Bergische Universität Wuppertal

Fakultät für Mathematik und Naturwissenschaften

Institute of Mathematical Modelling, Analysis and Computational Mathematics (IMACM)

Preprint BUW-IMACM 21/18

Anna Clevenhaus, Matthias Ehrhardt and Michael Günther

A parallel Sparse Grid Combination Technique using the Parareal Algorithm

June 22, 2021

http://www.imacm.uni-wuppertal.de

A parallel Sparse Grid Combination Technique using the Parareal Algorithm

Anna Clevenhaus, Matthias Ehrhardt, Michael Günther

Bergische Universität Wuppertal, School of Mathematics and Natural Scienes, IMACM,
Gaußstraße 20, 42119 Wuppertal, Germany

Abstract

The sparse grid combination technique is a well-known method for reducing the dimension of higher-dimensional problems. In this work, we combine this technique with a parallel-in-time algorithm, the Parareal algorithm, to reduce the computational cost in the time dimension as well. In addition, we take advantage of the combination technique and the Parareal algorithm to further reduce the computational cost and hence the run time. Here, we focus on one financial application, namely American option pricing with the Heston model, and use the modified Craig-Sneyd scheme as temporal solver.

Keywords: American Option, Sparse grid, Parareal Algorithm, Parallel Computing

1. Introduction

As the complexity of the models grows, new algorithms must be developed to reduce the computational cost. In the spatial dimension, we observe an exponential growth of grid points with increasing dimension, since a complete tensor-based grid contains $\mathcal{O}(N^d)$ grid nodes ('curse of dimensionality'). The sparse grid combination method is able to reduce the number of grid points because it combines certain sparse grid combinations with the highest information gain, thus breaking the curse. Therefore, it is often used to reduce the computational cost in the spatial dimension. Smolyak [1] developed the sparse grid for numerical integration, later Bungartz and Griebel [2], Zenger [3] and Schiekofer [4] extended the idea for the solution of partial differential equations. In addition, we use a parallel-in-time algorithm to further reduce the complexity. The algorithm chosen is the Parareal algorithm developed by Lions, Mayday and Turinici [5]. The algorithm consists of a parallel computation with a fine solver, followed by a serial correction step with a coarse solver, and can be considered either a multiple shooting or a multigrid method. By combining the sparse grids and the Parareal algorithm, we can apply improvement strategies. One strategy is to parallelize the computation of the sparse grids in the serial part of the algorithm. The other idea reduces the computational cost by reusing intermediate results from the fine or coarse

Email address: `{clevenhaus, ehrhardt, guenther}@uni-wuppertal.de` (Anna Clevenhaus, Matthias Ehrhardt, Michael Günther)

solver for the other solver. To verify our theoretical results, we use a financial application. We price an American option with the Heston model using an ADI scheme as the temporal solver and compare the accuracy and running time of the different methods in the numerical results. This paper begins by introducing the sparse grid combination technique as a spatial discretization technique that breaks the curse of dimensionality. The Parareal algorithm is then presented, along with improvement ideas to further reduce computational effort, this time in the temporal dimension, and the resulting theoretical speed advantages. Section 4 focuses on financial applications, namely American option pricing under the Heston model. Numerical results for both accuracy and running time are shown in Section 5. Finally, the paper closes with a conclusion and a brief outlook to extend the improvement of the Parareal algorithm combined with the sparse grid combination technique to higher dimensional models.

2. The Sparse Grid Combination Technique

As the number of dimensions of a model growth to obtain a better accuracy the number of grid points and the computational effort growth as well, unfortunately the growth of grid points is exponential. The sparse grid approach was introduced as spatial discretization for PDEs to break the course of the dimensional complexity growth from the spatial point of view. Since we consider a two dimensional financial application, we only present the two dimensional sparse grid combination technique following the approach of Reisinger [6]. In the continuous setting, let Ω_2 be a two dimensional domain, with $\mathbf{x} = (x_1, x_2) \in \Omega_2$. We define a tensor based grid on $\Omega_{(l_1,l_2)}$ with grid nodes

$$\mathbf{x} = (x_{l_1,j_1}, x_{l_2,j_2}) \quad \text{for} \quad j_1 = 0, 1, \dots, 2^{l_1} \quad \text{and} \quad j_2 = 0, \dots, 2^{l_2},$$

where $\mathbf{l} = (l_1, l_2)$ denotes a specific grid and (j_1, j_2) the specific (x_1, x_2) coordinates on $\Omega_{\mathbf{l}}$. We set $\Omega_2 = [0,1]^2$ as all grids can be transformed to the unit square, in Section 4 we present a transformation satisfying the special needs of the financial application. Due to construction of the sparse grids the mesh size of the discrete solution $u_{\mathbf{l}}$ on the grid $\Omega_{\mathbf{l}}$ is $h = (2^{-l_1}, 2^{-l_2})$. We consider the error splitting

$$u - u_{\mathbf{l}} = h_1^2 w_1(h_1) + h_2^2 w_2(h_2) + h_1^2 h_2^2 w_{1,2}(h_1, h_2),$$

where w_1 only depends on h_1, w_2 only on h_2 and h_1 and h_2 are independent from each other. Each of w_1, w_2, $w_{1,2}$ is bounded. Next, a *hierarchical surplus* is defined as

$$\delta(u_{\mathbf{l}}) = u_{\mathbf{l}} - u_{\mathbf{l}-e_1} - u_{\mathbf{l}-e_2} - u_{\mathbf{l}-e_1-e_2}, \quad e_1 = (1,0)^\top, \quad e_2 = (0,1)^\top.$$

Combining the error splitting and the hierarchical surplus, we obtain the expansion

$$\delta(u-u_{\mathbf{l}}) = h_1^2h_2^2w_{1,2}(h_1,h_2) - 4h_1^2h_2^2w_{1,2}(2h_1,h_2) - 4h_1^2h_2^2w_{1,2}(h_1,2h_2) \\ +16h_1^2h_2^2w_{1,2}(2h_1,2h_2) = \mathcal{O}(2^{-2|\mathbf{l}|_1}).$$

The combination technique is motivated by the aim to get the highest information gain from the sub-solutions $u_{\mathbf{l}}$ with the high information gain, being similar to a high surplus. Easily spoken, grids with the same number of grid points have the same information gain since $|\mathbf{l}|_1$ is equal for those. Therefore the combined sparse grid solution is given as the sum of all surpluses with $|\mathbf{l}|_1 \leq n$ and reduces in the two dimensional case to

$$u_n^s = \sum_{|\mathbf{l}|_1\leq n} \delta u_{\mathbf{l}} = \sum_{q=0}^{n}\Bigg(\sum_{|\mathbf{l}|_1=q} u_{\mathbf{l}} - 2\sum_{|\mathbf{l}|_1=q-1,q\geq 0} u_{\mathbf{l}} + \sum_{|\mathbf{l}|_1=q-2,q\geq 0} u_{\mathbf{l}}\Bigg) = \sum_{|\mathbf{l}|_1=n} u_{\mathbf{l}} - \sum_{|\mathbf{l}|_1=n-1} u_{\mathbf{l}}. \tag{1}$$

We observe a reduced growth of grid points of $\mathcal{O}(h^{-1}\log_2(h^{-1}))$ being determined by the number of grid points on each sub-grid $\mathcal{O}(2^n)$ multiplied by the increase of the number of grids $\mathcal{O}(n)$. Sensitivity to highly disordered grids are avoided, as we set a minimum mesh width in each dimension in our experiments with $(l_1, l_2) \geq (3,3)$, s.t. we have at least 9 grid points. By incorporating the surpluses of all sub-solutions with $|l|_1 < n$

$$||u_n^s - u|| \leq \mathcal{O}\big(h^2\log_2(h^{-1})\big).$$

3. The Parareal Algorithm

The first step of the Parareal algorithm is to divide the time interval $[0,T]$ into N_T equal cells $[t_i, t_{i+1}]$, s.t. $t_n = t_0 + n\cdot\Delta T$, with $\Delta t = \frac{T}{N_T}$, $n = 0,\dots,N_T$. There are two different temporal solvers, $\mathcal{G}$ and $\mathcal{F}$, in the iteration procedure of the time-parallel algorithm. It is assumed that both solvers are convergent and stable for the chosen step size. Within the Parareal algorithm, $\mathcal{G}$ is the bottleneck for the speedup and convergence rate, since this solver is of lower order than $\mathcal{F}$. Moreover, it is used in the serial correction step of the iterative procedure. In our case, the solvers differ only in the underlying grid on which the solution is computed. Since we consider a sparse grid structure, the solver $\mathcal{F}$ is defined as a temporal integrator for u_n^s with $N_{\mathcal{F}}$ time steps for each time slice and $\mathcal{G}$ for u_{n-1}^s with $N_{\mathcal{G}}$ time steps. Let u_i^k denote the discrete solution of u_n^s at the $k-$th iteration and t_i-th time step. Algorithm 1 introduces the Parareal algorithm. Besides the Parareal algorithm, we apply other ideas to improve the Parareal algorithm by exploiting the special properties of the combination technique. The first idea is based on the sparse grid combination technique. The other idea is to optimize the computation of the bottleneck solver $\mathcal{G}$ by parallelizing the computation of the sparse grids. In the case of a 2d problem, the combination solution of the fine solver is given by

$$u^s_{n_\mathcal{F}} = \sum_{|\mathbf{l}|_1=n_\mathcal{F}} u_\mathbf{l} - \boxed{\sum_{|\mathbf{l}|_1=n_\mathcal{F}-1} u_\mathbf{l}}$$

and for the coarse solver we obtain

$$u^s_{n_\mathcal{G}} = \sum_{|\mathbf{l}|_1=n_\mathcal{G}} u_\mathbf{l} - \sum_{|\mathbf{l}|_1=n_\mathcal{G}-1} u_\mathbf{l} = \boxed{\sum_{|\mathbf{l}|_1=n_\mathcal{F}-1} u_\mathbf{l}} - \sum_{|\mathbf{l}|_1=n_\mathcal{F}-2} u_\mathbf{l}$$

Therefore, both solvers compute the solution on the same grids, namely the grids on level $n_\mathcal{F} - 1$ and $n_\mathcal{G}$, respectively. To save computational effort, we use the already computed solution of the respective level from the other solver. Either we use the solution from the coarse solver when computing the fine grid solution, see Algorithm 2, or vice versa, see Algorithm 3. The communication time in Algorithm 2 can be reduced by communicating only $\sum_{|\mathbf{l}|_1=n_G} u_\mathbf{l}$ instead of all subgrids with $u_{|\mathbf{l}|_1=n_G}$ separately. For the Algorithm 3, this is only advantageous if the computation of the grids is not also done in parallel, since the communication arises either way due to parallelism.

Algorithm 1 Parareal Algorithm

```
$u_0^k = u_0^0$ given by initial condition
*Compute initial values for each time interval:*
**for** $i = 0 : N_T - 1$ **do**
    $u_{i+1}^0 = \mathcal{G}(u_i^0, t_i, t_{i+1})$
**end for**
*Parareal-Algorithm*
k=0
**while** $k <$ Iteration **do**
    *Parallel Approximation*
    **for** $i = k : N_T - 1$ **do**
        $\tilde{u}_{i+1}^k = \mathcal{F}(u_i^k, t_i, t_{i+1})$
    **end for**
    *Serial Update*
    **for** $i = k : N_T - 1$ **do**
        $u_{i+1}^{k+1} = \mathcal{G}(u_i^{k+1}, t_i, t_{i+1})$
        $u_{i+1}^{k+1} = u_{i+1}^{k+1} + \tilde{u}_{i+1}^k - u_{i+1}^k$
    **end for**
    k=k+1
**end while**
```

3.1. Speed up Parareal

In this section, we analyze the theoretical speedup for the different methods. In the analysis, we neglect the initialization time and compare the results with the theoretical results about the speedup of the Parareal algorithm without communication cost [7]. We assume that the computation time for all sparse

Algorithm 2 Parareal Algorithm with usage of the computation of $|\mathbf{l}|_1 = n_F - 1$ from the fine solver.

```
$u_0^k = u_0^0$ given by initial condition
*Compute initial values for each time interval:*
**for** $i = 0 : N_T - 1$ **do**
    $u_{i+1}^0 = \mathcal{G}(u_i^0, t_i, t_{i+1})$
**end for**
*Parareal-Algorithm*
k=0
**while** $k <$ Iteration **do**
    *Parallel Approximation*
    **for** $i = k : N_T - 1$ **do**
        $\tilde{u}_{i+1}^k, \boxed{u_{n_F-1,i+1}^k} = \mathcal{F}(u_i^k, t_i, t_{i+1})$
    **end for**
    *Serial Update*
    **for** $i = k : N_T - 1$ **do**
        $u_{i+1}^{k+1} = \tilde{\mathcal{G}}(u_i^{k+1}, \boxed{u_{n_F-1,i+1}^k}, t_i, t_{i+1})$
        $u_{i+1}^{k+1} = u_{i+1}^{k+1} + \tilde{u}_{i+1}^k - u_{i+1}^k$
    **end for**
    k=k+1
**end while**
```

Algorithm 3 Parareal Algorithm with usage of the computation of $|\mathbf{l}|_1 = n_G$ from the coarse solver.

```
$u_0^k = u_0^0$ given by initial condition
*Compute initial values for each time interval:*
**for** $i = 0 : N_T - 1$ **do**
    $u_{i+1}^0, \boxed{u_{n_G,i+1}^0} = \mathcal{G}(u_i^0, t_i, t_{i+1})$
**end for**
*Parareal-Algorithm*
k=0
**while** $k <$ Iteration **do**
    *Parallel Approximation*
    **for** $i = k : N_T - 1$ **do**
        $\tilde{u}_{i+1}^k = \tilde{\mathcal{F}}(u_i^k, \boxed{u_{n_G,i}^k}, t_i, t_{i+1})$
    **end for**
    *Serial Update*
    **for** $i = k : N_T - 1$ **do**
        $u_{i+1}^{k+1}, \boxed{u_{n_G,i+1}^{k+1}} = \mathcal{G}(u_i^{k+1}, t_i, t_{i+1})$
        $u_{i+1}^{k+1} = u_{i+1}^{k+1} + \tilde{u}_{i+1}^k - u_{i+1}^k$
    **end for**
    k=k+1
**end while**
```

grids of the same level n is the same for one time step and is determined by c_n. Since the difference between the fine and coarse solvers is given only by the underlying grid, the computation time for a sparse grid level n is the same for the coarse and fine solvers.

Let the computation time of a sparse grid combination technique for level n for one time step and N_T time steps be denoted by

$$c(n, N_T) = N_T \cdot c(n), \text{ with } c(n) = (n+1) \cdot c_n + n \cdot c_{n-1}$$

The computational time for the Parareal Algorithm is given by

$$c(n, N_T, N_{\mathcal{G}}, P, k) = (k+1)PN_{\mathcal{G}}c(n-1) + k\frac{N_T}{P}c(n),$$

with $N_{\mathcal{F}} = \frac{N_T}{P}$. We obtain the general speedup without communication costs by

$$\begin{aligned} \frac{c(n, N_T)}{c(n, N_T, N_{\mathcal{G}}, P, k)} &= \frac{PN_F c(n)}{(k+1)PN_{\mathcal{G}}c(n-1) + kN_F c(n)} \\ &= \frac{1}{(k+1)\frac{N_{\mathcal{G}}c(n-1)}{N_{\mathcal{F}}c(n)} + \frac{k}{P}} \\ &\leq \min\left\{\frac{N_{\mathcal{F}}c(n)}{N_{\mathcal{G}}c(n-1)}, \frac{P}{k}\right\} \end{aligned}$$

Now we include the communication time. Since parallel computations involve additional expensive communications, communication is the bottleneck of parallelism. The communication time can be described by

$$c^{\text{com}}(l) = \alpha^{\text{com}} + \beta^{\text{com}} \cdot l,$$

where α^{com} denotes the initialization time for parallelism, β^{com} denotes the communication cost per communicated length of the message, and l denotes the length of the bytes of the message. Both α^{com} and β^{com} are predefined by the computer architecture and are constant. Since each grid in the sparse grid level n consists of 2^n grid points stored as floating point 32 numbers, we can specify $l = 2^n \cdot 32 = 2^{n+5}$. The communication time is minimized since we are communicating only the sum of the subgrids, rather than each subgrid solution. Let N_T be fixed, we search for P processors that minimize the computation time for the Parareal algorithm with communication

$$\min_{P \in \mathbb{N}^+} c^c(n, N_T, N_{\mathcal{G}}, P, k) = \min(k+1)PN_{\mathcal{G}}c(n-1) + k\frac{N_T}{P}c(n) + kPc^{\text{com}}(n).$$

	$c(n)$	$c(n-1)$
Original	$(n+1)\cdot c_n + n\cdot c_{n-1}$	$n\cdot c_{n-1} + (n-1)\cdot c_{n-2}$
Fine solve	$(n+1)\cdot c_n + n\cdot c_{n-1}$	$(n-1)\cdot c_{n-2}$
Coarse solver	$(n+1)\cdot c_n$	$n\cdot c_{n-1} + (n-1)\cdot c_{n-2}$

Table 1: Changes in the computational time for the sparse grid in the Parareal algorithm and the improved algorithms.

The result has to be a unique positive integer solution. The continuous optimal solution is given by

$$P^* = \sqrt{\frac{kN_T c(n)}{(k+1)N_\mathcal{G} c(n-1) + kc^{\mathrm{com}}(n)}}, \tag{2}$$

the optimal integer solution is either $P = \lfloor P^* \rfloor$ or $P = \lceil P^* \rceil$. Using the continuous solution a lower bound for the computational time can be derived

$$c^c(n, N_T, N_\mathcal{G}, P, k) > 2\sqrt{kN_T c(n)\cdot((k+1)N_\mathcal{G} c(n-1) + kc^{\mathrm{com}}(n)}$$

and therefore an upper bound for the speedup

$$\begin{aligned} Sp^c &< \frac{N_T c(n)}{2\sqrt{kN_T c(n)\cdot((k+1)PN_\mathcal{G} c(n-1) + kc^{\mathrm{com}}(n))}} \\ &= \frac{1}{2}\sqrt{\frac{N_T c(n)}{(k^2+k)PN_\mathcal{G} c(n-1) + k^2 c^{\mathrm{com}}(n)}} \\ &\leq \min\left\{\sqrt{\frac{N_T c(n)}{c^{\mathrm{com}}(n)}}, \sqrt{\frac{N_\mathcal{F} c(n)}{N_\mathcal{G} c(n-1)}}\right\} \end{aligned} \tag{3}$$

Looking further into the components of the upper bounds depending on the sparse grid level

$$\lim_{n\to+\infty} \sqrt{\frac{N_T c(n)}{c^{\mathrm{com}}(n)}} = \lim_{n\to+\infty} \sqrt{\frac{N_T\left((n+1)c_n + nc_{n-1}\right)}{\alpha^{\mathrm{com}} + \beta^{\mathrm{com}}\cdot 2^{n+5}}} = 0 \tag{4}$$

and

$$\lim_{n\to+\infty} \sqrt{\frac{N_\mathcal{F} c(n)}{N_\mathcal{G} c(n-1)}} = \lim_{n\to+\infty} \sqrt{\frac{N_\mathcal{F}\left((n+1)c_n + nc_{n-1}\right)}{N_\mathcal{G}\left(nc_{n-1} + (n-1)c_{n-2}\right)}} = \sqrt{\frac{N_\mathcal{F}}{N_\mathcal{G}}} \tag{5}$$

we have shown in this theoretical result that the communication time is the real bottleneck of the algorithm, since the communication time increases faster than the computation time of the sparse grids. In the next approach, we consider the Parareal algorithm with a parallel computation of the sparse grids within the serial computation of the coarse solver. Since the communication time does not change with a serial computation due to the parallelization, the ideal speedup does not change either. Our final improvements involve the intermediate results of the fine or coarse solver, s.t. the computation time changes as shown in Table 1. Inserting the intermediate results from the fine solver, we obtain for the speedup the equation 4 as well and equation 5 changes to

$$\lim_{n\to+\infty}\sqrt{\frac{N_{\mathcal{F}}c(n)}{N_{\mathcal{G}}c(n-1)}}=\sqrt{\frac{N_{\mathcal{F}}\left((n+1)c_n+nc_{n-1}\right)}{N_{\mathcal{G}}\left((n-1)c_{n-2}\right)}},$$

s.t. we gain a larger speedup than before under the condition that the communication cost is low enough. When using the coarse solver, the speedup for both components of the speedup 3 grows, since the serial computation of the fine solver and the parallel part of the Parareal algorithm do not shorten, and therefore we obtain a larger prefactor than before.

4. Application to pricing American options under the Heston model

From the American option problem a *linear complementarity problem (LCP)*

$$\begin{cases}\big(P-\phi(S)\big)\cdot(\frac{\partial P}{\partial \tau}-\mathcal{L}_H[P]) & =0,\\ -(\frac{\partial P}{\partial \tau}-\mathcal{L}_H[P]) & \geq 0,\\ P-\phi(S) & \geq 0.\end{cases}$$

arises [8]. For the pricing of an American put option P we seek for the solution P and the associated free boundary S_f, i.e. the tuple $\big(P(S,t),S_f(t)\big)$ such that

$$\begin{aligned}P(S,t)&=\phi(S)\quad\text{for}\quad S\leq S_f(t),\\ P(S,t)&>\phi(S)\quad\text{for}\quad S>S_f(t),\end{aligned}$$

where S denotes the price of an asset at time t with $0\leq t\leq T$. The terminal condition at the expiry date $t=T$ reads

$$P(S,\nu,T)=\phi(S),\qquad S>S_f(T).$$

In order to solve a forward-in-time PDE, we utilize the time reversal $\tau=T-t$ and the differential operator has to fulfill the inequality. The differential operator for the fair price of an American put option $P(S,\nu,t)$ under risk neutral measure for the Heston model is given by

$$\mathcal{L}_H[P]=\frac{1}{2}\nu S^2\frac{\partial^2 P}{\partial S^2}+\rho_{S\nu}\sigma_\nu S\nu\frac{\partial^2 P}{\partial S\partial\nu}+\frac{1}{2}\sigma_\nu^2\nu\frac{\partial^2 P}{\partial\nu^2}+rS\frac{\partial P}{\partial S}+\kappa_\nu(\nu-\mu_\nu)\frac{\partial P}{\partial\nu}-rP,$$

where $\nu>0$ is the square of the volatility of the underlying, κ_ν is the mean-reversion rate and μ_ν is the long-term mean of the volatility ν and σ_ν is the volatility-of-variance. The correlation between S and ν is determined by a constant parameter $\rho \in [-1,1]$ [9]. If the Feller condition $2\kappa_\nu \mu_\nu > \sigma_\nu$ is fulfilled, $\nu > 0$ applies. The reformulation with an operator splitting of the LCP with an auxiliary variable λ is given by

$$\begin{cases} \mathcal{L}_H[P] - \frac{\partial P}{\partial \tau} = \lambda, \\ \lambda \geq 0, \quad P - \phi(S) \quad \geq 0, \quad (P - \phi(S))\,\lambda = 0, \end{cases}$$

for $(S, \nu, \tau) \in \Omega \times [0, T]$ with the initial and boundary conditions [10]. It results in a mixed formulation of the LCP problem, where λ plays the role of a Lagrange multiplier. The advantage of the LCP formulation of American Option problems is that an explicit computation for the free boundary value $S_f(\tau)$ is avoided. The spatial boundary conditions for S are given by

$$P(0, \nu, \tau) = K, \quad \lim_{S \to \infty} P(S, \nu, \tau) = 0, \qquad 0 \leq \tau \leq T,$$

for the boundaries $\nu = 0$ and $\nu = \nu_{\max}$, the equation $\frac{\partial P}{\partial \tau} - \mathcal{L}_H[P]$ has to be fulfilled. The 'spatial' boundary conditions at $S = S_f(\tau)$, $S \to \infty$ are given by

$$P\big(S_f(\tau), \nu_f(\tau), \tau\big) = \phi(S_f(\tau)), \qquad \frac{\partial P}{\partial S}\big(S_f(\tau), \nu_f(\tau), \tau\big) = -1, \qquad 0 \leq \tau \leq T,$$

An efficient computation of the sparse grid combination technique requires the transformation of the problem to the $\Omega_2 = [0,1]^2$ space. Therefore, the transformation $\hat{S} = \ln(S/K)$, which is usually applied in finance, is not used. To reconstruct a nonuniform grid for the spatial variables S and ν from the uniform sparse grids on Ω_2 with $\mathbf{x}$, we consider a smooth transformation function for each dimension.

$$y = \psi(x)$$

on $[x_{\min}, x_{\max}] = [0,1]$ to the the arbitrary interval $[y_{\min}, y_{\max}]$ with $y_{\min}, y_{\max} \in \mathbb{R}$ and $y_{\min} < y_0 < y_{\max}$

$$y = y_0 + \alpha \cdot \sinh(x \cdot (c_2 - c_1) + c_1), \quad c_1 = \sinh^{-1}\left(\frac{y_{\min} - y_0}{\alpha}\right), \quad c_2 = \sinh^{-1}\left(\frac{y_{\max} - y_0}{\alpha}\right).$$

Small α values lead to highly non-uniform grid in y, while large values of α lead to a uniform distribution of grid points [11, 12]. For the transformation for the asset S a common choice is $S_{\min} = 0$, $S_{\max} = 3\,K$ and $S_0 = K$. Since the initial condition processes a discontinuous derivative at K, we have to smooth the initial data. The operator for the transformed Heston PDE reads

$$\begin{aligned} \mathcal{L}[u] = \frac{1}{2}\nu S^2 a_S^2 \frac{\partial^2 u}{\partial {x_1}^2} + \big(rSa_S + \frac{1}{2}\nu S^2 b_S\big)\frac{\partial u}{\partial x_1} + \rho a_S a_\nu \sigma S \nu \frac{\partial^2 u}{\partial x_1 \partial x_2} \\ + \frac{1}{2}\sigma^2 \nu a_\nu^2 \frac{\partial^2 u}{\partial {x_2}^2} + \big(\kappa(\nu - \mu)a_\nu + \frac{1}{2}\sigma^2 \nu b_\nu\big)\frac{\partial u}{\partial x_2}, \end{aligned}$$

where $a_S = \frac{\partial\psi^{-1}(x_1)}{\partial S}$, $b_S = \frac{\partial^2\psi^{-1}(x_2)}{\partial S^2}$ and a_ν and b_ν analogously.

We use second-order finite differences to approximate the spatial derivatives. Since we use sparse grids, we solve the Heston PDE on several different grids using the same spatial approximations for the derivatives, namely central difference quotients of order two in each direction. In addition, we consider the forward and backward second-order difference quotients at the boundaries. Note that the mixed derivative at the boundaries is zero for $z = 0$, as is the diffusion term, so it is treated trivially. An improvement in accuracy can be achieved by using higher order stencils within finite difference methods (FDM) [13] or by considering spectral methods [14]. The spatial discretization leads to an approximation of the option value $P(x_1, x_2, \tau)$ at the spatial grid points $(x_1, x_2) \in [0, 1]$ for an LCP, analogous to, for the transformed operator $\mathcal{L}$. For $0 < \tau < T$, the solution vector $P(\tau)$ of the semi-discrete *partial differential complementarity problem* (PDCP) is

$$\frac{\partial P}{\partial \tau} = FP(\tau) + \lambda(\tau), \quad P(\tau) \geq \phi\big(\psi^{-1}(x_1)\big), \quad \Big(P(\tau) - \phi\big(\psi^{-1}(x_1)\big)\Big)^\top \lambda(\tau) = 0,$$

gives an approximations for $P(x_1, x_2, \tau)$. The inequalities are component-wise and F is a given real matrix, where $\phi(\psi^{-1}(x_1))$ is the initial condition. We discretize every time slice uniformly by $\Delta_\tau = (\tau - \tau)/N_{\mathcal{F}}$. The time points within one slice are $\tilde{\tau}_j = \tau_n + j \cdot \Delta_\tau$ with $j = 0, \ldots, N_{\mathcal{F}}$. Let u^j describe the discrete solution at time step τ_j and g the discrete payoff value, the fully *discrete linear complementarity problem* (DLCP) reads cf. [10]

$$\begin{cases} u^{j+1} = Au^j + \Delta_\tau \lambda^j, \\ \lambda^{j+1} \geq 0, u^{j+1} \geq g, (\lambda^{j+1})^\top (u^{j+1} - g). \end{cases} \tag{DLCPλ}$$

In the first step, we solve a system of linear equations and in the second one a variable update is done. The system of equations is solved by the modified Craig-Sneyd scheme with the additional parameter λ, cf. [15]. The additional parameter only accrues within the first equation

$$Y_0 = u^j + \Delta_\tau F(\tau^j, u^j) \boxed{+\Delta_\tau \lambda^j},$$

the rest of the scheme remains the same. We use an improved way of implementation of the ADI schemes [16]. The second step, the variable update can be done component-wise by applying

$$\begin{cases} u^{j+1} &= \max(\tilde{u}^{j+1} - \Delta_\tau \lambda^j, u^0), \\ \lambda^{j+1} &= \max(0, \lambda^j + (u^0 - \tilde{u}^{j+1})/\Delta_\tau) \end{cases}$$

Since we have an initial condition, we set λ^0 as the zero vector.

5. Numerical Results

We use a common parameter set for the American Put Option problem [17, 18, 19]

$$T = 0.25, \quad K = 10, \quad \kappa = 5, \quad \mu = 0.16, \quad \sigma_\nu = 0.9, \quad \rho = 0.1, \quad r = 0.1.$$

We set the sparse grid level to $n = |\mathbf{l}|_1 = 13$ with $l_{\min} = 3$ to obtain a large computational effort to show the effects of our ideas. For the grid transformation, we set $S_{\min} = 0$, $S_0 = 10$ and $S_{\max} = 3K$ as well as $\nu_{\min} = 0$, $\nu_0 = 0.0625$ and $\nu_{\max} = 3$. Due to our choice of $\alpha_S = 2$ and $\alpha_\nu = 2$, we obtain a highly nonuniform grid. The time parameters for the Parareal Algorithm are $N_T = 1200$ and $N_{\mathcal{G}} = 25$. From equation 2 we get for 1, 2 and 3 iterations the following optimal number of processors

$$k = 1 : P^* \approx 2.44, \quad k = 2 : P^* \approx 2.82, \quad k = 3 : P^* \approx 2.99$$

using $\beta^{\text{com}} = 8\ GT/s$, $c(13) = 0.22\ s$, $c(12) = 0.07\ s$ and $c(11) = 0.02\ s$. Beneath the optimal number of processors, we test other numbers of processors to qualify the theoretical results. Table 2 shows the accuracy results for the original Parareal Algorithm, denoted by "Original", and the improved algorithms using the intermediate results of either the fine or coarse solver, denoted by "Fine solver" and "Coarse solver" respectively. The accuracy is determined by the mean square error (MSE)

$$MSE = \frac{1}{N}\sqrt{\sum_{i=0}^{N_1}\sum_{j=0}^{N_2}\left(P(S_i,\nu_j) - \tilde{P}(S_i,\nu_j)\right)^2},$$

where N is the total number of elements of the solution and P the exact solution is given by the sparse grid solution resulting by using the modified Craig-Sneyd scheme and $\tilde{P}$ the approximated solution, s.t. the results are not effected by either the underlying grid structure nor by the temporal solver. The accuracy results show that the usage of the intermediate results of the fine solver increases the accuracy in comparison to the original algorithm independently from the number of processors and the number of iterations. Whereas the accuracy results from the usage of the coarse solver is in the same accuracy range of the Parareal Algorithm and highly depends on the number of processors and the iteration count. This behaviour results due to the incorporation of less accurate results into the result of the fine solver in each iteration. As the parallelism of the sparse grid computation in the coarse solver does not effect the accuracy, the accuracy with and without this parallelism is the same and therefore not listed in the table.

Table 3 shows the run times obtained by a benchmark time function. From the run time results, we obtain that the usage of the coarse intermediate results reduce the run time significantly for a small amount of iterations. The usage of the fine results only reduce the run time for a small iteration count in comparison to the number of processors. The additional usage of parallelism in the computation of the sparse grid

		MSE		
Processors	Iterations	Original	Fine solver	Coarse solver
2	1	6.6789E-07	1.2412E-07	7.7833E-07
3	1	5.6220E-07	9.7435E-08	3.8067E-07
3	2	3.3258E-07	6.1225E-08	1.0322E-06
4	1	4.6047E-07	7.4942E-08	2.2978E-07
4	2	3.3196E-07	5.6706E-08	6.1924E-07
4	3	2.0428E-07	4.0458E-08	8.9811E-07

Table 2: Accuracy results of the Parareal Algorithm and its improvements with the serial computation of the solution on a sparse grid.

		Run time(s)					
		Parareal			Parareal and Sparse Parallism		
Processors	Iterations	Original	Fine solver	Coarse solver	Original	Fine solver	Coarse solver
2	1	258.249	269.187	326.535	417.688	307.761	281.320
3	1	192.820	220.193	164.886	273.177	285.214	163.901
3	2	355.484	543.789	277.481	425.464	465.500	379.796
4	1	202.859	196.057	188.723	213.585	223.042	176.093
4	2	405.618	385.204	321.038	351.261	332.840	271.047
4	3	415.204	415.362	416.229	485.966	404.208	370.041

Table 3: Run times in seconds of the Parareal Algorithm and its improvements with and without the additional parallelism of the serial computation of the coarse solver.

results in the coarse solver is only feasible in combination of a large number of processors and a relatively high number of iterations. All results are computed on a Intel(R) Core(TM) i7-8700K CPU @ 3.70 GHz using the programming language Julia.

6. Conclusion

The numerical results show that for a high number of processors and a high number of iterations, using the intermediate results of the fine solver and parallelizing the calculation of the sparse grids is recommended for both accuracy and run time. For small iteration numbers, the coarse solver is practical in terms of running time. Overall, all ideas improve the original Parareal algorithm, which idea should be used depends on the application. In future research, an extension will be a combination between multidimensional problems with the MGRID approach.

References

[1] S. A. Smolyak, Quadrature and interpolation formulas for tensor products of certain classes of functions, Dokl. Akad. Nauk SSSR 148 (5) (1963) 1042–1045.

[2] H.-J. Bungartz, M. Griebel, Sparse grids, Acta Numerica 13 (2004) 147–269.

[3] C. Zenger, Sparse grids, Vol. Nr. 90,18 of SFB-Bericht / Sonderforschungsbereich 342, Institut für Informatik, Technische Universität München : Reihe A, SFB 342, München, 1990.

[4] T. Schiekofer, Die Methode der finiten Differenzen auf dünnen Gittern zur Lösung elliptischer und parabolischer partieller Differentialgleichungen, Dissertation, Universität Bonn (1998).

[5] J.-L. Lions, Y. Maday, G. Turinici, Résolution d'EDP par un schéma en temps «pararéel», Compt. Rend. Acad. Sci. - Ser. I - Math. 332 (7) (2001) 661–668.

[6] C. Reisinger, Numerische Methoden für hochdimensionale parabolische Gleichungen am Beispiel von Optionspreisaufgaben, Dissertation, Universität Heidelberg (2004).

[7] M. J. Gander, S. Vandewalle, Analysis of the parareal time–parallel time–integration method, SIAM J. Sci. Comput. 29 (2) (2007) 556–578.

[8] M. Günther, A. Jüngel, Finanzderivate mit MATLAB: Mathematische Modellierung und numerische Simulation, 2nd Edition, Vieweg+Teubner Verlag, Wiesbaden, 2010.

[9] S. L. Heston, A closed-form solution for options with stochastic volatility with applications to bond and currency options, Rev. Fin. Stud. 6 (2) (1993) 327–343.

[10] S. Ikonen, J. Toivanen, Operator splitting methods for pricing American options under stochastic volatility, Numerische Mathematik 113 (2) (2009) 299–324.

[11] K. J. in 't Hout, B. D. Welfert, Unconditional stability of second-order ADI schemes applied to multi-dimensional diffusion equations with mixed derivative terms, Appl. Numer. Math. 59 (3-4) (2009) 677–692.

[12] D. Tavella, C. Randall, Pricing financial instruments: The finite difference method, Wiley, New York, NY, 2000.

[13] C. Hendricks, High-order methods for parabolic equations in multiple space dimensions for option pricing problems, Dissertation, Bergische Universität Wuppertal (2017).

[14] C. Hendricks, M. Ehrhardt, M. Günther, Hybrid finite-difference/pseudospectral methods for the Heston and Heston–Hull–White partial differential equations, J. Comput. Fin. 21 (5) (2018) 1–33.

[15] K. J. in 't Hout, S. Foulon, ADI finite difference schemes for option pricing in the Heston model with correlation, Int. J. Numer. Anal. Mod. 7 (2).

[16] L. Teng, A. Clevenhaus, Accelerated implementation of the ADI schemes for the Heston model with stochastic correlation, J. Comput. Sci. 36 (2019) 101022.

[17] N. Clarke, K. Parrott, Multigrid for American option pricing with stochastic volatility, Appl. Math. Fin. 6 (3) (1999) 177–195.

[18] T. Haentjens, K. J. in 't Hout, ADI schemes for pricing American options under the heston model, Appl. Math. Fin. 22 (3) (2015) 207–237.

[19] C. W. Oosterlee, On multigrid for linear complementarity problems with application to American–style options, Electron. Trans. Numer. Anal. (2003) 165–185.