Technische Universität München, Boltzmannstr. 3, 85748 Garching, Germany

ARTICLE INFO

Article history:
Received 1 October 2010
Received in revised form 31 August 2011

Keywords:
Black–Scholes equation
Option pricing
Sparse grids
Finite elements
Adaptivity

ABSTRACT

We present an adaptive sparse grid algorithm for the solution of the Black–Scholes equation for option pricing, using the finite element method. Sparse grids enable us to deal with higher-dimensional problems better than full grids. In contrast to common approaches that are based on the combination technique, which combines different solutions on anisotropic coarse full grids, the direct sparse grid approach allows for local adaptive refinement. When dealing with non-smooth payoff functions, this reduces the computational effort significantly. In this paper, we introduce the spatially adaptive discretization of the Black–Scholes equation with sparse grids and describe the algorithmic structure of the numerical solver. We present several strategies for adaptive refinement, evaluate them for different dimensionalities, and demonstrate their performance showing numerical results.

1. Introduction

For solving multi-dimensional option pricing problems, Monte Carlo (MC) methods are widespread as the favorite method. They are flexible, can be implemented in a straightforward way and handle multiple dimensions, but they exhibit rather slow convergence rates. Several approaches based on variance reduction techniques exist, improving the speed of convergence. Consider, for example, quasi-MC methods, control variate techniques, adaptive MC, or multi-level MC (see, e.g., [1,2]). But in recent years, also PDE methods have been examined for financial problems (cf., e.g., [3–6]) due to their fast speed of convergence compared to MC techniques which theoretically allows to obtain higher accuracies. Additionally, PDEs allow the fast computation of Greeks, that are needed for hedging tasks, as an additional benefit. Unfortunately, up to now, PDE methods usually cannot complete against MC methods in three or more dimensions in terms of computing time: the computations are rather costly and the PDE approach suffers from the so-called curse of dimensionality, the exponential dependency on the dimensionality. Hence, common PDE approaches as the ones stated above are currently typically limited to two or, in the best case, three dimensions.

Our aim is to push forward the limit in terms of accuracy and dimensionality of what can be done employing PDE methods. To remedy the curse of dimensionality at least to some extent, sparse grids [7] can be employed. They enable us to reduce the complexity for a discretization with a mesh-width of $h_n := 2^{-n}$ (spending $2^n + 1$ grid points in every dimension) from $\mathcal{O}((2^n)^d)$ for full grids to $\mathcal{O}(2^n n^{d-1})$, while maintaining a comparable discretization error. With this, it is possible to tackle up to five dimensions as we will show.

In the context of sparse grids, typically the so-called sparse grid combination technique [8] is employed in different variants for financial option pricing (cf. [9,10, e.g.]). It is based on a multivariate extrapolation scheme, obtaining an approximation to the sparse grid solution by solving the problem on a sequence of smaller, but full anisotropic grids. This has the advantage that conventional solvers for full grids can be readily employed and that parallelization is straightforward. Their drawback is that they do not support local (spatial) adaptivity, which is highly desirable in the field of option pricing.

* Corresponding author.
E-mail addresses: bungartz@in.tum.de (H.-J. Bungartz), heinecke@in.tum.de (A. Heinecke), pflueged@in.tum.de (D. Pflüger), schraufs@in.tum.de (S. Schraufstetter).

0377-0427/$ – see front matter © 2011 Elsevier B.V. All rights reserved.
doi:10.1016/j.cam.2011.09.024

Since initial values given by the option's payoff function usually are non-smooth, the regularity conditions assumed in the theory of sparse grids are not fulfilled, and, thus, results of convergence estimations for the combination technique do not hold.

In contrast, we aim to directly work in the sparse grid's function space to solve the Black–Scholes equation using finite elements (FE). While this is algorithmically challenging (culminating in the UpDown algorithm [11] which enables to efficiently apply dense matrices, arising from hierarchical FE discretizations, to vectors), it allows to employ locally adaptive refinement. Previously, we have shown that the parallelization of the direct sparse grid approach can be achieved, too, though being much more sophisticated [12]. We now present first results obtained with different strategies for adaptive refinement. They clearly demonstrate the suitability of our approach, being able to adapt to the properties of the functions that arise during the solving process of the Black–Scholes equation.

First, we present the Black–Scholes equation and derive its finite element formulation in Section 2. Sparse grids and their core concepts and algorithmic structures are described in Section 3. In Section 4, we introduce strategies for adaptive refinement which are evaluated in Section 5. Section 6 finally concludes this work.

2. Numerical solution of the Black–Scholes equation

The Black–Scholes model is widely used in option pricing. It assumes that a stock value S follows a geometric Brownian motion defined by the stochastic differential equation

$$dS(t) = \mu_{\text{GBM}} S(t)\, dt + \sigma_{\text{GBM}} S(t)\, dW(t) \tag{1}$$

with a drift μ_{GBM}, a standard deviation σ_{GBM}, and a stochastic Wiener process $W(t)$. The Black–Scholes equation for pricing a European basket option on d assets and an expiration time T is then given by

$$\frac{\partial V}{\partial t} + \frac{1}{2}\sum_{i,j=1}^{d} \sigma_i \sigma_j \rho_{ij} S_i S_j \frac{\partial^2 V}{\partial S_i \partial S_j} + \sum_{i=1}^{d} \mu_i S_i \frac{\partial V}{\partial S_i} - rV = 0, \tag{2}$$

where $V(\mathbf{S}, t)$ denotes the value of the option, $t \in [0, T]$ the forward time, $\mathbf{S} = (S_1, \ldots, S_d) \in \mathbb{R}^d_+$ the stock values, σ_i the volatilities, ρ_{ij} the (symmetric) asset correlations, μ_i the drifts, and r the risk-free interest rate. Note that the μ_i do not correspond to the drift μ_{GBM} of the geometric Brownian motion (1), but usually contain the risk-free rate as well as dividend payments.

Eq. (2) has variable coefficients depending on S_i, since stock values grow exponentially in time,

$$S(t) = S(0) \exp\left[\left(\mu_{\text{GBM}} - 0.5\sigma^2_{\text{GBM}}\right) t + \sigma_{\text{GBM}} W(t)\right], \tag{3}$$

due to their geometric Brownian motion (1). This leads to bad numerical behavior when solving Eq. (2) numerically, discretized on a uniform Cartesian grid (see, e.g., [13]). To avoid this, there exist different approaches. Leentvaar [9] introduces stretched grids which take the exponential growth of the stock values into account and lead to a non-equidistant discretization. An alternative approach eliminates the variable coefficients by a logarithmic variable transformation. Wilmott [14] suggests a transformation to the heat equation, but also the simple and well-known log-transformation $S_i := \log S_i$ eliminates the variable coefficients.

In this paper, we restrict ourselves to the log-transformed Black–Scholes equation

$$\frac{\partial V}{\partial t} + \frac{1}{2}\sum_{i,j=1}^{d} \sigma_i \sigma_j \rho_{ij} \frac{\partial^2 V}{\partial S_i \partial S_j} + \sum_{i=1}^{d} \left(\mu_i - \frac{1}{2}\sigma_i^2\right) \frac{\partial V}{\partial S_i} - rV = 0. \tag{4}$$

Of course, all presented algorithms can be also applied in a straightforward way to the non-transformed Black–Scholes equation, but then convergence is slow if no other methods such as grid stretching are applied.

As the terminal condition, we will consider the standard payoff function

$$V(T, \mathbf{S}) := \begin{cases} \max\left\{K - \dfrac{1}{d}\displaystyle\sum_{i=1}^{d} S_i,\ 0\right\}, & \text{in case of a put option,} \\ \max\left\{\dfrac{1}{d}\displaystyle\sum_{i=1}^{d} S_i - K,\ 0\right\}, & \text{in case of a call option,} \end{cases} \tag{5}$$

which depends on the strike K and the arithmetic mean of the stock values S_i. Since Eq. (5) forms the terminal condition and, thus, Eq. (2) has to be solved backward in time, we additionally define the backward time $\tau := T - t$ and consider $u(\mathbf{S}, \tau) := V(\mathbf{S}, T - t)$ instead of $V(\mathbf{S}, t)$ in the following.

To solve the Black–Scholes equation, we apply the FE method. In contrast to the method of finite differences, it has weaker regularity assumptions, allows the choice of different basis functions such as hierarchical bases, and adaptivity can be integrated easily. Thus, the FE method fits well to the concept of sparse grids described in Section 3.

Applying the FE method to the Black–Scholes equation (4) leads to the weak form

$$\frac{\partial}{\partial \tau}\langle u, w\rangle + \frac{1}{2}\sum_{i,j=1}^{d}\sigma_i\sigma_j\rho_{ij}\left\langle \frac{\partial u}{\partial S_i}, \frac{\partial w}{\partial S_j}\right\rangle - \sum_{i=1}^{d}\left(\mu_i - \frac{1}{2}\sigma_i^2\right)\left\langle \frac{\partial u}{\partial S_i}, w\right\rangle + r\langle u, w\rangle = 0, \tag{6}$$

with the standard L^2 inner product $\langle\cdot,\cdot\rangle$ on a domain Ω and $w(\mathbf{S})$ denoting a test function. We use Dirichlet boundaries with discounting, which can be assumed if the domain is chosen large enough. Thus, the test functions $w \in H_0^1(\Omega)$ vanish at the boundary $\partial\Omega$, and boundary integrals are already neglected in Eq. (6).

The size of the bounded domain $\Omega = [0, S_{\max,1}] \times \cdots \times [0, S_{\max,d}]$ is estimated based on [15]. There, an estimation for the error resulting from the truncation of the domain and the introduction of artificial far-field boundaries with Dirichlet conditions is given. From this, it follows for our problem for the error at a point $(\mathbf{S}^*, t^*) \in R \times [0, T]$ with $R = (0, S_{\max,1}) \times \cdots \times (0, S_{\max,d})$ satisfying

$$\ln\frac{S_{\max,i}}{S_i^*} \geq (2\mu_i - \sigma_i^2)(T - t^*)$$

the estimation

$$|v(\mathbf{S}^*, t^*) - u(\mathbf{S}^*, t^*)| \leq \|v - u\|_{L^\infty((t,T)\times(\partial R\cap(0,\infty)^d))} \times \sum_{i=1}^{d}\exp\left[-\frac{\ln\frac{S_{\max,i}}{S_i^*}\left(\ln\frac{S_{\max,i}}{S_i^*} + \min\{0, \sigma_i^2 - 2\mu_i\}(T - t^*)\right)}{2\sigma_i^2(T - t^*)}\right]. \tag{7}$$

Hereby, $v(\mathbf{S}, t)$ denotes the exact solution of the initial value problem with the PDE (2) on an unbounded domain $[0, \infty)^d$, and $u(\mathbf{S}, t)$ denotes the solution of the initial value problem on the bounded domain Ω. To restrict the error resulting from the domain truncation to ε, after some calculations according to the examples in [15], it follows from Eq. (7) for the upper bounds $S_{\max,i}$ of the domain Ω the condition

$$S_{\max,i} \geq S_i^* \cdot \exp\left(C \cdot \sigma_i \cdot \sqrt{T}\right), \tag{8}$$

with $C = \sqrt{2\ln\frac{d\cdot K}{\varepsilon}}$ which is typically chosen as a value of the interval [3, 10], e.g. $C = 5$.

For the discretization in space, we use the well-known Galerkin projection onto the space of piecewise d-linear hierarchical basis functions $V_N := \operatorname{span}\{\varphi_q(\mathbf{S});\ q = 1, \ldots, N\}$, which is introduced in the next section. Following the Ritz–Galerkin approach, we make the ansatz

$$u(\mathbf{S}, \tau) = \sum_{q=1}^{N}\alpha_q(\tau)\varphi_q(\mathbf{S}), \tag{9}$$

which yields the discretized FE formulation

$$B\frac{\partial}{\partial \tau}\boldsymbol{\alpha}(\tau) = -\frac{1}{2}\sum_{i,j=1}^{d}\sigma_i\sigma_j\rho_{ij}C^{(ij)}\boldsymbol{\alpha} + \sum_{i=1}^{d}\left(\mu_i - \frac{1}{2}\sigma_i^2\right)D^{(i)}\boldsymbol{\alpha} - rB\boldsymbol{\alpha} \tag{10}$$

with a coefficient vector $\boldsymbol{\alpha} := (\alpha_q)_{q=1,\ldots,N}$ and matrices $B, C^{(ij)}, D^{(i)} \in \mathbb{R}^{N,N}$ defined by $B = (\langle\varphi_p, \varphi_q\rangle)_{p,q}$, $C^{(ij)} = \left(\left\langle\frac{\partial\varphi_p}{\partial S_j}, \frac{\partial\varphi_q}{\partial S_i}\right\rangle\right)_{p,q}$, and $D^{(i)} = \left(\left\langle\varphi_p, \frac{\partial\varphi_q}{\partial S_i}\right\rangle\right)_{p,q}$. Analogous calculations lead to the formulation for the non-transformed case.

We do discretization in time with the quadratic Crank–Nicolson scheme. Since the payoff function is non-smooth at initial time $\tau = 0$, we apply additionally a Rannacher smoothing [16] with backward Euler during the first few timesteps. The resulting non-symmetric system of linear equations can be solved with the conjugate gradient-type solver BiCGStab.

3. The direct sparse grid approach

To obtain the space of piecewise d-linear functions V_N with mesh-width h_n, and to overcome the curse of dimensionality to some extent, we employ sparse grids. They have been successfully employed for a variety of problems and applications, ranging from astrophysical parameter estimations to regression and classification in data mining, spatial adaptive refinement allowing to treat even non-smooth problems, cf. [8,11] and the references therein.

In the following, we briefly introduce the basic principles and sketch the algorithmics we require.

3.1. Sparse grids

Sparse grids are based on a hierarchical formulation of the underlying basis, extending the one-dimensional setting to the multivariate case via a tensor product approach. Therefore, we use the one-dimensional hierarchical hat functions

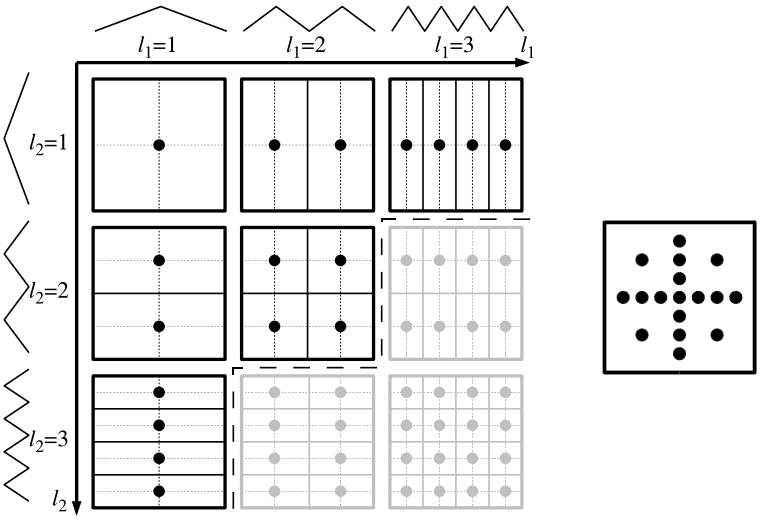

Fig. 1. The tableau of subspaces $W_{\mathbf{l}}$ in two dimensions for $V_3^{(1)}$ (left) and the resulting sparse grid (right).

$\varphi_{l,i}(x) = \varphi(2^l x - i)$ that depend on a level l and an index i and are based on the standard hat function $\varphi(x) = \max(1-|x|, 0)$, and define then piecewise d-linear basis functions by

$$\varphi_{\mathbf{l},\mathbf{i}}(\mathbf{x}) := \prod_{j=1}^{d} \varphi_{l_j,i_j}(x_j) \tag{11}$$

with multi-indices $\mathbf{l} = (l_1, \ldots, l_d)$ and $\mathbf{i} = (i_1, \ldots, i_d)$. Each basis function $\varphi_{\mathbf{l},\mathbf{i}}$ is then centered at the corresponding grid point $x_{\mathbf{l},\mathbf{i}} := \left(2^{-l_1} \cdot i_1, \ldots, 2^{-l_d} \cdot i_d\right) \in [0, 1]^d$. Note, that simple translation and scaling allows to adapt to arbitrary rectangular domains.

Leaving out the even indices on each level $l > 0$, we can define a hierarchy of d-dimensional increments (subspaces) $W_{\mathbf{l}}$ which are spanned by the bases

$$\Phi_{W_{\mathbf{l}}} := \left\{\varphi_{\mathbf{l},\mathbf{i}}(\mathbf{x}) : i_j = 1, \ldots, 2^{l_j} - 1,\ i_j \text{ odd}, 1 \le j \le d\right\}. \tag{12}$$

Note that all basis functions of a subspace $W_{\mathbf{l}}$ have supports of the same size and shape with pairwise disjoint interiors. To be able to deal with non-zero boundaries, the one-dimensional basis can be extended by the two basis functions $\varphi_{0,0}$ and $\varphi_{0,1}$ of level 0 which are located on the boundary $\partial\Omega$. Note further that we consider Dirichlet boundary values to solve the Black–Scholes equation, which implies that we have to solve the PDE for inner grid points only.

If the functions we are dealing with fulfill certain smoothness requirements (i.e., the mixed second order derivatives have to be bounded in Ω), we can select those hierarchical increments a priori which contribute most to the overall solution. This can be formulated as a continuous optimization problem [8] and results in the sparse grid space

$$V_n^{(1)} := \bigoplus_{|\mathbf{l}|_1 \le n+d-1} W_{\mathbf{l}}, \tag{13}$$

a direct sum of subspaces $W_{\mathbf{l}}$, which is the best choice with respect to both the L^2- and L^∞-norms for the overall level n. Fig. 1 (left) shows the tableau of subspaces in two dimensions for $n = 3$ (neglecting the grid points on the boundary) and the resulting sparse grid (right). Selected subspaces are colored black. Any sparse grid function $u(\mathbf{x}) \in V_n^{(1)}$ can then be written as a linear combination of the respective basis functions,

$$u(\mathbf{x}) = \sum_{\mathbf{l},\mathbf{i}} v_{\mathbf{l},\mathbf{i}} \varphi_{\mathbf{l},\mathbf{i}}(\mathbf{x}),$$

with coefficients $v_{\mathbf{l},\mathbf{i}}$. In the hierarchical representation, the coefficients are contributions to the sum of the basis functions on lower levels and are thus referred to as hierarchical surpluses.

While this already reduces the number of degrees of freedom significantly to $\mathcal{O}(2^n n^{d-1})$, they can be typically restricted even further. To adapt to the special characteristics of a function at hand, and to handle functions that do not meet the smoothness requirements, an additional a posteriori refinement should be used. This allows to select those grid points of the sparse grid's structure during the solution of the PDE which are most important, as we discuss and demonstrate in the next two sections.

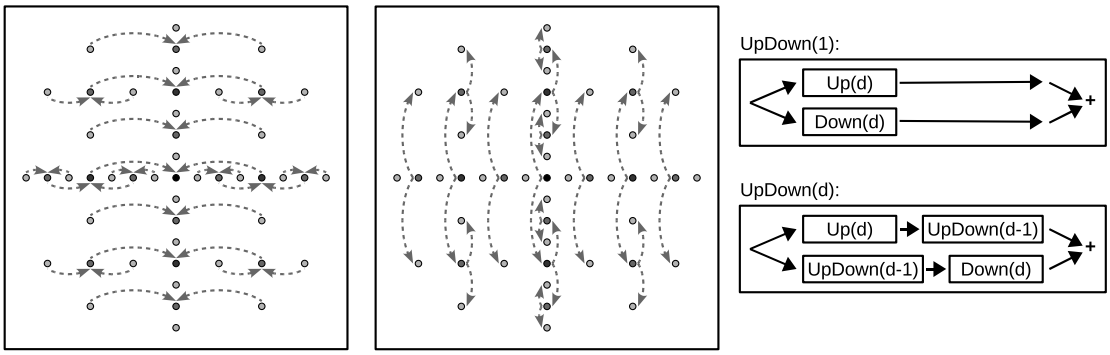

Fig. 2. Up (left) in the first dimension and Down (middle) in the second one for a two-dimensional sparse grid of level $n = 4$; arrows indicate the corresponding one-dimensional tree traversals. On the right, the algorithmic structure of the UpDown scheme, recursing in the dimension d.

3.2. Algorithms working on sparse grids

While sparse grids significantly reduce the number of grid points compared to full grids, algorithms working on them require a higher computational effort. Especially sophisticated and adapted algorithms have to be implemented to retain efficiency. This is due to the fact that, in general, more basis functions have overlapping supports: In the case of a full grid for the space of piecewise d-linear functions, a basis function has overlapping support with its $3d - 1$ direct neighbors; in the hierarchical basis it has overlapping support with all hierarchical ancestors and all hierarchical children.

Correspondingly, the matrices assembled when solving the Black–Scholes equations are not sparse. In case of N unknowns, a direct application of the matrices to a vector would require about $\mathcal{O}(N^2)$ operations. Fortunately, the tensor product structure of the underlying basis can be exploited: The matrices do not have to be assembled, which reduces the memory requirements, and they can even be applied in linear time with respect to the number of unknowns.

The basic idea is known as the UpDown scheme (see [11, e.g.]). It is based on a unidirectional principle [17] which exploits the tensor product structure of the underlying basis. The application of a matrix stemming from a bilinear form can be split into d successive applications of the corresponding one-dimensional counterparts to each one-dimensional substructure in the respective dimension. To illustrate this, consider, e.g., the matrix B (10) for the scalar product $\langle \varphi_{\mathbf{l},\mathbf{i}}, \varphi_{\mathbf{l}',\mathbf{i}'} \rangle$, for which we obtain

$$\langle \varphi_{\mathbf{l},\mathbf{i}}(\mathbf{x}), \varphi_{\mathbf{l}',\mathbf{i}'}(\mathbf{x}) \rangle = \left\langle \prod_{k=1}^{d} \varphi_{\mathbf{l}_k,\mathbf{i}_k}(x_k), \prod_{k=1}^{d} \varphi_{\mathbf{l}'_k,\mathbf{i}'_k}(x_k) \right\rangle = \prod_{k=1}^{d} \langle \varphi_{\mathbf{l}_k,\mathbf{i}_k}(x_k), \varphi_{\mathbf{l}'_k,\mathbf{i}'_k}(x_k) \rangle. \tag{14}$$

Thus, the contributions of the single dimensions can be decoupled and computed successively.

The approach leads to efficient algorithms which work for both regular (non-adaptive) and adaptive sparse grids, and which can even be parallelized as demonstrated for multi-core systems in [12]. An additional benefit is that only one-dimensional operations (scalar products) have to be derived and implemented, and can then be used multiple times. For (iteratively) solving the Black–Scholes equation, we thus have to derive the corresponding one-dimensional operations and combine them in an appropriate way according to Eq. (14) in order to apply the matrices B, $C^{(ij)}$, and $D^{(i)}$ in Eq. (10). Their one-dimensional versions are then based on (recursive) traversals of the one-dimensional subgrids based, and are implemented by a traversal of the respective binary tree of basis functions.

But whereas the iteration over all dimensions can be directly realized for full grids, all intermediate values for two basis functions have to be stored at a common parent in the hierarchical structure. This requires a certain nesting of the one-dimensional applications and forces to split the one-dimensional matrices into their contributions from lower to higher levels and vice versa, which explains the name of this scheme, UpDown. Fig. 2 shows both Up and Down for one dimension each in a two-dimensional setting and the algorithmic structure of the nesting of Ups and Downs in general. For all further details, which go far beyond the scope of this paper (see [11, e.g.]).

4. Adaptivity

The classical error bounds for the sparse grid technique only apply for sufficiently smooth functions. But it is well-known that functions which locally do not fulfill the smoothness assumptions can be successfully dealt with, too, if adaptive refinement is employed [11]. For the solution of the (log-transformed) Black–Scholes equation, the initial condition at time $\tau = 0$ given by the payoff function is clearly non-smooth, exhibiting a $d - 1$-dimensional singularity in the first derivative.

To cope with this, sparse grids of very high levels leading to a high resolution have to be spent in general or, alternatively, adaptive refinement has to be used. The latter one allows to spend a higher resolution locally in critical parts and only little effort in smooth regions. For this, a suitable criterion for refinement has to be found which invests grid points only where

necessary and which adapts to the function's change with backward time τ. To this end, we examined different criteria for an adaptive refinement of the payoff function at initial time and several ways for adapting the grid during the calculation process.

4.1. Adaptive discretization at initial time

Before solving the Black–Scholes equation numerically, the initial values have to be interpolated as well as possible: A highly accurate numerical solver does not help if the discretization errors of the initial values are too large. Indeed, the main challenge for adaptive refinement is to be able to resolve the singularity of the payoff function as well as possible.

Therefore we analyzed different adaptation strategies. All of them are based on the hierarchical surplus introduced in Section 3.1 which is a measure for the smoothness of the interpolated function. Non-smooth areas induce higher hierarchical surpluses than smooth regions; the surplus $\alpha_{\mathbf{l},\mathbf{i}}$ depends on the mixed second order derivatives of the function v which is to be interpolated via their integral representation [8]

$$\alpha_{\mathbf{l},\mathbf{i}} = \int_{\Omega} -2^{-(|\mathbf{l}|_1+d)} \varphi_{\mathbf{l},\mathbf{i}}(\mathbf{x}) \cdot D^{\mathbf{2}} v(\mathbf{x})\, d\mathbf{x}, \quad D^{\mathbf{2}} v(\mathbf{x}) = \frac{\partial^{2d}}{\partial x_1^2 \ldots \partial x_d^2} v(\mathbf{x}). \tag{15}$$

Refining the grid points with the largest surpluses tends to minimize the L^2-norm of the error.

The classical strategy for adaptive refinement which has been successfully employed in different settings [11] is to refine a certain number or percentage of the grid points with the largest absolute surpluses. The nodal refinement is typically done by generating all $2d$ hierarchical descendants in the hierarchical grid structure. Applying this refinement strategy, it is important to maintain the grid's hierarchical structure in each dimension in order to guarantee a working and correct UpDown scheme. This might require the creation of needed hierarchical ancestors up to level 0 on the boundaries. Note that for problems with irrelevant dimensions, this can be even extended further in a straightforward way, restricting the refinement to relevant dimensions.

In our setting, this leads to a refinement which is too steep due to the non-smoothness of the initial solution. Due to the simple and symmetric shape of the payoff function at initial time $\tau = 0$, a general adaptation strategy is to refine all points with surpluses that exceed a given threshold. This can be repeated, leading to a refinement in a breadth-first manner in the tree of hierarchical basis functions.

This approach works particularly well when it is additionally combined with a limitation of the maximum refinement level with respect to each dimension. For a certain level n this restricts new grid points to the corresponding full grid space $V_n = \bigoplus_{|\mathbf{l}|_\infty \leq n} W_{\mathbf{l}}$. Altogether, this refinement strategy leads, as desired, to grids with full-grid-like refinements around the kink of the payoff function, which enables to represent the initial payoff function very well, as we will show for some examples in Section 5. In the following, we refer to this strategy of adaptive refinement as *standard adaptation*.

To better adapt to the problem's properties, we have additionally examined a *weighted adaptation*. The idea is to make use of the knowledge that the region around the strike, i.e. $S_i \approx K$, is most important, and that the contribution of grid points which are located close to the domain's boundary is of less influence. Thus, adaptive refinement is most efficient around the strike. To incorporate this knowledge, we weight the hierarchical surplus of every point with the d-dimensional Gaussian function centered at the strike, and consider the resulting values instead of the mere surplus as refinement indicators.

4.2. Adaptation strategies during the solving process

After the first time step, the initial singularity of the payoff function is smoothed by the diffusion component of the Black–Scholes equation. Still, the curvature is highest around the former singularity, and a finer resolution is required there. Thus, the initial grid is still well-suited for the solving process in the next time steps.

Nevertheless, the function changes its behavior with evolving time, and it is therefore reasonable to adapt the grid to the new setting. Further refinements just increase the number of grid points, but have only little influence on the final error at $\tau = T$: They cannot reduce the error that has accumulated so far. On the other hand, the diffusion component tames the function more and more such that initially refined grid points get redundant. Their surpluses and, thus, contributions to the function decay quickly. We have therefore added a coarsening procedure after each time step that removes all grid points (having no hierarchical descendants) of which the surplus did fall below a certain coarsening threshold. This increases the error only slightly but reduces the overall effort significantly.

5. Numerical results

In this section, we present numerical results for our adaptive Black–Scholes solver. We restrict ourselves to put options, but call options lead to comparable results, too. All computations have been done with the payoff function (5) with strike $K = 1$ and expiration time $T = 1$. We set arbitrary values $\mu_i \in [0.05, 0.1]$, $\sigma_i \in [0.2, 0.4]$, and non-zero correlations $\rho_{ij} \in [-0.7, 0.4]$ to deal with complex and costly scenarios. The estimation of the domain size for $\mathbf{S}$ is based on the diffusion (3) induced by the stock values, but setting the left boundaries to 0.11 rather than 0 due to the log-transformation of the Black–Scholes equation. The relative error is always considered in the L^∞-norm on a full grid sub-domain (spanning 5% of the domain with 20 points per dimension) surrounding the evaluation point $S_i = 1$, relative to the solution obtained for a

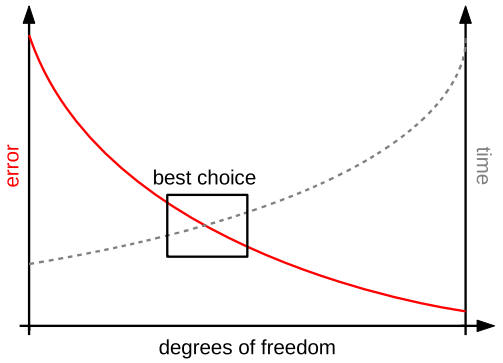

Fig. 3. Criteria for optimal adaptivity: the numerical error as well as computation time should be minimized.

regular sparse grid with a high resolution which we can take as a rather accurate result to compare against. Furthermore, we verified the reasonability of our results with Monte Carlo simulations.

In the following, we discuss criteria for well-suited adaptive refinement and show results for our refinement strategies based on them.

5.1. Choice of criteria for adaptive refinement

In general, spending more grid points leads to a lower error for any spatial discretization technique. But too many grid points result in runtimes which are just too high to be affordable. Thus, a suitable trade-off has to be found. Fig. 3 sketches the behavior.

How fast the error is reduced depends, of course, on how well-suited the criterion for the adaptive refinement is. Each criterion once again depends typically on a few parameters. A parameter can be, for example, the number of grid points which are to be refined simultaneously. This influences how steep or broad the refinement is; a steep refinement results in very unbalanced grids which lead to a higher computational effort per iteration of the Black–Scholes solver.

As higher dimensions require more computations per degree of freedom (and higher memory requirements), the computing time usually limits either the accuracy that can be reached or the number of dimensions that can be tackled. Thus, it is more critical to find an optimal adaptively refined grid which leads to a good trade-off. The refinement criterion's parameters can typically be optimized by a simple search strategy or by studying them in lower-dimensional settings (where solutions can be quickly obtained) and transferring them to the higher-dimensional case.

5.2. Results

As a good criterion for adaptive refinement has to target the $d-1$-dimensional singularity (ideally spending full-grid-like structures there), we limit the maximum level with respect to each dimension and refine all grid points at once of which the surplus exceeds a certain threshold. Starting with a regular grid for a reasonably high level, often only a few steps of adaptive refinement are necessary. Here, a threshold of refinement of 10^{-6} was used in all settings. Note, that having to resolve a $d-1$-dimensional singularity with full-grid-like structures is exactly what limits the maximum dimensionality we can deal with using sparse grids; it is not the complexity of our method. Fortunately, and due to our refinement strategy, this applies only locally to the region around the strike. Otherwise, the full curse of dimensionality would be back due to the payoff's singularity.

This refinement strategy leads to adaptively refined grids which can well express the initial condition provided by the payoff function. Fig. 4 shows an adaptively refined grid for a two-dimensional option. We plot the solution of the PDE at the grid points' locations, both for the log-transformed setting (left) and transferred back to Cartesian coordinates (right), to give an impression of the grids that are created. It can be clearly seen that most refinement is done around $\mathbf{S} = (0, 0)$ in the log-transformed case (and thus $\mathbf{S} = (1, 1)$ in Cartesian coordinates) where we are most interested in the option's price. The effect of using the weighted adaptation can be observed, putting less emphasis to less relevant regions close to the boundary.

The grid in Cartesian coordinates furthermore shows that the log-transformation additionally shifts the general structure of the sparse grid toward the region of interest. In total, the diagonal kink of the payoff function is resolved very well, allowing to start to solve the Black–Scholes equation. The refinement that can be observed to some extent in outer regions, and especially at the domain's boundary, is due to the hierarchical structure of the grid. If a new point is added, all hierarchical ancestors have to be created, too, to keep the hierarchical structure of the sparse grid consistent such that general algorithms as described in Section 3.2 can be applied without further modifications.

The benefit obtained by employing adaptive refinement is even more significant in higher dimensions. Fig. 5 shows the grid points for a three-dimensional option. There, a two-dimensional singularity has to be resolved, which can be achieved very well.

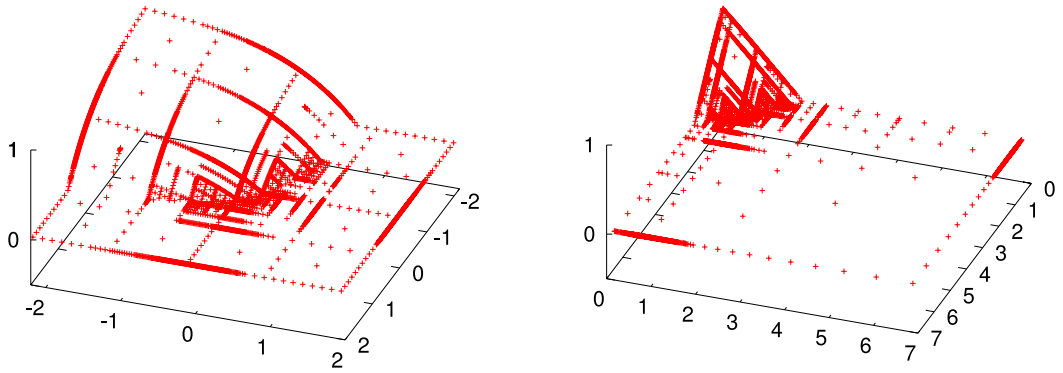

Fig. 4. A two-dimensional sparse grid function with weighted adaptation and a maximum 1*d* level of 9 in log-coordinates (left) and Cartesian coordinates (right).

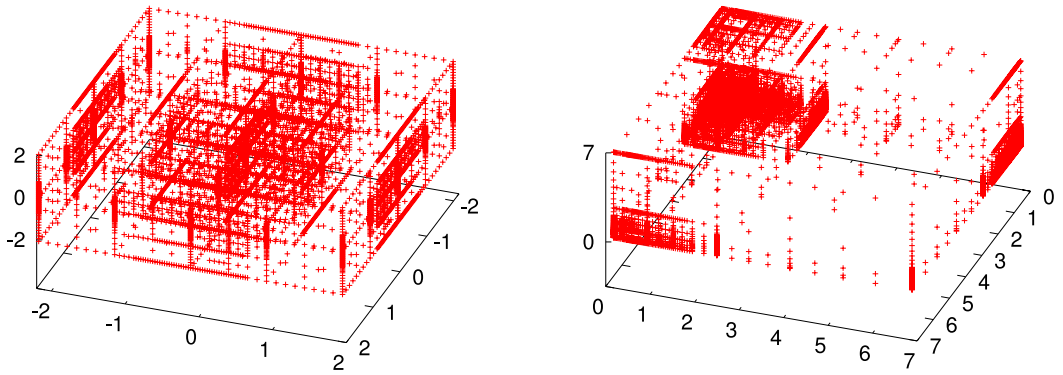

Fig. 5. A three-dimensional sparse grid with weighted adaptation and a maximum 1*d* level of 7 in log-coordinates (left) and Cartesian coordinates (right).

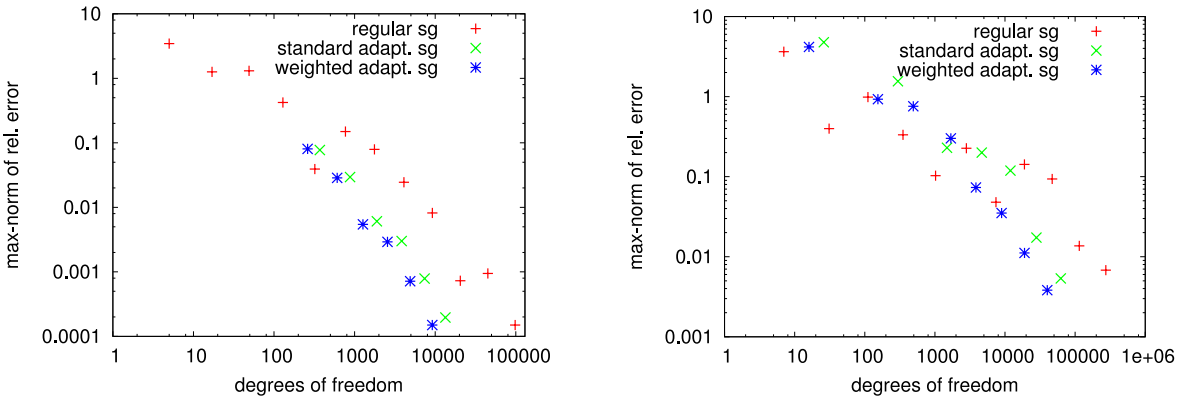

Fig. 6. The L^∞-error for a two-dimensional put option (left) and a three-dimensional put option (right) using regular sparse grids of different levels and adaptively refined grids obtained with the standard and the weighted adaptive refinement strategies.

Fig. 6 compares the different refinement strategies discussed in Section 4 for a two-dimensional option and a three-dimensional option. It shows the L^∞-norm of the error for regular sparse grids and both the standard and the weighted adaptation strategies. For all of them, we can observe convergence, the error being reduced with an increasing number of degrees of freedom. Note that we do not align the grid at which we evaluate the error to fit to the sparse grid points to keep the setting as general (and close to the worst case) as possible. This can explain the outliers obtained for the regular sparse grid, a fortunate alignment leading to a lower error by chance. Furthermore, for higher-dimensional scenarios, more grid points are required in order to fully gain from adaptivity.

Compared to regular sparse grids, already our standard strategy for adaptive refinement significantly reduces the number of grid points which are required to obtain a certain error. Using the same parameters for the adaptive refinement criterion, we can further reduce the size of the grid if we employ additionally the Gaussian weight function. This enables to obtain

Table 1
Number of grid points that are required to obtain a certain accuracy for regular and adaptive sparse grids with both the standard and the weighted adaptation strategies. The percentage values provided show how much the grid size is reduced compared to the non-adaptive case. For the 4*d* and 5*d* options, we can only compare to Monte Carlo results obtained at the strike (with 10^{10} paths). Both the regular sparse grids and the standard refinement scheme require too many grid points in higher dimensionalities to obtain high accuracies.

d	Acc. L^∞		Regular sg	Std. adaptation		W eighted adapt.	
2	4 digits	(0.0002)	98.305	13.402	(13.6%)	9.202	(9.4%)
3	3 digits	(0.005)	274.431	43.589	(15.9%)	28.488	(10.4%)
4	3 digits^{MC}	(0.001)	–	91.039	(–)	16.670	(–)
5	4 digits^{MC}	(0.0002)	–	–		89.704	(–)

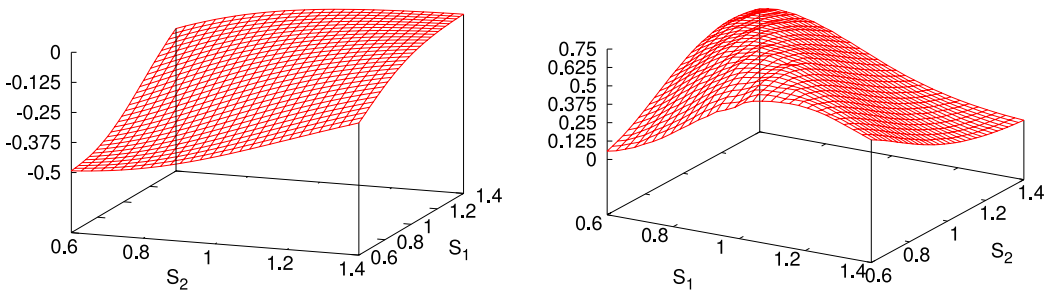

Fig. 7. Delta (left) and gamma (right) plots for underlying S_1 of a two-asset European put basket option with $K = 1$ and $T = 1$.

almost the same accuracy with less grid points, as the results of Table 1 show. In two dimensions, the regular sparse grid of level 13 requires 98.305 grid points (in comparison, a full grid with the same mesh-width would contain more than 67 million grid points). Employing the standard criterion for adaptive refinement reduces the number of unknowns to 13.402 which is only 13.6% of the regular sparse grid's size. Using the weighted adaptation scheme, even less grid points are required: With 9.202 degrees of freedom and only 9.4% of the regular sparse grid's size, the same accuracy of 4 digits can be reached.

In higher-dimensional settings, employing adaptivity cannot reduce the grid sizes by the same percentage which is due to the fact that a $d-1$-dimensional singularity has to be resolved with regular, full grid-like substructures. Starting with 4 dimensions, we have to compare with MC simulations, as even the size of (regular) sparse grids exceeds what we can compute in reasonable time. In 5 dimensions, even the standard refinement scheme (refining a $d-1$-dimensional singularity) requires to create too many grid points for reasonable runtimes, whereas the weighted, problem-adapted refinement strategy works very well.

In contrast to the L^∞-norm, which is considered in Table 1 as it is relevant for financial engineers who are interested in the worst case, measuring in the L^2-norm reduces the error by one magnitude. Similarly, simplifying the scenario by setting $\mu_i = \rho_{ij} = 0$ as it is often done reduces both error norms at least by another factor of 10.

We would like to conclude our numerical experiments by an outlook on delta- and gamma-hedging strategies, whose idea is to hedge changes of the underlying's price (for details see [18]). As stated in the introduction, calculating the first and second partial derivative for each asset comes nearly for free when using a PDE-based solution of the Black–Scholes equation: we obtain an explicit function as a solution, not only function evaluations.

Here, we employ finite differences for determining the delta and gamma of each underlying. Therefore, a huge number of function evaluations of the sparse grid function is required. However, this is not a limitation since there are efficient algorithms for data-driven problems on sparse grids, which allow one to unleash the potential of modern hardware such as multi-core CPUs and GPUs [19]. Fig. 7 illustrates delta and gamma in the area around the evaluation point ($(S_1, S_2) \in [0.6, 1.4]^2$). The left plot shows the delta calculated on an adaptively refined grid, the right plot the corresponding gamma.

The two plots nicely demonstrate that adaptive sparse grids do not only allow to reach options prices of very high accuracy with excellent convergence rates. They also allow a proper calculation of hedging parameters with a minimal additional effort: the interesting sub-domains of delta and gamma are located around the payoff function's kink. And exactly this area is very well-resolved by the adaptive refinement strategy which we introduced. Thus, a high-resolution computation of the Greeks using finite differences is possible as shown in Fig. 7. Due to the grid's adaptivity, even the computed second derivatives (gamma) are sufficiently smooth.

6. Concluding remarks

In this paper, we have introduced a spatially adaptive sparse grid approach for the solution of the Black–Scholes equation in computational finance. We described the FE discretization of the Black–Scholes equation and the technique of sparse grids. To be able to deal with option pricing problems in several dimensions and to resolve the non-smooth regions arising therein, we employed adaptive refinement. We discussed different strategies for adaptive refinement. The numerical results presented show that this enables one to reduce the number of grid points and, thus, degrees of freedom significantly while

maintaining the same accuracy. We showed that this allows us to successfully tackle problems in more dimensions than with non-adaptive discretizations.

Despite the good results that have already been obtained, further improvements can be achieved. Especially the adaptive refinement has to be studied further to fully understand the effects of different types of refinement strategies in higher dimensions. Furthermore, the adaptive sparse grid technique presented for the Black–Scholes equation is, of course, not limited to this equation, but can also be transferred to other parabolic PDE problems.

References

[1] P. Glasserman, Monte Carlo Methods in Financial Engineering, Springer, 2004.
[2] M.B. Giles, Multilevel Monte Carlo path simulation, Operations Research 56 (2008) 607–617.
[3] G. Linde, J. Persson, L. von Sydow, A highly accurate adaptive finite difference solver for the black-scholes equation, International Journal of Computer Mathematics 86 (2009) 2104–2121.
[4] M. Ehrhardt, R.E. Mickens, A fast, stable and accurate numerical method for the black-scholes equation of American options, International Journal of Theoretical and Applied Finance 11 (2008) 471–501.
[5] O. Pironneau, F. Hecht, Mesh adaptation for the black and scholes equations, East West Journal of Numerical Mathematics 9 (2000) 25–35.
[6] J. Toivanen, Numerical valuation of European and American options under Kou's jump–diffusion model, SIAM Journal on Scientific Computing 30 (2008) 1949–1970.
[7] C. Zenger, Sparse grids, in: W. Hackbusch (Ed.), Parallel Algorithms for Partial Differential Equations, in: Notes on Numerical Fluid Mechanics, vol. 31, Vieweg, 1991, pp. 241–251.
[8] H.-J. Bungartz, M. Griebel, Sparse grids, Acta Numerica 13 (2004) 147–269.
[9] C. Leentvaar, C. Oosterlee, On coordinate transformation and grid stretching for sparse grid pricing of basket options, Journal of Computational and Applied Mathematics 222 (2008) 193–209. (special issue: Numerical PDE Methods in Finance).
[10] C. Reisinger, G. Wittum, Efficient hierarchical approximation of high-dimensional option pricing problems, SIAM Journal on Scientific Computing 29 (2007) 440–458.
[11] D. Pflüger, Spatially Adaptive Sparse Grids for High-Dimensional Problems, Verlag Dr. Hut, München, ISBN: 978-3-86853-555-6, 2010.
[12] H.-J. Bungartz, A. Heinecke, D. Pflüger, S. Schraufstetter, Parallelizing a black-scholes solver based on finite elements and sparse grids, in: Proc. IEEE International Parallel & Distributed Processing Symposium, Atlanta, USA, 2010.
[13] Y. Achdou, O. Pironneau, Computational Methods for Option Pricing, Cambridge University Press, 2005.
[14] P. Wilmott, J. Dewynne, S. Howison, Option Pricing: Mathematical Models and Computations, Oxford Financial Press, 1994.
[15] R. Kangro, R. Nicolaides, Far field boundary conditions for black-scholes equations, SIAM Journal on Numerical Analysis 38 (2000) 1357–1368.
[16] R. Rannacher, Finite element solution of diffusion problems with irregular data, Numerische Mathematik 43 (1984) 309–327.
[17] H.-J. Bungartz, Dünne Gitter und deren Anwendung bei der adaptiven Lösung der dreidimensionalen Poisson-Gleichung, Dissertation, Fakultät für Informatik, Technische Universität München, 1992.
[18] J. Hull, Options, Futures, and Other Derivatives, 6th ed., Pearson Prentice Hall, Upper Saddle River, NJ, UA, 2006, pearson internat. ed. edition.
[19] A. Heinecke, D. Pflüger, Multi- and many-core data mining with adaptive sparse grids, in: Proceedings of the 8th ACM International Conference on Computing Frontiers, ACM, ISBN: 9781450306980, 2011, pp. 29:1–29:10.