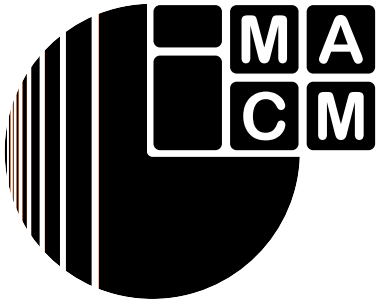

Bergische Universität Wuppertal

Fachbereich Mathematik und Naturwissenschaften

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

Preprint BUW-IMACM 16/14

Igor Kossaczký, Matthias Ehrhardt and Michael Günther

Modifications of the PCPT Method for HJB Equations

July 22, 2016

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

Modifications of the PCPT Method for HJB Equations

Igor Kossaczký[a),b)], Matthias Ehrhardt[c)] and Michael Günther[d)]

Lehrstuhl für Angewandte Mathematik und Numerische Analysis, Fakultät für Mathematik und Naturwissenschaften, Bergische Universität Wuppertal, Gaußstrasse 20, 42119 Wuppertal, Germany

[a)]Corresponding author: kossaczky@math.uni-wuppertal.de
[b)]URL: http://www-num.math.uni-wuppertal.de/en/amna/people/igor-kossaczky-msc.html
[c)]ehrhardt@math.uni-wuppertal.de
[d)]guenther@math.uni-wuppertal.de

Abstract. In this paper we will revisit the modification of the piecewise constant policy timestepping (PCPT) for solving Hamilton-Jacobi-Bellman (HJB) equations. This modification is called piecewise predicted policy timestepping (PPPT) method and if properly used, it may be significantly faster. We will quickly recapitulate the algorithms of PCPT, PPPT methods and of the classical implicit method and apply them on a passport option pricing problem with non-standard payoff. We will present modifications needed to solve this problem effectively with the PPPT method and compare the performance with the PCPT method and the classical implict method.

INTRODUCTION

The Hamilton-Jacobi-Bellman (HJB) equation is a partial differential equation (PDE) typically used to solve optimal control problems. As it is highly nonlinear, solutions in the classical sense may not exist, and the concept of viscosity solutions [1] is used. This has significant implications for numerical methods, which now need to be monotone according to the theory of Barles and Souganidis [2] in order to converge. Monotone implicit methods were already presented in several papers, for example in [3] . To determine the optimal control policy in each time layer, these methods use policy iteration. Another approach, used in papers [3],[4] is the piecewise constant policy (PCPT) timestepping method. The basic idea of this method is solving several different PDEs with different constant policies in each time layer and then pick in each node the maximum result. In [5], we presented a modification of this method, so called piecewise predicted policy timestepping (PPPT) method. In contrast to the PCPT method, in most time layers we will solve a smaller number of PDEs, however non-constant control policies will be used. We will "predict" these control policies from the solution of the problem on a coarse grid. Because of smaller number of PDEs to solve in each time layer, this method may be significantly faster. Here, we will present only the algorithm of the method, and its application to a problem from computational finance. A convergence result for the PPPT method and detailed relationship to PCPT method is described in [5].

NUMERICAL METHODS FOR HAMILTON-JACOBI-BELLMAN EQUATIONS

Let us start with the definition of the Hamilton-Jacobi-Bellman equation. The one-dimensional **Hamilton-Jacobi-Bellman equation** is a PDE of the form

$$\frac{\partial V}{\partial t} - \max_{\theta\in\Theta}\left(a(x,t,\theta)\frac{\partial^2 V}{\partial x^2} + b(x,t,\theta)\frac{\partial V}{\partial x}c(x,t,\theta)V + d(x,t,\theta)\right) = 0, \quad V(x,0) = V_0(x), \tag{1}$$

where $x \in \mathbb{R}$, $t \in \mathbb{R}^+$, $a(x,t,\theta)$ is always non-negative and Θ is called control set. In our setting, the control set Θ will be defined as a finite set $\Theta = \{\theta_1, \theta_2, \ldots, \theta_Q\}$. A solution of this equation (1) is a function $V(x,t) : S \times [0,T] \to \mathbb{R}$, $S \subset \mathbb{R}^K$, $T > 0$.

The analytical solution of the Hamilton-Jacobi-Bellman equation is rarely feasible, therefore numerical methods are obligatory. We seek the solution of the HJB equation in the viscosity sense and therefore according to [2] the scheme must be monotone in order ensure convergence.

Discretization of the Hamilton-Jacobi-Bellman equation

The first step for constructing numerical algorithm is to discretize the equation to be solved. We will use a rectangular grid with one time (t) dimension and one space (x) dimension. We will denote the nodes as (x_i, t_j), where indices $i \in \{0, 1, \dots, N\}$, $j \in \{0, 1, \dots M\}$ indicate the position of the node in the grid. The distance between nodes (x_i, t_j) and (x_i, t_{j+1}) will be denoted as $\Delta_j t$, and distance between nodes (x_i, t_j) and (x_{i+1}, t_j) will be denoted as $\Delta_i x$. With v_i^j we will denote an approximation of solution of the HJB equation $V(x_i, t_j)$.

We must ensure that this discretization will be monotone. We denote the discretized HJB equation in point (x_i, t_j) as $G_{i,j}(v_i^j, v_{i_1}^{j_1}, v_{i_2}^{j_2}, \dots, v_{i_K}^{j_K})$ where $i \neq i_k$, $j \neq j_k$, $\forall k = 1, 2, \dots, K$. Then this scheme is monotone, if the function G is non-increasing in $v_{i_1}^{j_1}, v_{i_2}^{j_2}, \dots, v_{i_K}^{j_K}$. Monotonicity of a numerical scheme simply means that an increase in input values would never lead to a decrease in the output. We will use the discretization introduced by Wang and Forsyth [6], that is monotone, and can be consistent of second order in ideal case:

$$
\begin{aligned}
\frac{\partial V}{\partial t}(x_i, t_j) &\approx \frac{v_i^j - v_i^{j-1}}{\Delta_{j-1} t}, \\
\frac{\partial^2 V}{\partial x^2}(x_i, t_j) &\approx \frac{v_{i-1}^j - 2v_i^j + v_{i+1}^j}{\Delta_{i-1} x \Delta_i x}, \\
\frac{\partial V}{\partial x}(x_i, t_j) &\approx \frac{v_{i+1}^j - v_{i-1}^j}{\Delta_{i-1} x + \Delta_i x}, \quad \text{if} \quad \frac{b(x_i, t_j, \theta)}{\Delta_{i-1} x + \Delta_i x} \leq \frac{a(x_i, t_j, \theta)}{\Delta_{i-1} x \Delta_i x}, \\
\frac{\partial V}{\partial x}(x_i, t_j) &\approx \frac{(s+1)v_{i+1}^j - 2s v_i^j + (s-1)v_{i-1}^j}{(1+s)\Delta_i x + (1-s)\Delta_{i-1} x}, \quad s = \operatorname{sign}(b(x_i, t_j, \theta)) \quad \text{otherwise.}
\end{aligned}
$$

For simplicity, we will denote the $(N+1)$-dimensional vector with i-th element v_i^j, as v^j. Now we can write the discretized HJB equation in the following form:

$$
\frac{v_i^j - v_i^{j-1}}{\Delta_{j-1} t} = \max_{\theta \in \Theta} L_{i,j,\theta} v^j, \tag{2}
$$

where

$$
L_{i,j,\theta} v^k = a(x_i, t_j, \theta) \frac{v_{i-1}^k - 2v_i^k + v_{i+1}^k}{(\Delta_j x)^2} + b(x_i, t_j, \theta) \frac{(s+1)v_{i+1}^k - 2s v_i^k + (s-1)v_{i+1}^k}{2\Delta_j x} + c(x_i, t_j, \theta) v_i^k + d(x_i, t_j, \theta). \tag{3}
$$

Next we will present 3 different algorithms based on this discretization. In all algorithms we will compute the values v_i^j successively with time t. That means, we will at first compute all values in time layer t^j before proceeding to time layer t_{j+1}. We will need values in first time layer t_0 as initial condition. Moreover, we will need some boundary conditions. In our case, we will use Dirichlet boundary conditions that preserve monotonicity. Therefore, instead of equation (2) in nodes (x_0, t_j), (x_N, t_j) we will used simple equations

$$
v_0^j = BC_0(x_0, t_j), \quad v_N^j = BC_N(x_N, t_j), \tag{4}
$$

with some predefined functions $BC_0(x, t)$, $BC_N(x, t)$.

Classical implicit method with policy iteration

First we will introduce the algorithm that is most widely used to solve the HJB equation. It's similar to classical implicit method for solving convection-diffusion equations, however in order to find the optimal control a policy iteration is needed.

Algorithm:

1. v^0 is determined by the initial condition.
2. FOR $j = 1, 2, \ldots, M$:
   - 2.1 set $v^{(0)} = v^{j-1}$, $k = 0$
   - 2.2 REPEAT 'policy iteration':
     - $k = k + 1$
     - $\theta_i^{(k)} = \arg\max_\theta L_{i,j,\theta} v^{(k-1)}$ for $i \in \{0, 1, \ldots, N\}$
     - Solve system of equations:
       $v_i^{(k)} = v_i^{j-1} + \Delta_{j-1} t L_{i,j,\theta_i^{(k)}} v^{(k)} \quad for \quad i \in \{1, \ldots, N-1\}$
       $v_0^{(k)} = BC_0(x_0, t_j), \quad v_N^{(k)} = BC_N(x_N, t_j)$
   - 2.2 UNTIL $\|v^{(k)} - v^{(k-1)}\|_2 < TOL$
   - 2.3 $v^j = v^{(k)}, \quad \bar{\theta}_i^j = \theta_i^{(k)}$.
2. END.

Let us note, that we search here for the optimal control $\theta_i^{(k)}$ simply by trying all possible controls. The repeat-until part of the algorithm is called policy iteration. The vector $\bar{\theta}^j \in \mathbb{R}^{N+1}$ with elements $\bar{\theta}_i^j$ will be called optimal control policy in time layer j.

Piecewise constant policy timestepping method

The next algorithm that we used in our numerical examples is the so-called piecewise constant policy timestepping (PCPT) method. It is described for example in the papers [7], [3], [4]. Using this method, we completely avoid the policy iteration. Another advantage is, that this method can be easily parallelized. The main idea of the method is solving in each time layer Q PDEs with constant controls $\theta_q, q = 1, 2, \ldots, Q$ and then choose in each node x_i the optimal control leading to the biggest value. The following algorithm will clarify this approach.

Algorithm:

1. v^0 is determined by the initial condition.
2. FOR $j = 1, 2, \ldots, M$:
   - 2.1 FOR $q = 1, 2, \ldots, Q$
     - Solve system of equations:
       $v_i^{(q)} = v_i^{j-1} + \Delta_{j-1} t L_{i,j,\theta_q} v^{(q)} \quad for \quad i \in \{1, \ldots, N-1\}$
       $v_0^{(q)} = BC_0(x_0, t_j), \quad v_N^{(q)} = BC_N(x_N, t_j)$
   - 2.1 END
   - 2.2 $\kappa_i = \arg\max_q v_i^{(q)}, \quad v_=^j v^{(\kappa_i)}, \quad \bar{\theta}_i^j = \theta_{\kappa_i}, \quad i \in \{0, 1, \ldots, N\}$.
2. END.

In this context we will use the term policy rather loosely. According to the context we may refer to the $(N+1)$-dimensional vector of controls used in one time layer with $(N+1)$ nodes, or to the whole $(N+1) \times (M+1)$ array of controls used on the whole grid. By policy we also understand a two-dimensional function of variables x, t or one-dimensional function of variable x that assigns a value of control to any point (x, t) on the computational domain, or (in one-dimensional case) to any point x in the particular time-layer.

Piecewise predicted policy timestepping method

In the PCPT method, we solved Q different PDEs in each time layer using constant policies at first, and then compared the results to choose the optimal policy in each node of the current time layer. Let us suppose that we already computed approximation of the solution on a coarse grid with some method, and now we want to compute a better approximation on finer grid. To do this we can use for example the PCPT method, and test all possible policies in each time layer. However, we already have some approximation of the optimal policy from the coarse grid. If the true optimal control policy is not far away from this approximation, then testing some of the constant policies in PCPT method will be abundant. Therefore, the idea of the new piecewise predicted policy timestepping (PPPT) method introduced in [5] is to use this prediction on the coarse grid to check only policies which are near to the "predicted" optimal policy from the coarse grid.

As well as in case of PCPT method, we will solve a few PDEs in each time layer. However, constant policies will not be used anymore. Instead of them, we will use a set of policies that are near to the predictions from the coarse grid. We can divide the algorithm into 3 steps:

1. Compute the solution of HJB equation on the coarse grid with PCPT or classic implicit method. Save the approximation of the optimal control used on this grid -this will be the benchmark for predictions of the control policies for the fine grid.
2. Create a set of "predicted" two dimensional (space, time) control policy functions from the optimal control approximation computed on the coarse grid
3. Compute the solution on the fine grid comparing in each node results of controls determined by the predicted control functions evaluated in that node.

First we will describe the first and second step of the algorithm: construction of the predicted control functions:

Construction of the predicted control functions -algorithm:

1. Solve HJB PDE with classical or PCPT method on a coarse grid. By-product of this solution should be the array of optimal controls $\tilde{\theta}_i^j$ for all nodes (x_i, t_j) with $i \in \{0, 1, 2, \ldots, \tilde{N}\}$, $j \in \{0, 1, 2, \ldots, \tilde{M}\}$, where $\tilde{N}+1, \tilde{M}+1$ are dimensions of the coarse grid.
2. Define control indices $\tilde{z}_i^j$, such that $\tilde{\theta}_i^j = \theta_{\tilde{z}_i^j}$
3. Determine number of control functions $Z = \max\limits_{(i,j)\in\tilde{I}} \left|\tilde{z}_i^j - \tilde{z}_i^{j+1}\right|$ where $\tilde{I} = \{1, 2, \ldots, \tilde{N}\} \times \{1, 2, \ldots, \tilde{M}-1\}$
4. Define $\tilde{M}$ 1-dimensional index functions in the layers of the coarse grid:
   for $j = 1, 2, \ldots, \tilde{M}$:
   - $\tilde{z}^j(x) = \tilde{x}_i^j$ where $i = \arg\min\limits_{k\in\{0,1,\ldots,\tilde{N}\}} \|x - x_k\|_\infty$
5. Define:
   $up(x,t) = \tilde{z}^j(x)$ where $j = \arg\min_{k\in\{1,2,\ldots,\tilde{M}\}, t\le t_k} |t - t_k|$
   $down(x,t) = \tilde{z}^j(x)$ where $j = \arg\min_{k\in\{1,2,\ldots,\tilde{M}\}, t\ge t_k} |t - t_k|$
   Define $Z-2$ 2-dimensional index functions:
   for $z = 1, 2, \ldots, Z-2$:
   - $\tilde{z}^z(x,t) = round\left(\frac{z-1}{Z-3} up(x,t) + \frac{Z-2-z}{Z-3} down(x,t)\right)$
6. Determine neighbor index functions:
   $\tilde{z}^{Z-1}(x,t) = \min\left(\max_{z\in\{1,2,\ldots,Z-2\}} \tilde{z}^z(x,t), J\right)$
   $\tilde{z}^{Z}(x,t) = \max\left(\min_{z\in\{1,2,\ldots,Z-2\}} \tilde{z}^z(x,t), 1\right)$
7. Create control functions:
   for $z = 1, 2, \ldots, Z$:
   - $\theta^z(x,t) = \theta_{\tilde{z}^z(x,t)}$
8. End.

Now the algorithm for computing the solution on the fine grid using the predicted control functions follows:

Computation of solution on fine grid using prediction -algorithm:

1. v^0 is determined by the initial condition.
2. FOR $j = 1, 2, \ldots, M$:
   2.1 FOR $z = 1, 2, \ldots, Z$:
   - Define $\bar{\theta}^{j,z}$: $\bar{\theta}_i^{j,z} = \theta^z(s_i, t_{j+1})$ $(i \in \{0, 1, 2, \ldots, N\}$
   - Solve system of equations:
     $v_i^{(z)} = v_i^j + \Delta_{j-1} t L_{i,j,\bar{\theta}_i^{j,z}} v^{(z)} \quad for \quad i \in \{1, \ldots, N-1\}$
     $v_0^{(z)} = BC_0(x_0, t_j), \quad v_N^{(z)} = BC_N(x_N, t_j)$

   2.1 END
   2.2 $\kappa_i = \arg\max_k v_i^{(z)}, \quad v_=^j v^{(\kappa_i)}, \quad \bar{\theta}_i^j = \theta_{\kappa_i}, \quad i \in \{0, 1, \ldots, N\}.$
2. END

Convergence of the methods

For convergence (according to [2]) of classic implicit method we refer to [3]. A proof of convergence of PCPT method can be found in [3], for 1-dimensional case or in [8] for the 2-dimensional case. Extension to the N-dimensional case is straightforward. Following the ideas of these proofs, convergence of PPPT method was shown in [5] under some restrictions. These restrictions basically express the requirement that the true optimal policy will be included in some of the predicted control functions for any node. Let us remark, that the main convergence requirement of all these method besides consistency and stability is the monotonicity property of the scheme.

NUMERICAL EXAMPLE: PRICING PASSPORT OPTION WITH NON-CONVEX PAYOFF

Problem setting

In this Part we will test all 3 methods on the Hamilton-Jacobi-Bellman equation for passport option pricing from [6]. Passport options are contracts that allow the buyer to run trading account for a certain amount of time. After the maturity, buyer of this contract can keep the profit, however the potential loss will be covered by the seller. Here we will examine the case in which the buyer is allowed to invest in one particular asset only. The price depends on buyer's wealth W, current asset price S and time to maturity t. According to [6], [9], the HJB equation for the current price of the contract can be simplified to the form

$$\frac{\partial V}{\partial t} = \max_{|\theta| \leq 1} \left(\frac{\sigma^2}{2}(x-\theta)^2 \frac{\partial^2 V}{\partial x^2} + ((r - r_c - \gamma)\theta - (r - r_t - \gamma)x) \frac{\partial V}{\partial x} - \gamma V \right), \quad V(x, 0) = V_0(x). \tag{5}$$

Here, t is time to maturity $x = W/S$ and V is the option price divided by S. By r, we denote risk-free interest rate, γ is the dividend rate, r_c is the cost of carry rate, r_t is the interest rate for the trading account and σ is the volatility. The number of shares that the investor holds (control variable) is denoted by θ, and it does not have to be an integer. In this case the seller of the option requires the constraint $|\theta| \leq 1$. For comparison reasons we used the same parameter values as in [6]: $r = 0.08, \gamma = 0.03, r_c = 012, r_t = 0.05, \sigma = 0.2$.

Computational domain: Maturity of the option will be one year, the space domain will be restricted to $[-3, 4]$. The grid will be equally spaced in time, and unequally in space (nodes will be more dense near to zero)

Initial and boundary conditions: According to [10], in the case of convex payoff (initial condition), it is always optimal to choose either $q = 1$ or $q = -1$. In that case the PCPT method requires to solve only 2 PDEs in each time-step, and therefore it is clearly better then our PPPT method. However, since we want to test also the PPPT method, we will use non-convex initial condition:

$$V_0(x) = \min\left(\max(0, x), \max(0, 0.8 + 0.2x)\right) \tag{6}$$

This initial condition can be also easily interpreted: Buyer of the option pays 80% of the profit above the current asset price to the seller. We should note that for validating our implementation we tested the method also with the convex payoff and got results similar to those in [6]. We use Dirichlet boundary conditions according to [6]:

$$V(x_{min}, t) = 0, \quad V(x_{max}, t) = 0.8 + 0.2x_{max}, \quad [x_{min}, x_{max}] = [-3, 4] \tag{7}$$

Numerical results

Using the discretization described in the first part of this paper together with initial and boundary conditions presented here, we implemented all 3 algorithms (classical implicit method, PCPT and PPPT methods) for the problem of pricing the passport option with non-convex payoff. We ran our simulation on grids with different level of refinement, and compared our estimation of error of the approximation, experimental order of convergence and computational time needed. Results of these numerical simulations are presented in Table 1. As reference solution, we used a solution computed with the classical implicit method on a grid with 12801 time layers and 3841 space layers. In case of PPPT method we compute the control prediction for each simulation on a coarser grid with 10-times larger time-step size and the same discretization in space. The error of the approximation is denoted as Err and estimated by the formula

$$Err \;\; A^k = \|A^k - A^{ref}\|_2, \tag{8}$$

where A^k denotes last time-layer of the approximation of the solution on the k-th refinement level, and A^{ref} denotes the approximation of the solution that is used as reference solution. Experimental order of convergence is denoted as EOC and computed using the formula

$$EOC \;\; A^k = \frac{\log(ErrA^{k-1}) - \log(ErrA^k)}{\log(h_{k-1}) - \log(h_k)} \tag{9}$$

where h_k is the step size on the k-th refinement level. Computational time is in Table 1 denoted simply as Time.

TABLE 1. Error, experimental order of convergence and computational time for classical implicit method, PCPT and PPPT methods, for different numbers of nodes

k	1	2	3	4	5	6	7
Time-layers	101	201	401	801	1601	3201	6401
Space-layers	31	61	121	241	481	961	1921
Classical							
Err	1,95E-005	2,15E-006	7,82E-007	3,72E-007	1,46E-007	5,19E-008	4,35E-009
EOC		3.18	1.46	1.07	1.35	1.49	3.57
Time	0.909	1.906	4.587	11.708	33.974	107.869	387.558
PCPT							
Err	2,01E-005	2,55E-006	1,05E-006	4,79E-007	1,83E-007	6,37E-008	6,16E-009
EOC		2.98	1.28	1.13	1.39	1.52	3.37
Time	0.519	1.243	2.912	7.231	20.899	67.107	234.978
PPPT							
Err	2,05E-005	2,74E-006	1,21E-006	5,54E-007	2,12E-007	7,36E-008	7,63E-009
EOC		2.90	1.18	1.13	1.39	1.52	3.27
Time	0.204	0.336	0.865	2.004	9.236	26.084	87.292

Figure 1 illustrates the dependence between computational time and error by all 3 methods. We clearly observe, that the PPPT method is always the most effective. For fine discretization it is about 2 times faster than the PCPT method on the same level of accuracy, and even faster than the classic implicit method. This argument for the PPPT method is even stronger if we take into account that the reference solution was computed with the classical implicit method, which means that the estimation of error is negatively biased towards the classical implicit method. We note that the increase of experimental order of convergence for the finest grids is also biased because we use a numerical approximation instead of an analytical solution as reference solution.

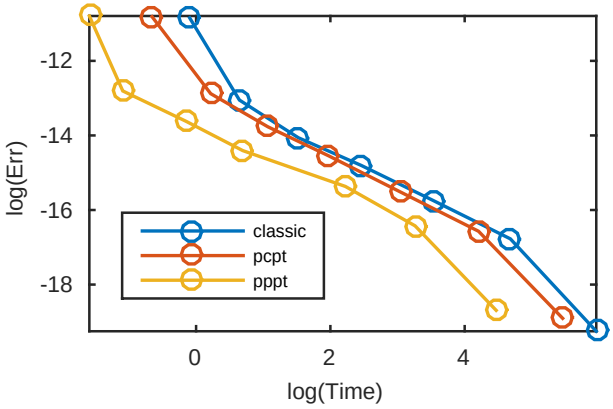

FIGURE 1. Comparison of logarithm error against logarithm of computational time for classic, PCPT and PPPT method with different grids.

CONCLUSION

In this paper, we described 3 different methods for solving the Hamilton-Jacobi-Bellman (HJB) equation. The first two schemes are the well-known classical implicit method and the piecewise constant policy timestepping PCPT method. The third of these methods, the so-called piecewise predicted policy timestepping (PPPT) method, that can be seen as some modification of the PCPT method, was described only recently in [5], together with one example where this method was the most time effective. Therefore, the aim of this paper was to compare these methods on an new benchmark problem. We have chosen the passport option pricing problem that is described for example in [10] or [6] and that leads to a HJB equation. We implemented all three methods for this problem and and computed the approximation of solution on different refinement levels. We observed an experimental order of convergence between 1 and 2 which corresponds to the theory in [6]. As shown in [11], one cannot construct a monotone scheme consistent of order higher than 2. The PPPT method was clearly most effective, therefore it should be considered in cases when computational time matters.

REFERENCES

[1] M. G. Crandall, H. Ishii, and P.-L. Lions, Bulletin of the American Mathematical Society **27**, 1–67 (1992).
[2] G. Barles and P. E. Souganidis, "Convergence of approximation schemes for fully nonlinear second order equations," in *Asymptotic Analysis*, 4 (1991), pp. 2347–2349.
[3] P. A. Forsyth and G. Labahn, Journal of Computational Finance **11**, p. 1 (2007).
[4] C. Reisinger and P. Forsyth, Applied Numerical Mathematics **103**, 27–47 (2016).
[5] I. Kossaczký, M. Ehrhardt, and M. Günther, "Piecewise fixed policy timestepping schemes for Hamilton-Jacobi-Bellman equations," (2016), preprint on www.imacm.uni-wuppertal.de.
[6] J. Wang and P. A. Forsyth, SIAM Journal on Numerical Analysis **46**, 1580–1601 (2008).
[7] N. Krylov, Electronic Journal of Probability **4**, 1–19 (1999).
[8] K. Ma and P. Forsyth, Submitted to IMA Journal of Numerical Analysis (2015).
[9] J. Topper, *A Finite Element Implementation of Passport Options*, Master's thesis (2003).
[10] D. M. Pooley, P. A. Forsyth, and K. R. Vetzal, IMA Journal of Numerical Analysis **23**, 241–267 (2003).
[11] I. Kossaczký, M. Ehrhardt, and M. Günther, Applied Mathematics Letters **52**, 53–57 (2016).