The Tree-Grid Method

Dissertation

zur Erlangung
des akademischen Grades
eines
Doktors der Naturwissenschaften (Dr. rer. nat.)

der
Fakultät für Mathematik und Naturwissenschaften
der
Bergischen Universität Wuppertal (BUW)
vorgelegt von

Igor Kossaczký

geboren am 19.12.1989 in Bratislava

betreut von: Prof. Dr. Matthias Ehrhardt (BUW)
Prof. Dr. Michael Günther (BUW)

Wuppertal, 2018

Die Dissertation kann wie folgt zitiert werden:

urn:nbn:de:hbz:468-20181029-153921-8
[http://nbn-resolving.de/urn/resolver.pl?urn=urn%3Anbn%3Ade%3Ahbz%3A468-20181029-153921-8]

Abstract

In this thesis we are concerned with the development of numerical schemes for solving the stochastic control problems and the related Hamilton-Jacobi-Bellman (HJB) equations. In the first part, we present the convergence theory and the standard finite difference methods (FDMs) used for solving HJB equations. We present then our result on non-existence of higher order monotone numerical methods. This result represents also the motivation for the numerical methods presented in this thesis. Rather than aiming for an high-order method, we focus on reducing the computational time.

The piecewise predicted policy timestepping method presented in this work represents a modification of the well-established piecewise constant policy timestepping method. The main idea of the method is reducing the control space based on the prediction computed on a coarse grid in order to reduce the computational time. We show the efficiency of the method on examples from finance.

The Tree-Grid methods represent the central topic of this thesis. The main essence of these methods is the combination of the tree structure, similar to that from trinomial tree methods, with the rectangular grid used in FDMs. We prove that these methods are unconditionally stable and convergent on an arbitrary grid. On the other hand, as the methods are explicit, they are faster and can be easily parallelized. We developed the methods for the cases of a one-dimensional and two-dimensional state variable, and have shown that for higher dimensions a monotone generalization is not feasible for a general problem. An additional advantage of the Tree-Grid method in the two-dimensional case is, that although the method uses a wide-stencil scheme, no interpolation is needed. For the case of a one-dimensional state variable, we developed a useful modification leading to a more efficient search for the optimal control. We tested all methods on examples from finance.

In the conclusion we also propose possible further research directions emerging from this thesis.

Acknowledgments

In the first place, I would like to express deepest thanks to my supervisors Prof. Matthias Ehrhardt and Prof. Michael Günther. They gave me the opportunity to write the Dissertation Thesis at the Department of Applied Mathematics and Numerical Analysis (AMNA). The discussions with them were inspiring, and their feedback and know-how was always a great help.

I'm very grateful to Prof. Daniel Ševčovič from the Comenius University in Bratislava for the interesting discussions. He was my master thesis supervisor and introduced me to the Hamilton-Jacobi-Bellman equation, the topic of this PhD Thesis.

Furthermore, I want to thank all my colleagues from AMNA for their friendly support in various situations and for making my doctoral studies more enjoyable.

Next, I'm thankful to all the teachers from my bachelor and master studies at the Comenius University for providing me with interesting insights into various areas of the applied mathematics. The knowledge that I acquired during those years was essential to start my PhD studies.

I'm most grateful to my beloved wife Tatiana for all her love. She was always my greatest support, also during the first two years of my PhD studies in Wuppertal, being thousand kilometers away.

I also want to express special thanks to my parents for all their support. Besides many other things, they brought me a positive attitude towards education and encouraged me to study in Germany.

Last but not least, I want to thank my siblings, family and my friends in Wuppertal and in Bratislava for all the good moments I could spend with them.

Contents

Notation

Let us introduce the most important notation used in this thesis:

- t – time variable
- T – final time (maturity in case of option pricing)
- $\bar{s}$ – state-space variable with dimension higher than two, or space variable in general (with undefined dimensionality)
- s – one-dimensional state-space variable
- (x, y) – two-dimensional state-space variable (both x and y are one dimensional)
- θ – control variable
- Θ – control set
- $\bar{\Theta}$ – set of control functions $\theta(\bar{s}, t)$
- $\bar{W}_t, \bar{S}_t$ – possibly higher-dimensional Wiener process and state process
- W_t, S_t – one dimensional Wiener process and state process
- $(W_t^x, W_t^y), (X_t, Y_t)$ – two dimensional Wiener process and state process
- $V(\bar{s}, t)$ resp. $V(s, t)$ or $V(x, y, t)$ – Value function
- $\mathbb{E}(\cdot)$ – Expected value

Abbreviations

BC	boundary condition
BS	Black-Scholes (model/equation)
CFL	Courant-Friedrichs-Lewy (condition)
EOC	experimental order of convergence
Err	error
FDM	Finite difference method
FSG	Forward shooting grid (method)
HJB	Hamilton-Jacobi-Bellman (equation)
HJBI	Hamilton-Jacobi-Bellman-Isaacs (equation)
PCPT	Piecewise constant policy timestepping (method)
PDE	partial differential equation
PPPT	Piecewise predicted policy timestepping (method)
RV	random variable
SCP	stochastic control problem
SDE	stochastic differential equation
TG	Tree-Grid (method)

Chapter 1

Introduction

In this thesis we are concerned with the development of the numerical schemes for solving the stochastic control problems (SCPs) and the related Hamilton-Jacobi-Bellman equations. At first, before defining a SCP in a formal manner, let us intuitively explain what a SCP is. A stochastic control problem can be defined as the problem of dynamically adapting a control variable to a stochastic process that depends on this variable during some finite (in our setting) time interval. The goal is to maximize the sum (in fact, the integral) of profits implied by the evolution of the controlled stochastic process together with the final profit depending only on the value of the controlled process at the final time. Alternatively, we can replace the profits by costs and the maximization by minimization. In our setting, we will model the uncertainty in the stochastic process by a Wiener process. Figure 1.1 illustrates this concept of SCP. We should note, that the underlying stochas-

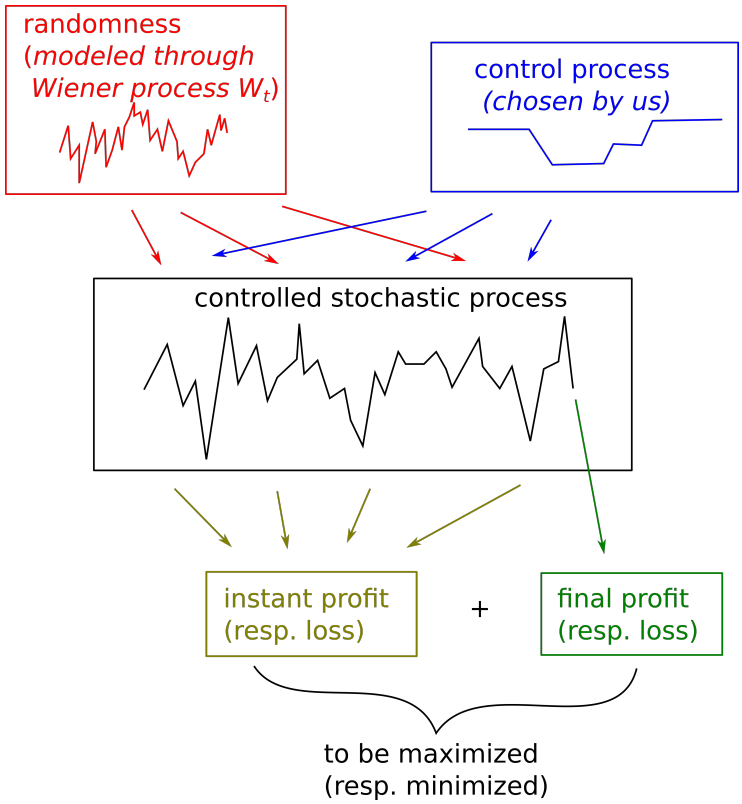

Figure 1.1: Schematic illustrating the SCP and its parts.

tic control process can also be multidimensional. However in this thesis, we will develop schemes only for one and two dimensional underlying processes. The difficulties arising by generalizing our numerical methods to higher dimensions are outlined in Chapter 8. Our definition of the SCP is of course not exhaustive. Other concepts of the SCP are for example:

- Stochastic control problems with infinite time horizon [31],
- Stochastic control problems with uncertainty modeled with Lévy process (or some other stochastic processes) [39],
- Optimal stopping time problems [31],
- Discrete SCPs [21].

Another possible generalization of the SCPs is the stochastic differential game [36, 38].

Although it is possible to solve the SCPs directly, it is often more convenient to solve the Hamilton-Jacobi-Bellman (HJB) equation. This is a nonlinear partial differential equation (PDE), therefore modeling of the stochasticity is not needed. The formulation of the HJB equation corresponding to the SCP is described in the next chapter.

Stochastic control problems and HJB equations arise in many applications. In this thesis we will solve numerically the HJB equations corresponding to the following problems from finance:

- Mean-variance optimal investment problem (Example 1, Chapter 4),
- Passport option pricing (Example 2, Chapters 4, 5, 6),
- Option pricing under uncertain volatility model (Example 3, Chapter 5),
- Option pricing under 2-factor uncertain volatility model (Example 4, Chapter 7).

Other SCPs or HJB equations arising in finance, but also in other areas are for example:

- Optimal portfolio allocation problem (see e.g. [24]),
- Gas storage valuation and optimal operation (see e.g. [10]),
- Optimal vaccination in SIR (Susceptible-Infectious-Recovered) model (see e.g. [23]),
- Monge-Ampère equation (see e.g. [32]).

As one can see, the SCPs play an important role in applied mathematics. In the following section we present the overview of the related scientific work with special emphasis on the numerical methods for HJB equations.

1.1 Related scientific works

For an overview on the general stochastic control theory, we refer the reader to [38, 50]. In [31], beside the control theory also Markov chains and numerical methods are discussed. For a reader interested in discrete optimal control theory we refer to [21]. For the study of stochastic differential equations and stochastic calculus we recommend for example [44] where the stochastic calculus is applied in finance, or a more theoretical work [2]. As it will become clear from the examples in this thesis, the HJB equation and control theory is also closely related to financial mathematics. For a reader interested in analytical and numerical methods for nonlinear financial models we recommend [43] and [14]. Finite

difference methods for option pricing can be found in [46] and also other numerical methods with focus on the implementation in Matlab can be found in [20]. For the study of numerical methods for PDE problems in general we refer to [19].

The Hamilton-Jacobi-Bellman (HJB) equation, as well as the other nonlinear PDEs may not have solution in the classical sense. Therefore, Crandall, Ishi and Lions [11] introduced in 1992 the concept of viscosity solution, suitable for HJB equations. For a brief introduction to the theory of viscosity solutions we refer to [34]. However, it can be a problem to find even such a viscosity solution analytically, therefore numerical methods are used. To prove the convergence of the approximation computed with these numerical methods to the viscosity solution often the theory of Barles and Souganidis [4] is used. Another, probabilistic approach to the convergence proofs is presented in the book of Kushner and Dupuis [31].

The numerical methods used to solve the above SCP can be divided into two classes, based on the formulation they are exploiting. The main idea of methods based on the partial differential equation (PDE) approach is to solve the HJB PDE with numerical methods as for example finite differences. Here, the implicit finite difference methods (FDM) using policy iteration were shown to be successful e.g. in the work of Forsyth and Vetzal [17]. An alternative approach to the policy iteration used in this method is the piecewise constant policy timestepping (PCPT) scheme used for example in [16], [41]. 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. A modification of this scheme leading to experimentally faster algorithms was proposed by the authors in [25] and is presented also in this thesis. An alternative approach based on the Ricatti transformation of the PDE was proposed by Kilianová and Ševčovič in [24]. The advantage of using this approach is that we don't need to solve the optimization problem in each time-layer.

Whereas methods based on the PDE approach are typically implicit, methods based on the original problem formulation (2.8),(2.9) are mostly explicit. Apparently the most famous of these approaches are the methods based on Markov chain approximations of the stochastic differential equation (SDE) (2.9) presented in the book of Kushner and Dupuis [31]. In finance, binomial and trinomial tree methods [1, 3] are widely used. These methods often present another viewpoint on the Markov chain approximation, and are equivalent up to some order to explicit FDMs (see for example [1]). These methods are known to suffer from instability if a certain condition on stepsizes is not met. This is also the reason why implicit methods are used more often. In tree and Markov chain methods, fulfilling this condition is achieved by a problem-specific construction of the grid or lattice. And finally, also based on the original problem formulation, there are forward shooting grid (FSG) methods [22], combining the tree lattice of binomial or trinomial method and the grid of the FDM or Markov chain approximation method. The FSG methods are typically used in path-dependent option pricing [5, 22], but can be easily implemented also for SCPs. These methods however may suffer from convergence problems as outlined by Forsyth, Vetzal and Zvan in [18]. A new explicit and unconditionally stable Tree-Grid method was developed in [28] and will be also presented in this thesis.

In two or more space dimensions, a generalization of the implicit unconditionally stable method from [48] was presented in [35] and later used in [9]. The main idea of this method is combining the wide and the fixed stencil depending on the correlation in the particular time-space node. Alternatively, explicit methods based on the ideas from [31] presented in papers [8, 12, 13] can be used. These are wide stencil schemes stable under some CFL

condition. Moreover, linear interpolation of the grid values is needed in these methods. In [29], a generalization of the Tree-Grid method for two space dimensions was developed. This generalization of the Tree-Grid method, presented also in this thesis, also falls into the class of the wide stencil schemes, however it is unconditionally stable for any grid and no interpolation of the grid values is needed.

1.2 Outline of the thesis

Let us now introduce the structure of the Thesis:

Chapter 1: In this chapter, we give the reader intuition on what a SCP is and for which applications it is important. In Section 1.1 we provide the reader with an overview of useful literature on numerical methods for solving SCPs and HJB equations and other related topics. Finally in this Section 1.2 we present the outline of the thesis.

Chapter 2: In Section 2.1 we formally define the SCP together with the dynamic programming equation and the HJB equation. We also look closer on the special cases of problems with one or two space dimensions, which are the target problems of this thesis. In Section 2.2, we define the viscosity solution and present the main results of the convergence theory of Barles and Souganidis [4].

Chapter 3: In Section 3.1 we describe the discretization of the HJB equation. Then we present the algorithms of the standard implicit methods from [16]: the implicit method with policy iteration and the PCPT method. The Section 3.2 is based on our paper [26]. We present here the result on non-existence of the higher order monotone schemes for solving the HJB equations. This result motivates also the further direction of our work: rather than aiming for higher order of consistency, we develop methods that are explicit while remaining unconditionally stable (Tree-Grid methods) or introduce heuristics for the search of the optimal control (PPPT method).

Chapter 4: In this chapter based on the paper [25] we introduce the PPPT method that can be seen as a modification of the PCPT method. In contrast to the PCPT method, we solve a smaller number of PDEs in most time layers, 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. In the Sections 4.2, 4.3, we test this method on two example problems from finance: on the mean-variance optimal investment problem and on the passport option pricing problem.

Chapter 5: This chapter is based on the paper [28]. We introduce here the Tree-Grid method for SCPs with one space dimensions. The name of the method is derived from the tree structure similar to that of the trinomial tree methods, which is however in our case defined on an arbitrary grid. The method is explicit, yet unconditionally stable, consistent and monotone. These properties are achieved by making the stencil dependent on the time step, but also by introducing the artificial diffusion in the numerical scheme. We prove the convergence of the method in Section 5.3 and compare the method to other alternatives: Finite difference methods, Markov chain approximation methods and FSG methods. In Sections 5.4, 5.5, we compare the performance of this method and of the standard implicit FDM on two example problems from finance: on the uncertain-volatility option pricing

problem and on the passport option pricing problem.

Chapter 6: In this chapter based on the paper [27], we address the following issue: as the stencil size changes for different values of the control in the Tree-Grid method, it is difficult to search for the optimal control in a way different from brute-force approach. Therefore, we propose here a modification of the Tree-Grid method leading to a scheme with a stencil of constant size. This gives us the possibility for more efficient search of the optimal control e.g. by using the Fibonacci algorithm, as we also illustrate on the passport option pricing example in the Section 6.3. Moreover, we present in this chapter a better way of introducing the artificial diffusion in the numerical scheme.

Chapter 7: Based on the paper [29], this chapter covers the generalization of the Tree-Grid method to two space dimensions. Beside artificial diffusion, also the reduction of covariance is introduced to keep the stencil as simple as possible, while still remaining consistent. The scheme can be classified as a wide-stencil scheme, but in contrast to other wide stencil schemes e.g. in [12], we do not need any interpolation. The method is also unconditionally stable on any rectangular grid and we prove its convergence in the Section 7.3. We exemplify usefulness of the method on two factor option pricing uncertain volatility model in the Section 7.4.

Chapter 8: The generalization of the Tree-Grid method from Chapter 7 opens the question, if also a generalization to higher dimensions is possible. In this chapter, we show that the generalization to dimensions higher than two is not possible for an arbitrary problem. The problem is, that for stronger covariance, negative weights may appear in the numerical scheme, causing it to be non-monotone and possibly unstable. However, for some higher dimensional problems with mild covariance, the generalization of the Tree-Grid method sketched in this chapter may be possible. In Section 8.4, we discuss which ideas from the Tree-Grid method can be employed also in other wide stencil schemes.

Chapter 9: In this last chapter we sum up the results of this thesis and provide the outlook on the possible directions of the future research.

2 Chapter 2

Stochastic control problems and Hamilton-Jacobi-Bellman equations

In the previous chapter, we intuitively formulated what a stochastic control problem (SCP) is. Now, we will define it in a formal manner. We also present here the dynamic programming equation and the Hamilton-Jacobi-Bellman equation. These equations are crucial for development of the numerical schemes and for the proofs of convergence. At first, we will describe the general P-dimensional stochastic control problem together with the dynamic programming equation and the HJB equation. Then, we will look at the special cases of the one-dimensional and the two-dimensional SCPs. In the last section, we define the viscosity solution and present the convergence theory for the numerical methods for nonlinear partial differentials equations that will be needed in our convergence proofs in the later chapters.

2.1 General stochastic control problem

Let us now define the P-dimensional **stochastic control problem** (SCP):

$$V(\bar{s},t) = \max_{\theta(s,t)\in\bar{\Theta}} \mathbb{E}\left(\int_t^T \exp\left(\int_t^k r(\bar{S}_l, l, \theta(\bar{S}_l, l))dl \right) f(\bar{S}_k, k, \theta(\bar{S}_k, k))dk \right.$$
$$\left. + \exp\left(\int_t^T r(\bar{S}_k, k, \theta(\bar{S}_k, k))dk \right) V_T(\bar{S}_T) \Big| \bar{S}_t = \bar{s} \right), \quad (2.1)$$

$$d\bar{S}_t = \bar{\mu}(\bar{S}_t, t, \theta(\bar{S}_t, t))dt + \bar{\sigma}(\bar{S}_t, t, \theta(\bar{S}_t, t))d\bar{W}_t, \quad (2.2)$$
$$0 < t < T, \quad \bar{s} \in \mathbb{R}^P, \quad P \in \mathbb{N},$$
$\bar{\rho}(\bar{S}_t, t, \theta(\bar{S}_t, t))$ is the correlation matrix of $\bar{W}_t$.

At first we will explain the notation:

- Equation (2.1) represents the definition of the so-called **value function** $V(\bar{s},t)$. The value function can be interpreted as a function that assigns to each state $\bar{s}$ in each time t a specific numeric value. This value is defined as the expectation of the overall future profit if the optimal control policy is used. The aim of our numerical methods presented in this work is to provide us with reasonable approximation of this value function.

- The stochastic differential equation (2.2) represents the P-dimensional controlled stochastic Itô process $\bar{S}_t$.

- $t \in [0,T]$ denotes time, T denotes the final time.
- $\bar{s} \in \mathbb{R}^P$ is called **state variable**. It represents the current value (state) of the process.
- $\bar{W}_t$ denotes a P-dimensional Wiener process with correlation matrix $\bar{\rho}(\bar{s},t,\theta)$ for $\bar{S}_t = \bar{s}$ and $\theta(\bar{s},t) = \theta$. The variance of each component is 1.
- Θ denotes the control set. For our purposes, we will suppose that Θ is discrete. If this is not the case, we can easily achieve this property by its discretization. Then, $\theta \in \Theta$ is called **control variable**.
- $\theta(\bar{s},t) : \mathbb{R}^P \times [0,T] \to \Theta$ is the control function. Its value changes in time and depends on time as well as on the current value of the stochastic process (2.2). On the other hand, the increment of the stochastic process (2.2) depends on the current value of the control function in each time t.
- $\bar{\Theta}$ is the space of all measurable control functions $\theta(\bar{s},t)$.
- The function $\bar{\mu}(\bar{s},t,\theta) : \mathbb{R}^P \times [0,T] \times \Theta \to \mathbb{R}^P$ represents the drift function of the process $\bar{S}_t$. The function $\bar{\sigma}(\bar{s},t,\theta) : \mathbb{R}^P \times [0,T] \times \Theta \to D^+_{P\times P}$ represents the volatility function of the process $\bar{S}_t$, where $D^+_{P\times P}$ is the set of all diagonal $P \times P$ matrices with non-negative entries.

  Let $\rho(\bar{s},t,\theta) = \zeta(\bar{s},t,\theta)\zeta(\bar{s},t,\theta)^\top$ and $\Sigma^{1/2} = \zeta(\bar{s},t,\theta)\bar{\sigma}(\bar{s},t,\theta)$. Then $\Sigma = \Sigma^{1/2}{\Sigma^{1/2}}^\top$ is the covariance matrix of the process $\bar{S}_t$ for $\bar{S}_t = \bar{s}$ and $\theta(\bar{s},t) = \theta$.

  We suppose that the functions $\bar{\mu}$ and $\Sigma^{1/2}$ satisfy the following conditions (see [38]):

$$|\bar{\mu}(\bar{s}_1,t_1,\theta_1) - \bar{\mu}(\bar{s}_2,t_2,\theta_2)| + |\Sigma^{1/2}(\bar{s}_1,t_1,\theta_1) - \Sigma^{1/2}(\bar{s}_2,t_2,\theta_2)| \\ \leq l|\bar{s}_1 - \bar{s}_2| + m(|t_1 - t_2| + |\theta_1 - \theta_2|), \quad (2.3)$$

$$|\bar{\mu}(0,t,\theta)| + |\Sigma^{1/2}(0,t,\theta)| \leq K \quad \forall t,\theta \in [0,T] \times \Theta, \quad (2.4)$$

  where $K, l \in \mathbb{R}^+$ and $m(\cdot)$ is a bounded function.
- The function $f(\bar{s},t,\theta) : \mathbb{R}^P \times [0,T] \times \Theta \to \mathbb{R}$ represents the increment of the instant profit (also called instant reward) or of the instant loss in time t, state $\bar{s}$ and under control θ. We suppose $f(\cdot)$ to be continuous in $\bar{s}$ and uniformly continuous in t (see [38]).
- The function $r(\bar{s},t,\theta) : \mathbb{R}^P \times [0,T] \times \Theta \to \mathbb{R}$ represents the discount factor in time t, state $\bar{s}$ and under control θ. We suppose $r(\cdot)$ to be continuous in $\bar{s}$ and uniformly continuous in t (see [38]).
- The function $V_T(\bar{s})$ (also called terminal condition) represents the profit or loss at the final time T in state $\bar{s}$. It holds $V(\bar{s},T) = V_T(\bar{s})$.

Let us note, that if the functions $f(\cdot), V_T(\cdot)$ represent the loss (or costs), the maximization should be replaced by minimization in (2.1).

Following the Bellman principle of optimality, introduced by Richard Bellman [6] (at first for deterministic discrete dynamic systems) the so called **dynamic programming**

equation holds:

$$V(s,t_j) = \max_{\theta(s,t)\in\bar{\Theta}_{t_j}} \mathbb{E}\left(\int_{t_j}^{t_{j+1}} \exp\left(\int_{t_j}^{k} r(\bar{S}_l, l, \theta(\bar{S}_l, l))dl \right) f(\bar{S}_k, k, \theta(\bar{S}_k, k))dk \right.$$
$$\left. + \exp\left(\int_{t_j}^{t_{j+1}} r(\bar{S}_k, k, \theta(\bar{S}_k, k))dk \right) V(\bar{S}_{t_{j+1}}, t_{j+1}) \Big| \bar{S}_{t_j} = \bar{s} \right), \quad (2.5)$$

where $0 \leq t_j < t_{j+1} \leq T$ are some time-points and $\bar{\Theta}_{t_j}$ is a set of control functions from $\bar{\Theta}$ restricted to the $\mathbb{R} \times [t_j, t_{j+1})$ domain. The dynamic programming equation provides us with a formula for the value function in time t_j depending only on the value function in an arbitrary later time t_{j+1}. Therefore, this equation is after discretization very suitable for successive computation of the value function in different time layers (from final time layer in $t = T$ defined by $V_T(\bar{s})$, up to the first in $t = 0$). We employ this idea later in Chapters 5-7. Using this dynamic programming equation (2.5), it can be shown [38], that solving the SCP (2.1),(2.2) is equivalent to solving a specific partial differential equation (PDE) the so-called **Hamilton-Jacobi-Bellman (HJB) equation**:

$$\frac{\partial V}{\partial t} + \max_{\theta\in\Theta} \left(\frac{1}{2}\mathrm{tr}\left(\Sigma(\cdot)\frac{\partial}{\partial \bar{s}}\left(\frac{\partial V}{\partial \bar{s}} \right) \right) + \bar{\mu}(\cdot)^\top \frac{\partial V}{\partial \bar{s}} + r(\cdot)V + f(\cdot) \right) = 0, \quad (2.6)$$
$$V(\bar{s}, T) = V_T(\bar{s}), \quad (2.7)$$
$$0 < t < T, \quad s \in \mathbb{R}^P,$$

where $\Sigma(\cdot)$, $\bar{\mu}(\cdot)$, $r(\cdot)$, $f(\cdot)$ are functions of $\bar{s}, t, \theta$ defined above. We remark, that if the maximum operator is replaced by the minimum operator in the SCP (2.1)-(2.2), it should be replaced by the minimum operator also in the HJB equation (2.6).

Possible generalizations of the stochastic control problems are the stochastic differential games where maximization and minimization is done simultaneously. The PDEs corresponding to this class of problems are the Hamilton-Jacobi-Bellman-Isaac equations [36]. In case of these equations, the single maximum operator is substituted by one maximum and one minimum (resp. supremum and infimum) operator each defined on a different control set. Use of even more general operators is analyzed for example in [45]. However, in this work we will restrict ourselves to the case of minimum and maximum operators. Let us note that this covers also the case of omitting the operator completely, as this can be seen as maximum through an one-element set. In that case, a relationship between (2.1), (2.2) and (2.6), (2.7) is established by the classical Feynman-Kac formula. In financial mathematics, this relationship is represented by a connection between the option pricing problem and the Black-Scholes equation [7].

For a deeper understanding of the principles of the stochastic control theory and of the HJB equation, we refer the reader to some classic literature on the topic e.g. [38, 31, 50]. Before moving to the next section, let us for convenience rewrite equations (2.1), (2.2) and (2.5)-(2.7) for the one-dimensional and two-dimensional setting, as for these cases numerical methods are developed in this thesis.

2.1.1 One-dimensional stochastic control problem and HJB equation

The one-dimensional stochastic control problem (2.1), (2.2) can be rewritten as follows:

$$V(s,t) = \max_{\theta(s,t)\in\bar{\Theta}} \mathbb{E}\left(\int_t^T \exp\left(\int_t^k r(S_l,l,\theta(S_l,l))dl \right) f(S_k,k,\theta(S_k,k))dk \right.$$
$$\left. + \exp\left(\int_t^T r(S_k,k,\theta(S_k,k))dk \right) V_T(S_T) \Big| S_t = s \right), \quad (2.8)$$
$$dS_t = \mu(S_t,t,\theta(S_t,t))dt + \sigma(S_t,t,\theta(S_t,t))dW_t, \quad (2.9)$$
$$0 < t < T, \quad s \in \mathbb{R}.$$

Here, s denotes the one-dimensional state variable, t is time, and the stochastic process S_t is also one-dimensional. The dynamic programming equation (2.5) following from Bellman's principle reads:

$$V(s,t_j) = \max_{\theta(s,t)\in\bar{\Theta}_{t_j}} \mathbb{E}\left(\int_{t_j}^{t_{j+1}} \exp\left(\int_{t_j}^k r(S_l,l,\theta(S_l,l))dl \right) f(S_k,k,\theta(S_k,k))dk \right.$$
$$\left. + \exp\left(\int_{t_j}^{t_{j+1}} r(S_k,k,\theta(S_k,k))dk \right) V(S_{t_{j+1}},t_{j+1}) \Big| S_{t_j} = s \right), \quad (2.10)$$

and the Hamilton-Jacobi-Bellman equation (2.6) that can be derived from the dynamic programming equation has in this setting the following form:

$$\frac{\partial V}{\partial t} + \max_{\theta\in\Theta}\left(\frac{\sigma(\cdot)^2}{2}\frac{\partial^2 V}{\partial s^2} + \mu(\cdot)\frac{\partial V}{\partial s} + r(\cdot)V + f(\cdot) \right) = 0, \quad (2.11)$$
$$V(s,T) = V_T(s), \quad (2.12)$$
$$0 < t < T, \quad s \in \mathbb{R},$$

where $\sigma(\cdot)$, $\mu(\cdot)$, $r(\cdot)$, $f(\cdot)$ are functions of s,t,θ.

2.1.2 Two-dimensional stochastic control problem and HJB equation

The two-dimensional stochastic control problem (2.1), (2.2) can be rewritten as follows:

$$V(x,y,t) = \max_{\theta(x,y,t)\in\bar{\Theta}} \mathbb{E}\left(\int_t^T \exp\left(\int_t^k r(*_l)dl \right) f(*_k)dk \right.$$
$$\left. + \exp\left(\int_t^T r(*_k)dk \right) V_T(X_T,Y_T) \Big| X_t = x, Y_t = y \right), \quad (2.13)$$
$$dX_t = \mu_x(*_t)dt + \sigma_x(*_t)dW_t^x, \quad (2.14)$$
$$dY_t = \mu_y(*_t)dt + \sigma_y(*_t)dW_t^y, \quad (2.15)$$
$$dW_t^x dW_t^y = \sigma_{xy}(*_t)/(\sigma_x(*_t)\sigma_y(*_t)) \quad (2.16)$$
$$*_t = (X_t, Y_t, t, \theta(X_t, Y_t, t)), \quad 0 < t < T, \quad x, y \in \mathbb{R},$$

where $x \in \mathbb{R}, y \in \mathbb{R}$ are state variables, the stochastic processes X_t, Y_t are one-dimensional ($\bar{s} = (x,y)$, $\bar{S}_t = (X_t, Y_t)$) and t is time. The dynamic programming equation (2.5) following from Bellman's principle reads:

$$V(x,y,t_j) = \max_{\theta(x,y,t)\in\bar{\Theta}_{t_j}} \mathbb{E}\left(\int_{t_j}^{t_{j+1}} \exp\left(\int_{t_j}^k r(*_l)dl \right) f(*_k)dk \right.$$
$$\left. + \exp\left(\int_{t_j}^{t_{j+1}} r(*_k)dk \right) V(X_{t_{j+1}}, Y_{t_{j+1}}, t_{j+1}) \Big| X_{t_j} = x, Y_{t_j} = y \right), \quad (2.17)$$

and the Hamilton-Jacobi-Bellman equation (2.6) that can be derived from the dynamic programming equation has in this setting the following form:

$$\frac{\partial V}{\partial t} + \max_{\theta\in\Theta}(LV + r(\cdot)V + f(\cdot)) = 0, \quad (2.18)$$
$$LV = \frac{\sigma_x(\cdot)^2}{2}\frac{\partial^2 V}{\partial x^2} + \sigma_{xy}(\cdot)\frac{\partial^2 V}{\partial x \partial y} + \frac{\sigma_y(\cdot)^2}{2}\frac{\partial^2 V}{\partial y^2} + \mu_x(\cdot)\frac{\partial V}{\partial x} + \mu_y(\cdot)\frac{\partial V}{\partial y}, \quad (2.19)$$
$$V(x,y,T) = V_T(x,y), \quad (2.20)$$
$$0 < t < T, \quad x, y \in \mathbb{R},$$

where $\sigma_x(\cdot)$, $\sigma_y(\cdot)$, $\sigma_{xy}(\cdot)$, $\mu_x(\cdot)$, $\mu_y(\cdot)$, $r(\cdot)$, $f(\cdot)$ are functions of x, y, t, θ.

2.2 Viscosity solutions and convergence theory

The Hamilton-Jacobi-Bellman equations are fully non-linear PDEs and might not posses a solution in the classical sense. Therefore, the concept of viscosity solutions was developed. However, as the closed form of the viscosity solution is rarely feasible, numerical methods for its approximation are needed. In this section, we will at first present the definition of the viscosity solution and then the convergence theory developed by Barles and Souganidis [4]. This theory provides us with sufficient conditions for a numerical scheme to converge to

the viscosity solution. We will use it to prove the convergence of the numerical schemes for solving the HJB equation, but we present here the theory for even more general nonlinear differential operators. Let us note that the K-dimensional variable $\bar{s}$ used here is split for the HJB equation into two parts, the 1-dimensional time variable t and the P-dimensional variable $\bar{s}$ (or s for $P = 1$, resp. (x, y) for $P = 2$).

Let

$$FV(\bar{s}) := F\left(\frac{\partial}{\partial \bar{s}}\left(\frac{\partial V(\bar{s})}{\partial \bar{s}}\right), \frac{\partial V(\bar{s})}{\partial \bar{s}}, V(\bar{s}), \bar{s}\right) = 0 \tag{2.21}$$

denote a fully nonlinear second order parabolic or elliptic PDE fulfilling the *ellipticity property*:

$$F(A_1, b, c, d) \leq F(A_2, b, c, d) \text{ for all } A_1 \geq A_2, A_1, A_2 \in S_{K\times K}, b \in \mathbb{R}^K, c \in \mathbb{R}, d \in \Omega \subseteq \mathbb{R}^K$$

where $S_{K\times K}$ is the space of all symmetric $K \times K$ matrices with natural ordering denoted as "$\leq$" and $\Omega \subseteq \mathbb{R}^K$ is the domain of definition of $V(\bar{s})$. At first let us formulate the definition of the viscosity solution:

Definition 1 (Viscosity solution [17])**.** *The function $V(\bar{s}) : \Omega \to \mathbb{R}$ is called viscosity subsolution (resp. supersolution) of* (2.21) *if for all $\bar{s} \in \Omega$ and all C^2-smooth test functions $\phi(\bar{s})$ such that $V - \phi$ has a local maximum (resp. minimum) at $\bar{s}$ holds:*

$$F\left(\frac{\partial}{\partial \bar{s}}\left(\frac{\partial \phi(\bar{s})}{\partial \bar{s}}\right), \frac{\partial \phi(\bar{s})}{\partial \bar{s}}, V(\bar{s}), \bar{s}\right) \leq 0 \quad \textit{(viscosity subsolution)} \tag{2.22}$$

$$\textit{resp. } F\left(\frac{\partial}{\partial \bar{s}}\left(\frac{\partial \phi(\bar{s})}{\partial \bar{s}}\right), \frac{\partial \phi(\bar{s})}{\partial \bar{s}}, V(\bar{s}), \bar{s}\right) \geq 0 \quad \textit{(viscosity supersolution)} \tag{2.23}$$

The function $V(\bar{s}) : \Omega \to \mathbb{R}$ is called viscosity solution, if it is both viscosity subsolution and viscosity supersolution.

Notice, that if a classical solution exists, it also fulfills the conditions of viscosity solutions. However, the conditions can be also examined for function $V(\bar{s})$ that is not sufficiently smooth everywhere. Therefore, even equations of the form (2.21) that do not posses a classical solution may still posses a viscosity solution. For a deeper understanding of the theory behind the viscosity solutions we refer to [11]. A short intuitive explanation of the concept can be also found in [17].

Let us now present the pioneering convergence theory of Barles and Souganidis [4] that allows us to develop numerical schemes with solution approximations that are guaranteed to converge to the viscosity solution. We assume that the equation $FV(\bar{s}) = 0$ has a viscosity solution and denote this solution simply by $V(\bar{s})$. To find some approximation of this viscosity solution we define a **discrete approximation scheme**

$$Gv(\bar{s}) = G\big(v(\bar{s}), v(\bar{s} + b_1 h), v(\bar{s} + b_2 h), \ldots, v(\bar{s} + b_n h)\big), \tag{2.24}$$

where $v(\bar{s})$, $\bar{s} \in \mathbb{R}^K$ is defined as (possibly) multidimensional function, $b_i \in \mathbb{R}^K$, $i = 1, 2, ..., n$ and $h \in \mathbb{R}^+$.

Let us consider the system of sets called **discretized domains**

$$S_h = \{\bar{s}_i \in \mathbb{R}^K | i = 1, 2, \ldots, N_h\}, \tag{2.25}$$

defined for different values of h, which is often referred as **step-size**.

Definition 2 (Numerical scheme)**.** *The system of equations $Gv(\bar{s}) = 0$ with $\bar{s} \in S_h$ depending on a parameter $h \in \mathbb{R}^+$ is called numerical scheme.*

The numerical scheme is well-defined, if it possess an unique solution. We will assume that this condition is met for any feasible h. By $v(\bar{s})$, we will denote an approximation of the solution of $FV(\bar{s}) = 0$, computed by solving the system of equations $Gv(\bar{s}) = 0$, $\bar{s} \in S_h$. In order to distinguish between approximations with different h, we will sometimes denote $v(\bar{s})$ as $v_h(\bar{s})$.

The monotonicity is an important property that a numerical scheme for solving the non-linear PDEs should have. This property can be seen as a discrete version of the ellipticity. Monotonicity represents an additional constraint for numerical schemes for non-linear PDEs possessing only viscosity solutions in contrast to linear PDEs with classical solutions. In case of the linear PDEs, only the consistency and the stability are needed for the convergence. However, as shown in [17], the approximations computed with non-monotone schemes may converge to wrong solutions in case of nonlinear PDEs.

Definition 3 (Monotonicity)**.** *A discrete approximation scheme*

$$Gv(\bar{s}) = G\big(v(\bar{s}), v(\bar{s} + b_1 h), v(\bar{s} + b_2 h), \ldots, v(\bar{s} + b_n h)\big)$$

is monotone, if the function G is non-increasing in $v(\bar{s} + b_i h)$ for $b_i \neq 0$, $i = 1, \ldots, n$ and increasing in $v(\bar{s})$.

Before defining the consistency in the viscosity sense, we formulate here (for comparison reasons) the classical definition of consistency. This is used by proving the convergence of the numerical schemes for linear PDEs.

Definition 4 (Classical Consistency)**.** *The discrete scheme*

$$GV(\bar{s}) = G\big(V(\bar{s}), V(\bar{s} + b_1 h), V(\bar{s} + b_2 h), \ldots, V(\bar{s} + b_n h)\big)$$

is a consistent *approximation of $FV(\bar{s})$, if $\lim_{h \to 0} \|FV(\bar{s}) - GV(\bar{s})\|_\infty = 0$, where $V(\bar{s})$ is a solution of the equation $FV(\bar{s}) = 0$. Further, $GV(\bar{s})$ is said to be* consistent of order *$p > 0$, if $\|FV(\bar{s}) - GV(\bar{s})\|_\infty = \mathcal{O}(h^p)$, $h \to 0$.*

Now the definition of consistency in the viscosity sense follows:

Definition 5 (Consistency (in the viscosity sense))**.** *The scheme $Gv(\bar{s}) = G(v(\bar{s}), v(\bar{s} + b_1 h), v(\bar{s} + b_2 h), \ldots, v(\bar{s} + b_n h))$ is a consistent approximation of $FV(\bar{s})$, if $\lim_{h \to 0} |F\phi(\bar{s}) - G\phi(\bar{s})| = 0$, for any C^∞-smooth test function $\phi(\bar{s})$.*

A more general definition of consistency can be found in [4]. In this thesis we will call the consistency in viscosity sense simply consistency, in contrast to the classical consistency, that is used in the case of linear PDEs. Let us note, that the order of consistency in the classical sense and in the viscosity sense may be different, as for example in the case of the nine-point stencil from [42]. A scheme is consistent on a numerical domain, if it is consistent in all points of this numerical domain. In such case we will call the scheme

consistent. In the literature, often C^2-smooth test functions are used (as in the Definition 1). However, as shown for example in [33], this leads to an equivalent definition.

The last important property of a convergent numerical scheme is the stability. This property is closely related to the monotonicity, and numerical schemes that are consistent and monotone are in most cases also stable. On the other hand, stable non-monotone schemes are very common: for example the Crank-Nicholson scheme or the numerical schemes with higher order of consistency used for solving linear PDEs.

Definition 6 (Stability)**.** *The numerical scheme defined by the system of equations* $Gv_h(\bar{s}) = 0$, $\bar{s} \in S_h$ *with solution* $v_h(\bar{s})$ *is stable, if there exists some constant* C *so that* $\|v_h(\bar{s})\|_\infty < C$, $\forall h > 0$.

The following Theorem of Barles and Souganidis [4] is the key tool for proving convergence of a numerical scheme approximating a nonlinear PDE:

Theorem 1 (Barles-Souganidis [4])**.** *If the equation* $FV(\bar{s}) = 0$ *satisfies the strong uniqueness property (see [4]) and if the numerical scheme* $Gv_h(\bar{s}) = 0$, $\bar{s} \in S_h$ *approximating the equation* $FV(\bar{s}) = 0$ *is monotone, consistent and stable, its solution* $v_h(\bar{s})$ *converges locally uniformly to the solution* $V(\bar{s})$ *of* $FV(\bar{s}) = 0$ *with* $h \to 0$.

Remark 1. *The above mentioned strong uniqueness property [4] is a property of the problem and not of the numerical scheme. Therefore, we will simply assume that our problem possess this property without actually proving it.*

Chapter 3

Finite difference numerical methods

The goal of this chapter is to present some standard approaches of solving the one-dimensional HJB equation, as well as their limitations. In Section 3.1 we will present two widely used finite-difference methods from [16], [48]. In Section 3.2, we will discuss the fact, that the higher order schemes for solving the HJB equations are not monotone and therefore might not converge. The Section 3.2 is based on the paper [26].

3.1 Standard finite difference methods

For convenience we repeat here the HJB equation to be solved:

$$\frac{\partial V}{\partial t} + \max_{\theta \in \Theta} \left(\frac{\sigma(\cdot)^2}{2} \frac{\partial^2 V}{\partial s^2} + \mu(\cdot) \frac{\partial V}{\partial s} + r(\cdot) V + f(\cdot) \right) = 0, \tag{3.1}$$

$$V(s,T) = V_T(s), \tag{3.2}$$

$$0 < t < T, \quad s \in \mathbb{R},$$

where $\sigma(\cdot)$, $\mu(\cdot)$, $r(\cdot)$, $f(\cdot)$ are functions of s, t, θ and $V_T(s)$ is the terminal condition. Now we can turn to the discretization of this equation. We suppose Θ to be discrete, i.e. $\Theta = \{\theta_1, \theta_2, \ldots, \theta_Q\}$.

3.1.1 Discretization of the Hamilton-Jacobi-Bellman equation

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

We must ensure that this discretization will be monotone. We denote the discretized HJB equation in point (s_i, t_j) as $G_{i,j}(v_i^j, v_{i_1}^{j_1}, v_{i_2}^{j_2}, \ldots, v_{i_K}^{j_K})$ where $(i_k, j_k) \neq (i, j)$, $\forall k = 1, 2, \ldots, K$. Then this scheme is monotone, if the function G is non-increasing in $v_{i_1}^{j_1}, v_{i_2}^{j_2}, \ldots, v_{i_K}^{j_K}$. 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 monotone discretization introduced by Wang and Forsyth [48], that can be consistent of second order in space in an ideal case:

$$
\begin{aligned}
\frac{\partial V}{\partial t}(s_i,t_j) &\approx \frac{v_i^{j+1}-v_i^j}{\Delta_j t},\\
\frac{\partial^2 V}{\partial s^2}(s_i,t_j) &\approx \frac{v_{i-1}^j-2v_i^j+v_{i+1}^j}{\Delta_{i-1}s\Delta_i s},\\
\frac{\partial V}{\partial s}(s_i,t_j) &\approx D_1(v_{i-1}^j,v_i^j,v_{i+1}^j), \quad \text{where}\\
D_1(v_{i-1}^j,v_i^j,v_{i+1}^j) &= \begin{cases} \dfrac{v_{i+1}^j-v_{i-1}^j}{\Delta_{i-1}s+\Delta_i s}, \text{ if } \dfrac{b(s_i,t_j,\theta)}{\Delta_{i-1}s+\Delta_i s} \le \dfrac{a(s_i,t_j,\theta)}{\Delta_{i-1}s\Delta_i s} \\ \dfrac{(\xi+1)v_{i+1}^j-2\xi v_i^j+(\xi-1)v_{i-1}^j}{(1+\xi)\Delta_i s+(1-\xi)\Delta_{i-1}s}, \ \xi=\operatorname{sign}(b(s_i,t_j,\theta)), \text{ else.}\end{cases}
\end{aligned}
$$

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

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

where

$$
\begin{aligned}
L_{i,j,\theta}v^j = \frac{\sigma^2(s_i,t_j,\theta)}{2}\frac{v_{i-1}^j-2v_i^j+v_{i+1}^j}{\Delta_{i-1}s\Delta_i s} + \mu(s_i,t_j,\theta)D_1(v_{i-1}^j,v_i^j,v_{i+1}^j)\\
+ r(s_i,t_j,\theta)v_i^j + f(s_i,t_j,\theta). \quad (3.3)
\end{aligned}
$$

Next we will present two different algorithms based on this discretization. In both algorithms we will compute the values v_i^j from the terminal time layer t_M back to the initial time layer t_0. That means, we will at first compute all values in the time layer t_{j+1} before proceeding to the time layer t_j. We will need the values in the last time layer t_M as a terminal condition. Moreover, we will need some boundary conditions (BCs). In our case, we will use the Dirichlet boundary conditions that preserve the monotonicity. Therefore, instead of equation (3.3) in nodes (s_0,t_j), (s_N,t_j) we will use the simple equations

$$v_0^j = BC_L(s_0,t_j), \quad v_N^j = BC_R(s_N,t_j), \tag{3.4}$$

with some predefined functions $BC_L(s,t)$, $BC_R(s,t)$.

3.1.2 Classical implicit FDM with policy iteration

First we will introduce a standard FDM algorithm widely used to solve the HJB equations. It's similar to the classical implicit method for solving convection-diffusion equations, however in order to find the optimal control, the policy iteration is needed. In this context we understand under the term policy the $(N+1)$-dimensional vector of controls used in one time layer with $(N+1)$ nodes.

Algorithm 1 Classical implicit FDM with policy iteration

v^M is determined by the terminal condition.
for $j = M-1, M-2, \ldots, 0$ **do**
  set $v^{(0)} = v^{j+1}$, $k = 0$
  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:
    $v_i^{(k)} = v_i^{j+1} + \Delta_j t L_{i,j,\theta_i^{(k)}} v^{(k)} \quad \text{for} \quad i \in \{1, \ldots, N-1\}$
    $v_0^{(k)} = BC_L(s_0, t_j), \quad v_N^{(k)} = BC_R(s_N, t_j)$
  until $\|v^{(k)} - v^{(k-1)}\|_2 < TOL$
  $v^j = v^{(k)}, \quad \bar{\theta}_i^j = \theta_i^{(k)} \quad \text{for} \quad i \in \{0, 1, \ldots, N\}.$
end for

Let us note, that it is possible to find the optimal control $\theta_i^{(k)}$ analytically, however we search here for the optimal control $\theta_i^{(k)}$ by simply 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 $\bar{\theta}_i^j$ is called *optimal control policy* in time layer j.

3.1.3 Piecewise constant policy timestepping method

The second algorithm that we will present here is the so-called *piecewise constant policy timestepping* (PCPT) method. It is described for example in the papers [30], [16], [41]. 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 s_i the optimal control leading to the biggest value. The following Algorithm 2 will clarify this approach.

Algorithm 2 PCPT method

v^M is determined by the terminal condition.
for $j = M-1, M-2, \ldots, 0$ **do**
  for $q = 1, 2, \ldots, Q$ **do**
    Solve system of equations:
    $v_i^{(q)} = v_i^{j+1} + \Delta_j t L_{i,j,\theta_q} v^{(q)} \quad \text{for} \quad i \in \{1, \ldots, N-1\}$
    $v_0^{(q)} = BC_L(s_0, t_j), \quad v_N^{(q)} = BC_R(s_N, t_j)$
  end for
  $\kappa_i = \arg\max_q v_i^{(q)}, \quad v_i^j = v_i^{(\kappa_i)}, \quad \bar{\theta}_i^j = \theta_{\kappa_i}, \quad \text{for} \quad i \in \{0, 1, \ldots, N\}.$
end for

3.2 Non-Existence of higher order monotone approximation schemes

The standard methods presented in the previous section are consistent of order 1 in time and up to 2 in space. Therefore, a question arises, if we can't solve the HJB equation with some higher order scheme. In this section we will prove that there is no monotone scheme consistent of order higher than 2. This section is based on the paper [26].

At first, we will start with examining a general two-dimensional differential operator. Let $V(x,y) : \mathbb{R}^2 \to \mathbb{R}$ be a locally C^2-function ($x, y \in \mathbb{R}$ are one-dimensional). We define the differential operator $\mathcal{L} : C^2(\mathbb{R}^2) \to C(\mathbb{R}^2)$

$$\mathcal{L}V(x,y) = \alpha_1 \frac{\partial^2 V}{\partial x^2} + \alpha_{12} \frac{\partial^2 V}{\partial x \partial y} + \alpha_2 \frac{\partial^2 V}{\partial y^2} + \beta_1 \frac{\partial V}{\partial x} + \beta_2 \frac{\partial V}{\partial y} + \gamma V. \tag{3.5}$$

We assume $\alpha_1 \neq 0$ and investigate some properties of the linear operator $L : C^2(\mathbb{R}^2) \to C(\mathbb{R}^2)$ given by

$$\begin{aligned} LV(x,y) = a_0(h)V(x,y) + a_1(h)V(x + b_1 h, y + c_1 h) \\ + a_2(h)V(x + b_2 h, y + c_2 h) + \cdots + a_n(h)V(x + b_n h, y + c_n h), \end{aligned} \tag{3.6}$$

where $b_i \neq 0$, or $c_i \neq 0$, $i = 1, 2, \ldots, n$ and there exist j, k such that $b_j \neq 0$, $c_k \neq 0$. (3.6) should be an approximation of the differential operator $\mathcal{L}V(x,y)$.

Definition 7 (Positive coefficients approximation [17])**.** *The linear discrete approximation scheme* (3.6) *satisfies the positive coefficients condition if* $a_i(h) \geq 0$ *for* $i = 1, 2, \ldots, n$*, for all* $h > 0$*.*

Often a scheme is monotone, if and only if its linear part satisfies the positive coefficient condition.

3.2.1 Main Results

Theorem 2. *There exist no discrete linear approximation* $LV(x)$ *of* $\mathcal{L}V(x)$ *satisfying the positive coefficients condition which is consistent (in the viscosity sense) of order higher than* 2*.*

Proof. We rewrite $L\phi(x,y)$ in the form of a Taylor expansion up to order m:

$$\begin{aligned} L\phi(x,y) &= a_0(h)\phi(x,y) + a_1(h)\phi(x + b_1 h, y + c_1 h) \\ &\quad + a_2(h)\phi(x + b_2 h, y + c_2 h) + \cdots + a_n(h)\phi(x + b_n h, y + c_n h) \\ &= a_0(h)\phi(x,y) + \sum_{i=1}^{n} a_i(h)(\phi(x,y) + \frac{1}{1!}\sum_{j=0}^{1} \binom{1}{j} \frac{\partial^1 \phi}{\partial x^{1-j} \partial y^j} (b_i h)^{1-j} (c_i h)^j \\ &\quad + \cdots + \frac{1}{m!}\sum_{j=0}^{m} \binom{m}{j} \frac{\partial^m \phi}{\partial x^{m-j} \partial y^j} (b_i h)^{m-j} (c_i h)^j) + \mathcal{O}(h^{m+1}). \end{aligned} \tag{3.7}$$

For an approximation of order p we have $\|\mathcal{L}\phi(x,y) - L\phi(x,y)\|_\infty = \mathcal{O}(h^p)$. Using the expansion (3.7), this yields the matrix equation

$$\begin{pmatrix} 1 & 1 & 1 & \cdots & 1 \\ 0 & b_1h & b_2h & \cdots & b_nh \\ 0 & c_1h & c_2h & \cdots & c_nh \\ 0 & \frac{(b_1h)^2}{2} & \frac{(b_2h)^2}{2} & \cdots & \frac{(b_nh)^2}{2} \\ 0 & b_1c_1h^2 & b_2c_1h^2 & \cdots & b_nc_nh^2 \\ 0 & \frac{(c_1h)^2}{2} & \frac{(c_2h)^2}{2} & \cdots & \frac{(c_nh)^2}{2} \\ 0 & \frac{(b_1h)^3}{3!} & \frac{(b_2h)^3}{3!} & \cdots & \frac{(b_nh)^3}{3!} \\ \vdots & & & & \vdots \\ 0 & \frac{(c_1h)^m}{m!} & \frac{(c_2h)^m}{m!} & \cdots & \frac{(c_nh)^m}{m!} \end{pmatrix} \cdot \begin{pmatrix} a_0(h) \\ a_1(h) \\ a_2(h) \\ \vdots \\ a_n(h) \end{pmatrix} = \begin{pmatrix} \gamma \\ \beta_1 \\ \beta_2 \\ \alpha_1 \\ \alpha_{12} \\ \alpha_2 \\ \mathcal{O}(h^p) \\ \vdots \\ \mathcal{O}(h^p) \end{pmatrix}. \tag{3.8}$$

We can write (3.8) shortly as $A(h)a(h) = g(h)$. Let us look at the fourth row of the system $A(h)a(h) = g(h)$:

$$a_1(h)\frac{(b_1h)^2}{2} + a_2(h)\frac{(b_2h)^2}{2} + \cdots + a_n(h)\frac{(b_nh)^2}{2} = \alpha_1. \tag{3.9}$$

The right-hand side is of order $\mathcal{O}(h^0)$, so should be the left hand side. Therefore, at least one $a_i(h)$ should be of order $\mathcal{O}(h^k)$, $k \leq -2$ such that $b_i \neq 0$. If for all $b_i \neq 0$, $a_i(h) = \mathcal{O}(h^j)$, $j > -2$ holds, then each non-zero term of the left-hand side of (3.9) is of order $h^2\mathcal{O}(h^j) = \mathcal{O}(h^{2+j})$, where $2 + j > 0$, so the whole left hand side is of order greater than zero.

Now let us assume that we have a solution of $A(h)a(h) = g(h)$ for $p > 2$ satisfying the positive coefficients condition. We consider the 11th row of (3.8):

$$a_1(h)\frac{(b_1h)^4}{4!} + a_2(h)\frac{(b_2h)^4}{4!} + \cdots + a_n(h)\frac{(b_nh)^4}{4!} = \mathcal{O}(h^p). \tag{3.10}$$

As we noted, there exists an i such that $b_i \neq 0$ and $a_i(h) = \mathcal{O}(h^k)$, $k \leq -2$. Then, also the term in (3.10) $a_i(h)\frac{(b_ih)^4}{4!}$ is of order $\mathcal{O}(h^q)$, $q = k + 4 \leq 2$. Due to the positive coefficients condition, each term of (3.10) is non-negative and hence also the whole left-hand side of (3.10) will be of order $\mathcal{O}(h^c)$, $c \leq 2$. However, the right hand side should be of order higher than 2, which leads to a contradiction. □

Remark 2. *The proof of the Theorem 2 does not take into account the case of schemes without a node in x itself. However, this can be seen as a subcase of the above schemes with fixed $a_0(h) = 0$.*

Remark 3. *The proof for a higher dimensional function V, with the corresponding second order PDE operator $\mathcal{L}V$ can be done in a similar manner.*

Remark 4. *In the case of a linear differential operator with derivatives of order higher than 2 similar results on the non-existence may be feasible, with higher maximal order of consistency (in the viscosity sense).*

Let us define

$$\mathcal{L}_\theta V(x,y) = \alpha_1(\theta)\frac{\partial^2 V}{\partial x^2} + \alpha_{12}(\theta)\frac{\partial^2 V}{\partial x \partial y} + \alpha_2(\theta)\frac{\partial^2 V}{\partial y^2} + \beta_1(\theta)\frac{\partial V}{\partial x} + \beta_2(\theta)\frac{\partial V}{\partial y} + \gamma(\theta)V, \quad (3.11)$$

where θ is a parameter, $x, y \in \mathbb{R}$. We now formulate the main result of Chapter 3.

Theorem 3. *There exist no monotone discrete approximation*

$$-\max_{\theta\in\Theta}\big(L_\theta V(x,y) + \delta(\theta)\big) \quad \textit{of} \quad -\max_{\theta\in\Theta}\big(\mathcal{L}_\theta V(x,y) + \delta(\theta)\big)$$

consistent (in the viscosity sense) of order higher than 2.

Proof. Since the maximum is a non-decreasing function, $L_\theta V(x,y)$ has to satisfy the positive coefficients condition so that $-\max_{\theta\in\Theta}(L_\theta V(x,y) + \delta(\theta))$ will be monotone. Then,

$$-\max_{\theta\in\Theta}\big(\mathcal{L}_\theta V(x) + \delta(\theta)\big) = -\max_{\theta\in\Theta}\big(L_\theta V(x,y) + \mathcal{O}(h^k) + \delta(\theta)\big) = -\max_{\theta\in\Theta}\big(L_\theta V(x,y) + \delta(\theta)\big) + \mathcal{O}(h^k),$$

where according to Theorem 2, $k \le 2$. □

Remark 5. *The non-existence of higher order monotone discrete approximations of* $f\big(\mathcal{L}V(x,y)\big)$ *for a monotone non-increasing function* f *can be proven in the same way as in Theorem 3.*

3.2.2 Application of the results to the HJB equation

Now we apply this result to the HJB equation

$$\frac{\partial V(s,t)}{\partial t} + \max_{\theta\in\Theta}\Big(\alpha(s,t,\theta)\frac{\partial^2 V}{\partial s^2} + \beta(s,t,\theta)\frac{\partial V}{\partial s} + \gamma(s,t,\theta)V + \delta(s,t,\theta)\Big) = 0, \quad (3.12)$$

with one space dimension. The coefficients α, β, γ, δ depend on θ as well as on s and t. However, in each particular time and space, we can treat them as constants with respect to s, t. This allow us to write the HJB equation (3.12) in the form

$$\max_{\theta\in\Theta}\Big(-\frac{\partial V}{\partial t} + \alpha(\theta)\frac{\partial^2 V}{\partial s^2} + \beta(\theta)\frac{\partial V}{\partial s} + \gamma(\theta)V + \delta(\theta)\Big) = 0 \quad (3.13)$$

for any particular values of t and s. Now, Theorem 3 applied on the left hand side of (3.13) with $y := t$, $x := s$ and

$$\mathcal{L}_\theta V(s,t) = -\frac{\partial V}{\partial t} + \alpha(\theta)\frac{\partial^2 V}{\partial s^2} + \beta(\theta)\frac{\partial V}{\partial s} + \gamma(\theta)V$$

states, that we cannot obtain a monotone discrete linear scheme for the HJB equation (3.13) consistent of order higher than 2 in the viscosity sense.

Remark 6. *As noted in Remark 3, Theorem 3 can be proved also for higher dimensions. Therefore, the same result can be obtained in the case of HJB equations with more space dimensions.*

In this section we showed that we cannot apply the convergence theory [4] to prove the convergence of linear discrete schemes which are consistent of order higher than 2 (in the viscosity sense), since this theory relies on the monotonicity of the scheme.

We used the Definition 5 of the consistency (viscosity sense) because for HJB equations the standard Definition 4 cannot be used. For linear PDEs, also consistency in the classical sense of higher order is feasible. A typical example is a monotone nine-point stencil for the Poisson equation [42], which is $\mathcal{O}(h^4)$-consistent in the classical sense. An interesting question that remains is, if any monotone scheme for the linear part of the HJB equation $-\frac{\partial V(x,t)}{\partial t} + \mathcal{L}_\theta V(x,t)$ being consistent of order higher than 2 in the sense of Definition 4 exists.

Chapter 4

Piecewise predicted policy timestepping method

In the previous chapter we proved that we cannot obtain a monotone scheme with order of convergence higher than 2. Therefore, in this chapter we will try to make the convergence faster (in means of computational time, not in the means of convergence order) by reduction of the number of possible controls in our algorithms. We introduce here the *piecewise predicted policy timestepping* (PPPT) method, that can be seen as a modification of the PCPT method from the previous Chapter 3. This chapter is based on the paper [25].

4.1 Main idea and algorithm

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 an approximation of the solution on a coarse grid with some numerical method, and now we want to compute a better approximation on a 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 the PCPT method will be redundant. Therefore, the idea of the piecewise predicted policy timestepping (PPPT) method 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. We should note, that 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 s,t or a one-dimensional function of variable s that assigns a value of control to any point (s,t) on the computational domain, or (in the one-dimensional case) to any point s in the particular time-layer.

As well as in the 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.

The control policies in a particular time layer are chosen in such manner, that they cover in each node all "predicted" controls for that node. These "predicted" controls are chosen as either controls that are between the two "predictor" controls for that node, or are neighbors of the "predictor" controls. Here, the "predictor" controls in each space node are the approximations of the optimal controls in that node from the nearest earlier and nearest later time layer. Figure 4.1 illustrates the choice of control policies in the PPPT method in a particular time layer as well as the comparison with the constant control policies used in the PCPT method in each time layer. The exact construction of the control policies in PPPT method is described in Algorithm 3. The computation of the approximation of solution of the HJB equation on a fine grid using these predicted control policies is described in Algorithm 4.

Convergence: By defining new controls z from the predicted control policy functions $\theta^z(s,t)$ (see algorithm 4), we can consider the PPPT method to be a PCPT method for these new controls. Therefore, stability of the PPPT method is a consequence of the stability of the PCPT method. However, the answer on the question if the method converges to the solution of the HJB equation depends on whether the the true optimal control in each node really belongs to the set of predicted controls. As this cannot be guaranteed, the method is rather heuristic, however seem to be converging in the numerical examples presented in the following sections.

Possible generalizations: It is of course possible to create more predicted control policies, e.g. by taking more neighbors of predictor controls. Also, one may use similar control prediction ideas also with classical implicit FDM or with the Tree-Grid methods presented in the later chapters.

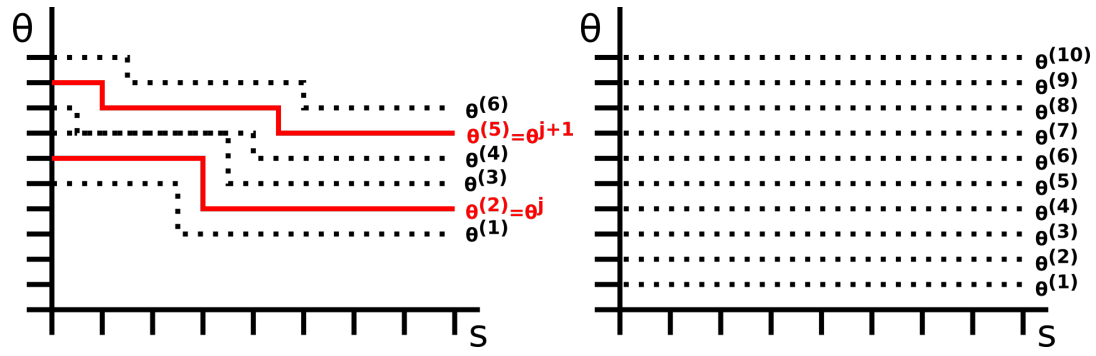

Figure 4.1: Illustration of the control policies in the PPPT method (left) and in the PCPT method (right). The full red lines in the left figure represent approximations of the optimal control policies in the j-th and $(j+1)$-th time layer of the coarse grid (predictor control policies) and together with the dotted lines also the control policies to be tested in all time layers of the fine grid lying between the j-th and $(j+1)$-th coarse grid time layers. On the right figure, dotted lines represent constant control policies to be tested in all time layers in case of the PCPT method.

The Algorithm 3 for construction of the predicted control policy functions follows:

Algorithm 3 Construction of the predicted control functions for PPPT method

Solve HJB PDE with classical or PCPT method on a coarse grid. The by-product of this solution should be the array of optimal controls $\tilde{\theta}_i^j$ for all nodes (s_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.
Define control indices $\tilde{z}_i^j$, such that $\tilde{\theta}_i^j = \theta_{\tilde{z}_i^j}$
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 $\tilde{I} = \{1, 2, \ldots, \tilde{N}\} \times \{1, 2, \ldots, \tilde{M}-1\}$
Define $\tilde{M}$ one-dimensional index functions in the layers of the coarse grid:
for $j = 1, 2, \ldots, \tilde{M}$ **do**
    $\tilde{z}^j(s) = \tilde{s}_i^j$ where $i = \arg \min\limits_{k\in\{0,1,\ldots,\tilde{N}\}} \|s - s_k\|_\infty$
end for
Define:
$up(s,t) = \tilde{z}^j(s)$ where $j = \arg\min_{k\in\{1,2,\ldots,\tilde{M}\}, t\le t_k} |t - t_k|$
$down(s,t) = \tilde{z}^j(s)$ where $j = \arg\min_{k\in\{1,2,\ldots,\tilde{M}\}, t\ge t_k} |t - t_k|$
Define $Z-2$ two-dimensional index functions:
for $z = 1, 2, \ldots, Z-2$ **do**
    $\tilde{z}^z(s,t) = round\left(\frac{z-1}{Z-3} up(s,t) + \frac{Z-2-z}{Z-3} down(s,t)\right)$
end for
Determine neighbor index functions:
$\tilde{z}^{Z-1}(s,t) = \min\left(\max_{z\in\{1,2,\ldots,Z-2\}} \tilde{z}^z(s,t), J\right)$
$\tilde{z}^{Z}(s,t) = \max\left(\min_{z\in\{1,2,\ldots,Z-2\}} \tilde{z}^z(s,t), 1\right)$
Create control functions:
for $z = 1, 2, \ldots, Z$ **do**
    $\theta^z(s,t) = \theta_{\tilde{z}^z(s,t)}$
end for

The Algorithm 4 of the PPPT method using the predicted control policy functions follows. The discrete operator $L_{i,j,\theta}v$ was defined in Section 3.1.1.

Algorithm 4 PPPT method

v^M is determined by the terminal condition.
for $j = M-1, M-2, \ldots, 0$ **do**
    for $z = 1, 2, \ldots, Z$ **do**
        Define $\bar{\theta}^{j,z}$: $\bar{\theta}_i^{j,z} = \theta^z(s_i, t_j)$ $(i \in \{0, 1, 2, \ldots, N\}$
        Solve system of equations:
        $v_i^{(z)} = v_i^{j+1} + \Delta_j t L_{i,j,\bar{\theta}_i^{j,z}} v^{(z)} \quad for \quad i \in \{1, \ldots, N-1\}$
        $v_0^{(z)} = BC_0(s_0, t_j), \quad v_N^{(z)} = BC_N(s_N, t_j)$
    end for
    $\kappa_i = \arg\max_k v_i^{(z)}, \quad v_i^j = v_i^{(\kappa_i)}, \quad \bar{\theta}_i^j = \theta_{\kappa_i}, \quad i \in \{0, 1, \ldots, N\}.$
end for

4.2 Numerical example: mean-variance optimal investment problem

In this section we will compare the performance of the classical implicit FDM, PCPT scheme and the PPPT scheme on a numerical example from finance. For comparison and verification reasons, we will take the whole example with boundary conditions, and up to small changes also with the discretization, from [41]. For reader's convenience, we repeat here the main characteristics of the problem.

Example 1 (Mean-variance optimal investment problem)**.** *We will start with a problem of dynamic investment allocation between a stock and a risk-free asset in an mean-variance framework. Let our stock follow a SDE of the form:*

$$dS_t = (r + \xi\sigma)S_t dt + \sigma S_t dW_t, \tag{4.1}$$

where S_t is a stock price process, r is the risk-free interest rate, σ is the volatility, ξ is the market price of risk and w_t is the wealth process. Moreover, we will suppose that the investor contributes to the portfolio at a constant rate π. Then, our task is to solve the following problem:

$$\max_{\theta\in\Theta} \big(\mathbb{E}_{t=0}(w_T) - \lambda Var_{t=0}(w_T)\big), \tag{4.2}$$

$$dw_t = \big((r + \theta(w_t, t)\xi\sigma)w_t + \pi\big)dt + \theta(w_t, t)\sigma w_t dW_t, \tag{4.3}$$

$$w_0 = K, \tag{4.4}$$

where λ is the investors coefficient of risk aversion (or also Lagrange multiplier similar as in the Markowitz model, see [37]), $\theta(w_t, t)$ is the proportion of investors wealth invested in the stock in time t for a current wealth w_t, P is the set of all admissible functions $\theta(w_t, t)$, K is some constant representing the terminal wealth and T is the final time.

For different values of λ, we expect different $\mathbb{E}_{t=0}w_T$. The set of all possible pairs $(\lambda, \mathbb{E}_{t=0}w_T)$ will be called "efficient frontier". Note that often also set of pairs $(\mathbb{E}_{t=0}w_T, Var_{t=0}w_T)$ is referred as efficient frontier [37]. In [49], it is explained, how to compute pairs on this efficient frontier numerically. The main part of that problem is solving the HJB equation in the following form:

$$\frac{\partial V}{\partial t} + \min_{\theta\in[\theta_{min},\theta_{max}]} \mathcal{L}_\theta V = 0, \tag{4.5}$$

$$\mathcal{L}_p V = \frac{1}{2}\sigma^2\theta^2 w^2 \frac{\partial^2 V}{\partial w^2} + \big(\pi + (r + \theta\sigma\xi)w\big)\frac{\partial V}{\partial w}, \tag{4.6}$$

$$V(w, T) = \Big(w - \frac{\gamma}{2}\Big)^2, \tag{4.7}$$

where γ is a parameter given in advance, and dependent on the unknown pair $(\lambda, \mathbb{E}_{t=0}w_T)$. As solution, we will get besides the value function $V(w, t)$ also the optimal control $\theta(w, t)$ which is the optimal investment strategy for the unknown value λ. This value of λ, also with $\mathbb{E}_{t=0}w_T$ can be computed afterwards, using the optimal investment strategy $\theta(w, t)$. For more details see [49]. Here, we will be concerned with solving the HJB equation (4.5)-(4.7) *numerically.*

Computational domain: For our problem we will use an equidistant discretization of

the domain $[0, w_N] \times [0, T]$, with M time-steps of size Δt and N space-nodes, with distance between 2 neighboring nodes Δw. We use $w_N = 5$ and $T = 20$.

Terminal and boundary conditions: Let us assume a positive money inflow rate π. Then, as the left boundary is in $w_0 = 0$, we need no boundary condition, as the second derivative from (4.6) vanishes and as π is positive we do not need any data from outside the domain to approximate the first derivative. For the right boundary condition, we will use an approximation from [41] in form of a Dirichlet boundary condition:

$$V(w_N, \tau) = \frac{1}{2}\alpha(\tau) w_N^2 + \beta(\tau) w_N + \delta(\tau), \tag{4.8}$$

where

$$\begin{aligned}
\tau &= T - t, \\
\alpha(\tau) &= \exp\big((a^2 + 2b)\tau\big), \\
\beta(\tau) &= -(\gamma + c)\exp(b\tau) + c\exp\big((a^2 + 2b)\tau\big), \\
\delta(\tau) &= -\frac{\pi(\gamma + c)}{b}\big(\exp(b\tau) - 1\big) + \frac{\pi c}{a^2 + b}\big(\exp\big((a^2 + 2b)\tau\big) - 1\big) + \frac{\gamma^2}{4}, \\
c &= 2\pi/(a^2 + b), \\
a &= \sigma\theta, \\
b &= r + \theta\sigma\xi.
\end{aligned}$$

The terminal condition is defined by equation (4.7).

Numerical results: For solving mean-variance optimal investment problem we implemented the classical implicit method, the PCPT method and the PPPT method. We implemented all methods in Matlab and tested on an Intel Core i7-4770 CPU 3.40GHz computer with 8 GB RAM. For comparison reasons, we used the same parameter values as in [41]: $r = 0.03$, $\sigma = 0.15$, $\xi = 0.33$, $\pi = 0.1$, $\gamma = 14.47$. We use the time-step size $\Delta t = h_k$ and space-step size $\Delta w = 0.25 h_k$ where h_k is a discretization parameter. We use the control set $\Theta = [0, 1.5]$ equidistantly discretized on 31 different controls $0, 0.05, 0.1, \ldots, 1.5$.

In Table 4.1 we can see the results of the numerical simulations. We tested the PCPT method, the classical implicit method and two PPPT methods with a different approach for predictions. We ran the methods with $h_k = 2^{1-k}$, $k = 1, 2, \ldots, 10$. To compute the prediction of controls (control functions) for the PPPT methods, we used the PCPT scheme. In our first approach we have done the predictions on a grid with mesh size $h = 2^{-4}$ (that is even finer than some of the main algorithm grids). This PPPT method is denoted as PPPT (1).

To verify our results, we checked the values of the solutions in $t = 0, w = 1$, which are also computed in [41]. This values are denoted as **Val**. We will estimate the error of the approximation **Err** with values from the final time layer A^k (computed with stepsize h_k). The exact solution is not known, therefore we use an approximation of the final time layer A^{11} computed with the classical implicit method and $h = 2^{-10}$ as reference solution. The formula for estimating the error is

$$Err\ \ A^k = \|A^k - A^{11}\|_2. \tag{4.9}$$

However, in [41] the error is estimated as absolute value of the difference of the values of solutions in $t = 0, w = 1$ computed with stepsizes h_{k-1} and h_k. As this pointwise error is dependent only on the values Val, we will denote it as **Err(Val)**.

The **experimental order of convergence** (**EOC**) will be computed as

$$EOC\ \ A^k = \frac{\log(ErrA^{k-1}) - \log(ErrA^k)}{\log(h_{k-1}) - \log(h_k)}. \tag{4.10}$$

We will compute the experimental order of convergence using the above formula also with error estimation Err(Val) and denote this experimental order of convergence as **EOC(Val)**. The computational time of each method (depending on the computer), is denoted as **Time**. The time is measured in seconds and is just informative, as the value varies with each new run. In case of the PPPT method, the time needed to compute the prediction of the controls is added.

In the left plot of Figure 4.2 we show the logarithm of the computational time against the logarithm of the error, to see how much time we need for each method to get the same level of accuracy. We observe, that the PPPT (1) method is slower on a low level of accuracy at first, what is caused by relatively high time-costs spent on computing prediction of control in contrast to fast low-accuracy PCPT and classical implicit method. For higher levels of accuracy that are more time-demanding for classical and PCPT methods, the prediction already saves computational time and the PPPT (1) method is most effective.

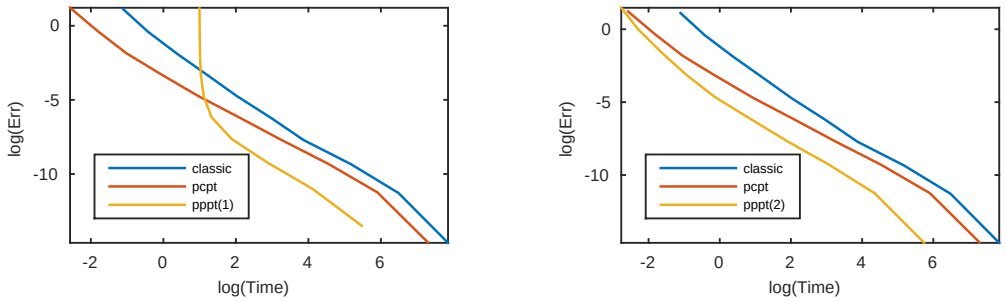

Figure 4.2: Comparison of the classic and the PCPT methods (both plots) with the PPPT (1) method on the left plot and PPPT (2) method on the right plot. In the PPPT (1) method prediction is computed with the discretization parameter $h = 2^{-4}$ and in the PPPT (2) method the prediction is computed with $h = 4h_k$ where h_k is discretization parameter of the fine grid.

The previous analysis of results leads us to an idea of running a PPPT method with different prediction on each refinement level, so that the prediction grid will be adjusted to the desired level of accuracy. To do so, we will compute for each refinement level $k = 1, 2, \ldots, 10$ new prediction of controls with a PCPT scheme with $h = 4h_k$. Results of this approach to the PPPT scheme is in Table 4.1 denoted as PPPT (2). The right plot of Figure 4.2 illustrates the dependence of computational time and accuracy for this case. We see, that this PPPT (2) method is allways the most efficient one. For higher levels of accuracy it is 4.7 times faster than PCPT and 8 times faster than the classical implicit method, what is a significant speed-up.

Table 4.1: Error, experimental order of convergence, computational time, value, pointwise error and pointwise convergence in $t = 0, w = 1$ for classical implicit method, PCPT and PPPT methods, for different values of the discretization parameters h_k

h_k	1	2^{-1}	2^{-2}	2^{-3}	2^{-4}	2^{-5}	2^{-6}	2^{-7}	2^{-8}	2^{-9}
Classical										
Err	3,03E+00	6,66E-01	1,51E-01	3,61E-02	8,58E-03	2,02E-03	4,48E-04	8,82E-05	1,26E-05	4,36E-07
EOC		2.185	2.146	2.058	2.075	2.085	2.175	2.344	2.809	4.852
Time	0.33	0.65	1.47	3.35	7.67	19.34	47.86	176.36	655.58	2570.57
Val	2.783	2.025	1.769	1.648	1.589	1.561	1.546	1.540	1.536	1.534
Err(Val)		0.7581	0.2558	0.1210	0.0586	0.0283	0.0144	0.0069	0.0034	0.0017
EOC(Val)			1.568	1.080	1.045	1.051	0.970	1.064	1.007	1.000
PCPT										
Err	3,37E+00	7,13E-01	1,58E-01	3,77E-02	8,91E-03	2,10E-03	4,64E-04	9,17E-05	1,32E-05	4,51E-07
EOC		2.239	2.171	2.071	2.081	2.087	2.175	2.340	2.798	4.871
Time	0.08	0.16	0.36	0.94	2.58	8.00	25.38	93.19	365.59	1475.30
Val	3.050	2.109	1.799	1.660	1.595	1.564	1.548	1.540	1.536	1.535
Err(Val)		0.9409	0.3092	0.1392	0.0654	0.0312	0.0158	0.0076	0.0038	0.0019
EOC(Val)			1.606	1.152	1.089	1.067	0.983	1.064	1.008	1.001
PPPT(1)										
Err	3,20E+00	6,85E-01	1,54E-01	3,64E-02	8,61E-03	2,09E-03	4,76E-04	9,88E-05	1,65E-05	1,38E-06
EOC		2.223	2.153	2.081	2.080	2.045	2.130	2.269	2.585	3.576
Time	2.71	2.72	2.74	2.81	3.05	3.78	6.61	17.55	61.57	238.09
Val	2.893	2.045	1.776	1.649	1.590	1.563	1.548	1.541	1.537	1.536
Err(Val)		0.8477	0.2691	0.1274	0.0591	0.0271	0.0144	0.0072	0.0036	0.0018
EOC(Val)			1.656	1.078	1.107	1.128	0.906	1.006	0.998	0.980
PPPT(2)										
Err	4,35E+00	9,92E-01	1,88E-01	4,20E-02	9,45E-03	2,16E-03	4,76E-04	9,28E-05	1,31E-05	4,43E-07
EOC		2.133	2.402	2.160	2.151	2.127	2.183	2.360	2.826	4.885
Time	0.06	0.10	0.20	0.40	0.88	2.35	6.53	21.48	77.60	310.84
Val	3.571	2.594	1.875	1.688	1.599	1.564	1.548	1.540	1.536	1.534
Err(Val)		0.9768	0.7193	0.1868	0.0883	0.0353	0.0159	0.0080	0.0039	0.0019
EOC(Val)			0.442	1.945	1.081	1.321	1.150	0.993	1.029	1.073

Table 4.2: Parallelized methods, computational time (2 time steps) and speed-up in % in comparison to non-parallelized methods.

# Space nodes:	5×10^1	5×10^2	5×10^3	5×10^4	5×10^5
Classic implicit					
-time	0.321	0.337	0.631	3.886	49.488
-% speed-up	-401	-171	13	48	46
PCPT					
-time	0.054	0.060	0.112	0.758	9.320
-% speed-up	-521	-175	18	49	47
PPPT					
-time	0,030	0.031	0.043	0.144	1.279
-% speed-up	-941	-647	-142	10	32

4.3 Numerical example: passport option pricing problem

In this section we will test all three methods on the Hamilton-Jacobi-Bellman equation for passport option pricing from [48]:

Example 2 (Passport option pricing problem)**.** *Passport options are contracts that allow the buyer to run trading account for a certain amount of time. After the maturity, the 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 [48], [47], the HJB equation for the current price of the contract can be simplified to the form*

$$\frac{\partial V}{\partial t} + \max_{|\theta|\leq 1}\Big(\frac{\sigma^2}{2}(x-\theta)^2\frac{\partial^2 V}{\partial x^2} + \Big((r-r_c-\gamma)\theta - (r-r_t-\gamma)x\Big)\frac{\partial V}{\partial x} - \gamma V\Big) = 0 \quad (4.11)$$

Here, t is time, $x = w/S$ and V is the option price divided by S. By r, we denote the 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 [48]: $r = 0.08, \gamma = 0.03, r_c = 0.12, r_t = 0.05, \sigma = 0.2$.

As discrete control set Θ, we took 41 equidistant points from the interval $[-1, 1]$.

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 nonuniform in space (nodes will be more dense near to zero)

Terminal and boundary conditions: According to [40], in the case of convex payoff (terminal 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 than our PPPT method. However, since we want to test also the PPPT method, we will use the non-convex terminal condition:

$$V_T(x) = \min\big(\max(0, x), \max(0, 0.8 + 0.2x)\big). \quad (4.12)$$

This terminal condition can be 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 obtained results similar to those in [48]. We use Dirichlet boundary conditions according to [48]:

$$V(x_{min}, t) = 0, \quad V(x_{max}, t) = 0.8 + 0.2x_{max}, \quad [x_{min}, x_{max}] = [-3, 4]. \quad (4.13)$$

Numerical results: We implemented three 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. The methods were implemented in Matlab and tested on an Intel Core i7-4770 CPU 3.40GHz computer with 8 GB RAM.

The results of these numerical simulations are presented in Table 4.3. As a 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 the PPPT method we computed 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

$$Err\ \ A^k = \|A^k - A^{ref}\|_2, \tag{4.14}$$

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. The experimental order of convergence is denoted as EOC and computed using the formula (4.10), where h_k is the step size on the k-th refinement level. The computational time is in Table 4.3 denoted simply as Time.

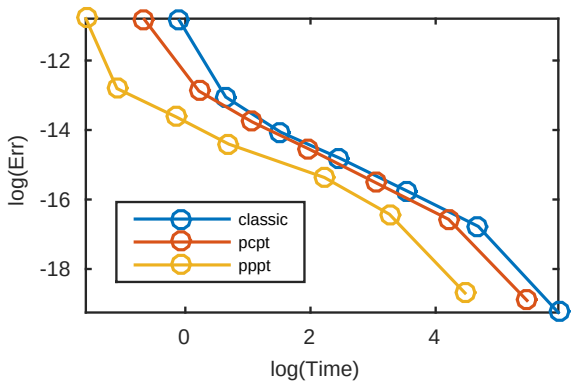

Figure 4.3: Comparison of natural logarithm of estimated absolute error of the approximation of solution against natural logarithm of computational time (in seconds) for classic implicit, PCPT and PPPT method with different grids.

Table 4.3: 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 4.3 illustrates the dependence between computational time and error for all three methods. We observe, that the PPPT method is always the most effective. For a fine discretization it is about 2 times faster than the PCPT method on the same level of

accuracy, and even faster than the classical 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 error estimation is negatively biased towards the classical implicit method. We note that the increase of EOC for the finest grids is also biased because we use a numerical approximation instead of an analytical solution as the reference solution.

5 Chapter 5
One-dimensional Tree-Grid method

In Chapter 3 we proved that we cannot obtain a monotone scheme with order of convergence higher than 2. Therefore, we tried to speed up the convergence (in means of computational time, not in means of convergence order) by reduction of control space in the piecewise predicted policy timestepping method in Chapter 4. In this chapter, we present another approach on how to reduce the computational time - we will move from the implicit methods to faster explicit methods. It is in fact easy to construct explicit versions of the implicit methods from Chapter 3, however, these methods are not unconditionally stable and monotone. Here we present a method that is unconditionally stable, monotone, consistent and explicit. This chapter is based on paper [28].

5.1 Recapitulation: problem formulation

For convenience we repeat here the problem formulation from Chapter 2. We are concerned with searching for the value function $V(s,t)$ of the following *general stochastic control problem* (SCP):

$$V(s,t) = \max_{\theta(s,t)\in\bar{\Theta}} \mathbb{E}\left(\int_t^T \exp\left(\int_t^k r(S_l,l,\theta(S_l,l))dl \right) f(S_k,k,\theta(S_k,k))dk \right.$$
$$\left. + \exp\left(\int_t^T r(S_k,k,\theta(S_k,k))dk \right) V_T(S_T) \Big| S_t = s \right), \quad (5.1)$$

$$dS_t = \mu(S_t,t,\theta(S_t,t))dt + \sigma(S_t,t,\theta(S_t,t))dW_t, \quad (5.2)$$
$$0 < t < T, \quad s \in \mathbb{R},$$

where s is state variable and t is time. Here, $\bar{\Theta}$ is space of all *suitable* control functions from $\mathbb{R}\times[0,T]$ to a set Θ. For our purpose, we will suppose Θ to be discrete. If this is not the case, we can easily achieve this property by its discretization. We also suppose that the functions r, f, μ, σ, V_T are chosen *suitably* (see Chapter 2). Now following Bellman's principle, the *dynamic programming equation* holds:

$$V(s,t_j) = \max_{\theta(s,t)\in\bar{\Theta}_{t_j}} \mathbb{E}\left(\int_{t_j}^{t_{j+1}} \exp\left(\int_{t_j}^k r(S_l,l,\theta(S_l,l))dl \right) f(S_k,k,\theta(S_k,k))dk \right.$$
$$\left. + \exp\left(\int_{t_j}^{t_{j+1}} r(S_k,k,\theta(S_k,k))dk \right) V(S_{t_{j+1}},t_{j+1}) \Big| S_{t_j} = s \right), \quad (5.3)$$

where $0 \leq t_j < t_{j+1} \leq T$ are some time-points and $\bar{\Theta}_{t_j}$ is a set of control functions from $\bar{\Theta}$ restricted to the $\mathbb{R} \times [t_j, t_{j+1})$ domain. Using this equation (5.3), it can be shown [38], that solving the SCP (5.1),(5.2) is equivalent to solving the Hamilton-Jacobi-Bellman (HJB) equation:

$$\frac{\partial V}{\partial t} + \max_{\theta \in \Theta} \left(\frac{\sigma(\cdot)^2}{2} \frac{\partial^2 V}{\partial s^2} + \mu(\cdot) \frac{\partial V}{\partial s} + r(\cdot) V + f(\cdot) \right) = 0, \tag{5.4}$$

$$V(s, T) = V_T(s), \tag{5.5}$$

$$0 < t < T, \quad s \in \mathbb{R}.$$

We repeat intentionally that the maximum operator in (5.1) and (5.4) can be replaced by a minimum operator and the whole following analysis will hold analogously.

5.2 Construction of the Tree-Grid method

5.2.1 The basic idea

In our proposed method we compute the approximation of the solution on a rectangular domain $[s_L, s_R] \times [0, T]$ with some grid as in PDE-based schemes. The gridpoints are denoted as $[s_i, t_j]$, $i \in \{1, 2, \ldots, N\}$, $j \in \{1, 2, \ldots, M\}$, $k < l \Rightarrow S_k < s_l, t_k < t_l$, $t_1 = 0, t_M = T$, $s_1 = s_L, s_N = s_R$. For the step-sizes we use the following notation: $\Delta_i s = s_{i+1} - s_i$, $\Delta_j t = t_{j+1} - t_j$. We point out that the grid defined in such manner is very general, in contrast to grids or lattices used for Markov chain approximations or tree methods. Later, we will show that in our new explicit method no additional restrictions on the grid are required, in contrast to standard explicit methods. This gives us a lot of freedom to choose the discretization not only according to the problem coefficients, but also according to the terminal condition (5.5), being an important advantage of implicit methods.

The numerical approximation of $V(s_i, t_j)$ will be denoted as $v(s_i, t_j)$ or simply as v_i^j. We define a terminal condition $v_i^M = V_T(s_i)$ and some suitable boundary conditions. In this paper we suppose that the solution on the intervals $[-\infty, s_1] \times [0, T]$ resp. $[s_N, \infty] \times [0, T]$ can by approximated with known functions $BC_L(s, t)$ resp. $BC_R(s, t)$. This also includes the case of Dirichlet boundary conditions, where BC_L and BC_R are constant in s. In case of Neumann boundary conditions, BC_L and BC_R can be set to linear functions with a prescribed slope, fulfilling $BC_L(s_2, t_j) = v(s_2, t_j)$, $BC_R(s_{N-1}, t_j) = v(s_{N-1}, t_j)$. Also generalization of the method to other cases of boundary conditions will be straightforward.

The main idea of this scheme follows the same principle as most numerical methods for this kind of problems: we will start in the last time layer $t_M = T$ and then subsequently compute values in the previous time layers t_j. To intuitively derive the method we will use the original problem formulation (5.1),(5.2) and the dynamic programming equation (5.3). To prove the convergence we will however regard the scheme as an approximation of the PDE-problem (5.4),(5.5).

Let us assume, we are at time t_j, $S_{t_j} = s_i$ and we want to compute an approximation of the current value of the value function v_i^j. Also assume that we already somehow know the values v_l^{j+1}, $\forall l = 1, 2, \ldots, N$ from the previous time layer t_{j+1}. Now we can compute

the approximate probability distribution of $S_{t_{j+1}}$ using the SDE (5.2) for the stochastic process:

$$\begin{aligned}\hat{S}_{t_{j+1}} &= S_{t_j} + \mu(S_{t_j}, t_j, \theta(S_{t_j}, t_j))\Delta_j t + \sigma(S_{t_j}, t_j, \theta(S_{t_j}, t_j))\Delta_j W \\ &= s_i + \mu(s_i, t_j, \theta)\Delta_j t + \sigma(s_i, t_j, \theta)\Delta_j W. \end{aligned} \tag{5.6}$$

Here, and also later if misunderstanding is not possible, we abbreviate $\theta(s_i, t_j)$ simply as θ and $\Delta_j W$ is a normally distributed random variable (RV) with mean 0 and variance $\Delta_j t$. Using this approximation, we want to compute v_i^j, the approximation of $V(s_i, t_j)$, by using again some discrete approximation of the dynamic programming equation (5.3). However, as we only know approximations of $V(s, t_{j+1})$ in discrete points s_i, a continuous RV $\hat{S}_{t_{j+1}}$ is not suitable and should be replaced by a discrete one. This is the main idea of our method as well as of the forward shooting grid (FSG) methods, Tree methods, and Markov chain approximation methods.

Problem: Discrete random variable (RV) with values from $\{s_1, s_2, \ldots, s_N\}$ suitably approximating normally distributed $\hat{S}_{t_{j+1}}$ from (5.6) should be found.

5.2.2 Excursion: FSG method

The FSG approach [5] to this problem is to approximate $\hat{S}_{t_{j+1}}$ with a RV $\tilde{S}_{t_{j+1}}$ that attains a finite number (typically 2, $\tilde{s}_+$ and $\tilde{s}_-$) of values with in-forward given probabilities (p_+ and p_-). These probabilities with corresponding values arise typically from a binomial tree model, therefore the first two moments of $\hat{S}_{t_{j+1}}$ and of $\tilde{S}_{t_{j+1}}$ are matching, what is a desirable property, as $\hat{S}_{t_{j+1}}$ is normally distributed and therefore fully characterized by its first two moments. However as the values $\tilde{s}_+$ and $\tilde{s}_-$ of $\tilde{S}_{t_{j+1}}$ typically do not coincide with the gridpoints from the set $\{s_1, s_2, \ldots, s_N\}$, approximations v_+^{j+1}, v_-^{j+1}, of $V(\tilde{s}_+, t_{j+1})$, $V(\tilde{s}_-, t_{j+1})$ are not known. Therefore, these values are computed by some interpolation formula from known values $v_1^{j+1}, v_2^{j+1}, \ldots, v_N^{j+1}$: $v_\pm^{j+1} = \sum_{k_\pm=1}^{K_\pm} \alpha_{k_\pm}^{\pm} v_{i_{k_\pm}}^{j+1}$. After that a discrete version of the dynamic programming equation (5.3) will be used, and at this step we will be interested only in an approximation of the expected value $\mathbb{E}\big(V(S_{t_{j+1}} t_{j+1})\big)$, that will be computed as

$$\begin{aligned}\mathbb{E}(V(S_{t_{j+1}}, t_{j+1})) &\approx \mathbb{E}(V(\hat{S}_{t_{j+1}}, t_{j+1})) \approx \mathbb{E}(V(\tilde{S}_{t_{j+1}}, t_{j+1})) \\ &\approx p_+ v_+^{j+1} + p_- v_-^{j+1} = \sum_{k_+=1}^{K_+} p_+ \alpha_{k_+}^{+} v_{i_{k_+}}^{j+1} + \sum_{k_-=1}^{K_-} p_- \alpha_{k_-}^{-} v_{i_{k_-}}^{j+1}, \end{aligned} \tag{5.7}$$

where the first approximation in (5.7) is with respect to time, second one is with respect to space and the last approximation is an interpolation with the known values in the grid points. However, the final approximation (5.7) can be again interpreted as an expected value $\mathbb{E}\big(V(\tilde{S}'_{t_{j+1}}, t_{j+1})\big)$ where $\tilde{S}'_{t_{j+1}}$ is a discrete RV taking values $s_{i_{k_\pm}}$ with "probabilities" $p_\pm \alpha_{k_\pm}^{\pm}$. However in contrast to $\tilde{S}_{t_{j+1}}$, the moments of $\tilde{S}'_{t_{j+1}}$ will most probably not match with the moments of $\hat{S}_{t_{j+1}}$. This can be interpreted in such manner, that using this approach we solve a SCP driven by an SDE different from (5.2). Moreover, depending on the interpolation formula the "probabilities" might not sum up to 1 and even not be non-negative anymore (not in the case of constant or linear interpolation), what may lead

to instability of the whole scheme. This possible defect of the method is analogous to the instability of explicit FDM schemes if the timestep-spacestep condition is not met: both defects harm the monotonicity of the schemes. This makes such methods unsuitable for searching for viscosity solutions and possibly even unstable.

We should note, that the above analysis was done for a FSG method based on $\tilde{S}_{t_{j+1}}$ attaining two values, but the case of more values is completely analogous. Of course interpolation used in this method may be in many cases "good enough", meaning that the moment matching is done in the limit case, and the whole method may converge to the solution. However this is not automatic, and it is problem-specific. An analysis of when FSG is successful and when not for the problem of path-dependent option pricing can be found in [18].

5.2.3 The basic Tree-Grid method

In our new scheme we will also approximate $\hat{S}_{t_{j+1}}$ from (5.6) with a discrete RV $\tilde{S}_{t_{j+1}}$. However, because of the problems with standard FSG schemes, we choose a different approach to construct this RV. In order to avoid interpolation $\tilde{S}_{t_{j+1}}$ will attain only values from the set $\{s_1, s_2, \ldots, s_N\}$. Exceptions arising close to a boundary will be discussed later. In this section, we will derive a scheme where $\tilde{S}_{t_{j+1}}$ attains three possible values $s_-, s_o, s_+ \in \{s_1, s_2, \ldots, s_N\}$, $s_- < s_o < s_+$, with corresponding probabilities p_-, p_o, p_+. Of course, because of (5.6), these values will depend on the current state s_i, time t_j and control θ and should be denoted as $s_-(s_i, t_j, \theta)$, $s_o(s_i, t_j, \theta)$, $s_+(s_i, t_j, \theta)$, $p_-(s_i, t_j, \theta)$, $p_o(s_i, t_j, \theta)$, $p_+(s_i, t_j, \theta)$, however for simplicity we prefer the shorter form. We will try to choose the values in such manner, that the following conditions are satisfied:

$$p_-, p_o, p_+ \geq 0, \tag{5.8}$$

$$p_- + p_o + p_+ = 1, \tag{5.9}$$

$$p_- s_- + p_o s_o + p_+ s_+ = E, \tag{5.10}$$

$$p_- s_-^2 + p_o s_o^2 + p_+ s_+^2 = Var + E^2, \tag{5.11}$$

where

$$E := \mathbb{E}(\hat{S}_{t_{j+1}}) = s_i + \mu(s_i, t_j, \theta)\Delta_j t, \tag{5.12}$$

$$Var := Var(\hat{S}_{t_{j+1}}) = \sigma(s_i, t_j, \theta)^2 \Delta_j t. \tag{5.13}$$

The first two conditions (5.8),(5.9) state that p_-, p_o, p_+ can be interpreted as probabilities and, as we will see later, they also ensure stability and monotonicity of the scheme. The following two conditions (5.10),(5.11) ensure that the first two moments of the RVs $\tilde{S}_{t_{j+1}}$ and $\hat{S}_{t_{j+1}}$ are matching, and as shown later, together with (5.9) also ensure the consistency of the scheme with the PDE (5.4). Solving equations (5.9)-(5.11) we get

$$p_- = \frac{(s_o - E)(s_+ - E) + Var}{(s_- - s_o)(s_- - s_+)}, \tag{5.14}$$

$$p_o = \frac{(s_- - E)(s_+ - E) + Var}{(s_o - s_-)(s_o - s_+)}, \tag{5.15}$$

$$p_+ = \frac{(s_- - E)(s_o - E) + Var}{(s_+ - s_-)(s_+ - s_o).} \tag{5.16}$$

The question remains, if we can choose s_-, s_o, s_+ in such manner, that the non-negativity condition (5.8) is also fulfilled.

Let us suppose without loss of generality that $\mu(s_i, t_j, \theta) \geq 0 \Rightarrow E \geq s_i$. If $E < s_-$ or $E > s_+$ then $p_o < 0$. If $Var > 0$, also $E = s_-$ or $E = s_+$ lead to $p_o < 0$. Therefore, we will choose s_-, s_+ so that $s_- < E < s_+$. As we will see later, for unconditional consistency of the scheme, it is necessary that one of s_-, s_o, or s_+ equals to s_i. As $s_i \leq E < s_+$ we can't choose $s_+ = s_i$. Analogously in case of a negative drift $\mu(s_i, t_j, \theta) \leq 0$, we would not be able to choose $s_- = s_i$. To make a suitable choice in both cases we will choose $s_o = s_i$. Now, (case of a positive drift), $p_+ \geq 0$ automatically (in case of a negative drift it would be $p_- \geq 0$). The denominator of p_- is positive and hence we want a positive numerator. The denominator of p_o is negative therefore we want a negative numerator.

Since s_- only appears in the numerator of p_o and not in the numerator of p_-, for any choice of s_+ we can choose s_- small enough such that $p_o \geq 0$ for any choice of s_+. Choices of s_- behind the boundary ($s_- < s_1$) are also possible as a special case and will be explained later. Therefore the only question remains, if we can choose such s_+, that the numerator of p_- will be positive. We will use abbreviations $\Delta_{i+}s = s_+ - s_i = s_+ - s_o$ and $\Delta_{-i}s = s_i - s_- = s_o - s_-$. Moreover, we will use abbreviations $\mu := \mu(s_i, t_j, \theta)$ and $\sigma = \sigma(s_i, t_j, \theta)$ if confusion is not possible. For the condition on numerator of p_- from equation (5.14) holds

$$\begin{aligned}(s_o - E)(s_+ - E) + Var \geq 0 &\Leftrightarrow -\mu\Delta_j t(\Delta_{i+}s - \mu\Delta_j t) + \sigma^2\Delta_j t \geq 0 \\ &\Leftrightarrow \Delta_{i+}s \leq \mu\Delta_j t + \sigma^2/\mu. \end{aligned} \tag{5.17}$$

It should also hold $s_+ > E \Rightarrow \Delta_{i+}s > \mu\Delta_j t$. Combining this with (5.17), we get the following condition: $\mu\Delta_j t < \Delta_{i+}s \leq \mu\Delta_j t + \sigma^2/\mu$. A sufficient condition, under which s_+ leading to fulfillment of this inequality can be found, is

$$\Delta s \leq \frac{\sigma(s_i, t_j, \theta)^2}{\mu(s_i, t_j, \theta)}, \tag{5.18}$$

where $\Delta s = \max_{i \in \{1,2,\ldots,N-1\}} \Delta_i s$. This seems a good result, however as we will see later, the convergence of this method depends on the distances $\Delta_{i+}s$, $\Delta_{-i}s$. Unfortunately, we do not have any bound on $\Delta_{-i}s$ now, we just know that it can be chosen to ensure that p_o will be positive. Therefore we will try another approach: we will try to minimize the distances $\Delta_{i+}s$, $\Delta_{-i}s$ while keeping p_o positive. Then we will check, if also p_- is positive.
Problem:

$$\min_{s_+, s_- \in \{s_1, s_2, \ldots, s_n\}} \min\left(s_+ - s_i, s_i - s_-\right) \tag{5.19}$$

$$|(s_- - E)(s_+ - E)| \geq Var. \tag{5.20}$$

Solving this problem may not be trivial in general, however an exact solution is also not needed. If we assume E to be "close enough" to s_i, then some "close to optimal" (and possibly also optimal) solution will be:

$$s_- = \left\lfloor E - \sqrt{Var} \right\rfloor_s = \left\lfloor s_i + \mu\Delta_j t - \sqrt{Var} \right\rfloor_s, \tag{5.21}$$

$$s_+ = \left\lceil E + \sqrt{Var} \right\rceil_s = \left\lceil s_i + \mu\Delta_j t + \sqrt{Var} \right\rceil_s, \tag{5.22}$$

where $\lceil\rceil_s$ denotes rounding to the nearest greater element from $s_1, s_2, \ldots, s_N$, and $\lfloor\rfloor_s$ denotes rounding to nearest smaller element from $s_1, s_2, \ldots, s_N$. If such element does not exist, $\lceil x\rceil_s$ and $\lfloor x\rfloor_s$ will return just x. This corresponds to the boundary cases where $x < s_1$ or $x > s_N$ and will be discussed later. As we will see later, on an equidistant or "locally equidistant" grid it may be advantageous to choose s_+ and s_- symmetric around s_i. Therefore we propose here also another choices of s_+, s_-, that will ensure this symmetry, fulfill condition (5.20), however possibly lead to a greater value of the minimized expression (5.19):

$$s_- = \left\lfloor s_i - \sqrt{(\mu\Delta_j t)^2 + Var}\right\rfloor_s, \tag{5.23}$$

$$s_+ = \left\lceil s_i + \sqrt{(\mu\Delta_j t)^2 + Var}\right\rceil_s. \tag{5.24}$$

Now, for (5.21),(5.22) the following estimates hold

$$s_- \geq s_i + \mu\Delta_j t - \sqrt{Var} - \Delta s, \tag{5.25}$$

$$s_+ \leq s_i + \mu\Delta_j t + \sqrt{Var} + \Delta s, \tag{5.26}$$

and for (5.23),(5.24) the estimates hold

$$s_- \geq s_i - \sqrt{(\mu\Delta_j t)^2 + Var} - \Delta s \geq s_i - |\mu|\Delta_j t - \sqrt{Var} - \Delta s, \tag{5.27}$$

$$s_+ \leq s_i + \sqrt{(\mu\Delta_j t)^2 + Var} + \Delta s \leq s_i + |\mu|\Delta_j t + \sqrt{Var} + \Delta s. \tag{5.28}$$

Let us now check if p_- is non-negative, that means, if its numerator is non-negative. Substituting (5.22) or (5.24) into it, and further supposing $\mu \geq 0$, we get

$$\begin{aligned}(s_o - E)(s_+ - E) + Var &= -\mu\Delta_j t(s_+ - s_o - \mu\Delta_j t) + Var \\ &\geq -\mu\Delta_j t\left(\sqrt{Var} + \Delta s\right) + Var = -\mu\Delta_j t\left(\sigma\sqrt{\Delta_j t} + \Delta s\right) + \sigma^2\Delta_j t.\end{aligned} \tag{5.29}$$

This is greater than 0 if $\Delta s \leq \sigma^2/\mu - \sigma\sqrt{\Delta_j t}$. A completely analogous analysis can be done for the case of negative drift $\mu < 0 \Rightarrow E < x_i$. Joining both cases into one condition, we get that if $s_o = s_i$ and (5.21), (5.22) or (5.23), (5.24) holds, then

$$\Delta s \leq \frac{\sigma(s_i, t_j, \theta)^2}{|\mu(s_i, t_j, \theta)|} - \sigma(s_i, t_j, \theta)\sqrt{\Delta_j t} \tag{5.30}$$

is a sufficient condition for the non-negativity of p_-,p_o,p_+ defined in (5.14)-(5.16). The last question is, if $s_- < s_o < s_+$ holds. This may be a problem in (5.21) (in case of a positive drift) or in (5.22) (in case of a negative drift). However it is easy to check that the condition (5.30) is sufficient for this inequality to be fulfilled. The condition (5.30) is quite weak for σ large enough. However for problems with vanishing σ it may be hard or even impossible to fulfill. In the next section we will describe how to tackle this problem.

5.2.4 The Tree-Grid method with artificial diffusion

Let us now examine the case that condition (5.30) is not fulfilled. This can only happen if the variance Var defined in (5.13) is not large enough to compensate the negative part

in (5.29). We solve this problem by redefining the variance Var in such manner that we add to the variance (5.13) some additional positive term of higher order in $\Delta_j t$:

$$Var := \sigma(s_i, t_j, \theta)^2 \Delta_j t + a(s_i, t_j, \theta)^2 (\Delta_j t)^2. \tag{5.31}$$

Here $a(s_i, t_j, \theta)^2(\Delta_j t)^2$ is the so-called *artificial diffusion term*, and if large enough, use of this new modified variance (5.31) should lead to positive weights. Moreover, as the whole term should be vanishing with $\Delta_j t \to 0$ and the true variance term $\sigma\Delta_t$ should dominate. For this, we need however $a(s_i, t_j, \theta)$ (later denoted simply as a) to be bounded. Now assuming a positive μ, we will repeat the analysis (5.29) of the numerator of p_- with the new Var (results for p_o and p_+ still hold).

$$\begin{aligned}(s_o - E)(s_+ - E) + Var &\geq -\mu\Delta_j t \left(\sqrt{Var} + \Delta s\right) + Var \qquad (5.32)\\ &= -\mu\Delta_j t \left(\sqrt{\sigma^2\Delta_j t + a^2(\Delta_j t)^2} + \Delta s\right) + \sigma^2\Delta_j t + a^2(\Delta_j t)^2 \\ &\geq -\mu\Delta_j t \left(\sigma\sqrt{\Delta_j t} + a\Delta_j t + \Delta s\right) + \sigma^2\Delta_j t + a^2(\Delta_j t)^2. \qquad (5.33)\end{aligned}$$

In the last step, we decided that a is positive, so that we can do the above estimation. Now in order to introduce as small artificial diffusion as needed, but still having (5.33) non-negative, we will choose a as the root of $-\mu\Delta_j t \left(\sigma\sqrt{\Delta_j t} + a\Delta_j t + \Delta s\right) + \sigma^2\Delta_j t + a^2(\Delta_j t)^2 = 0$. Moreover, to ensure the positivity of a, we will choose the larger root:

$$a = \frac{\mu\Delta_j t + \sqrt{\mu^2(\Delta_j t)^2 - 4|\mu|\Delta_j t \left(\sigma^2/|\mu| - \sigma\sqrt{\Delta_j t} - \Delta s\right)}}{2\Delta_j t} \tag{5.34}$$

We should note, that we substituted a non-negative μ with $|\mu|$ such that (5.34) holds also as result of fully analogous analysis for negative μ.

If $\sigma^2/\mu - \sigma\sqrt{\Delta_j t} - \Delta s \geq 0$, then the condition (5.30) is fulfilled and we can switch to $a = 0$. Let us now examine the case that $\sigma^2/\mu - \sigma\sqrt{\Delta_j t} - \Delta s < 0$; in that case, the whole discriminant and therefore also a is positive. That means, the numerator of p_- as well as p_- itself is positive and we found positive probabilities p_-, p_o, p_+. Now we will try to find an upper bound on a, satisfying convergence of new variance to the true variance from the problem setting. Following (5.34),

$$\begin{aligned}a &\leq \frac{\mu\Delta_j t + \sqrt{\mu^2(\Delta_j t)^2 - 4|\mu|\Delta_j t \min_{\sigma\in\mathbb{R}^+}\left(\sigma^2/|\mu| - \sigma\sqrt{\Delta_j t} - \Delta s\right)}}{2\Delta_j t} \\ &= \frac{\mu\Delta_j t + \sqrt{2\mu^2(\Delta_j t)^2 + 4|\mu|\Delta_j t\Delta s}}{2\Delta_j t} \\ &\leq \frac{\mu\Delta_j t + \sqrt{2\mu^2(\Delta_j t)^2} + \sqrt{4|\mu|\Delta_j t\Delta s}}{2\Delta_j t} \\ &= \frac{1}{\Delta_j t}\left(\frac{(1+\sqrt{2})|\mu|}{2}\Delta_j t + 2\sqrt{|\mu|}\sqrt{\Delta_j t\Delta s}\right). \qquad (5.35)\end{aligned}$$

Let us define the abbreviations:

$$m_1 = (1+\sqrt{2})|\mu|/2, \quad m_2 = 2\sqrt{|\mu|}. \tag{5.36}$$

Following the estimation, for the whole artificial diffusion term holds:

$$\begin{aligned} a^2(\Delta_j t)^2 &= (m_1\Delta_j t + m_2\sqrt{\Delta_j t \Delta s})^2 \\ &= m_1^2(\Delta_j t)^2 + 2m_1m_2\Delta_j t\sqrt{\Delta_j t\Delta s} + m_2^2\Delta_j t\Delta s \\ &\leq m_1^2(\Delta_j t)^2 + m_1m_2\Delta_j t(\Delta_j t + \Delta s) + m_2^2\Delta_j t\Delta s \\ &= m_1(m_1+m_2)(\Delta_j t)^2 + m_2(m_1+m_2)\Delta_j t\Delta s \\ &= \mathcal{O}\big(\Delta t(\Delta t + \Delta s)\big), \end{aligned} \tag{5.37}$$

where $\Delta t = \max_{j\in\{1,2,\dots,M-1\}} \Delta_j t$. Together with (5.31) we get finally the estimate

$$Var = \mathcal{O}(\Delta t). \tag{5.38}$$

We will use this estimation of the artificial diffusion term later to prove the consistency.

5.2.5 The final Tree-Grid method algorithm

In the following algorithm, we are interested in the values v_i^{j+1} corresponding to the states s_-, s_o and s_+. Therefore, we define the following function:

$$\begin{aligned} \text{if} \quad s \in \{s_1, s_2, \dots, s_N\} &: v^{j+1}(s) = v_k^{j+1} \text{ so that } s = s_k \\ \text{else if} \quad s < s_1 &: v^{j+1}(s) = BC_L(s, t_{j+1}) \\ \text{else if} \quad s > s_N &: v^{j+1}(s) = BC_R(s, t_{j+1}). \end{aligned} \tag{5.39}$$

Moreover, we define the short notation $v_-^{j+1} = v^{j+1}(s_-)$, $v_o^{j+1} = v^{j+1}(s_o)$, $v_+^{j+1} = v^{j+1}(s_+)$ and $f_i^j(\theta) = f(s_i, t_j, \theta(s_i, t_j))$, $r_i^j(\theta) = r(s_i, t_j, \theta(s_i, t_j))$. Now, assuming $S_{t_j} = s_i$ in order to discretize the equation (5.3) we use the following approximations:

- $\int_{t_j}^{t_{j+1}} \exp\left(\int_{t_j}^{k} r(S_l, l, \theta(S_l, l))dl\right) f(S_k, k, \theta(S_k, k))dk \approx f_i^j(\theta)\Delta_j t$,
- $\exp\left(\int_{t_j}^{t_{j+1}} r(S_k, k, \theta(S_k, k))dk\right) \approx 1 + r_i^j(\theta)\Delta_j t$,
- $\mathbb{E}\left(V(S_{t_{j+1}}, t_{j+1}) \middle| S_{t_j} = s_i\right) \approx p_- v_-^{j+1} + p_o v_o^{j+1} + p_+ v_+^{j+1}$.

Then the discretized version of the dynamic programming equation (5.3) for $i = 2, 3, \dots, N-1$ reads

$$v_i^j = \max_{\theta\in\Theta}\Big(f_i^j(\theta)\Delta_j t + (1 + r_i^j(\theta)\Delta_j t)\left(p_- v_-^{j+1} + p_o v_o^{j+1} + p_+ v_+^{j+1}\right)\Big). \tag{5.40}$$

or

$$v_i^j = \max_{\theta\in\Theta} w_i^j(\theta), \tag{5.41}$$

$$w_i^j(\theta) = f_i^j(\theta)\Delta_j t + (1 + r_i^j(\theta)\Delta_j t)\left(p_- v_-^{j+1} + p_o v_o^{j+1} + p_+ v_+^{j+1}\right). \tag{5.42}$$

We note that uniqueness of the maximum in (5.41) is not needed. For $i = 1$ and $i = N$ we employ the boundary conditions:

$$v_1^j = BC_L(s_1, t_j), \quad v_N^j = BC_R(s_N, t_j). \tag{5.43}$$

Finally we can summarize the whole algorithm of the Tree-Grid method for solving the SCP (5.1),(5.2) (and the HJB equation (5.4),(5.5)):

Algorithm 5 The Tree-Grid method

Set $v_i^M = V_T(s_i)$ for $i = 1, 2, \ldots, N$
for $j = M-1, M-2, \ldots, 1$ **do**
  Compute v_1^j, v_N^j according to (5.43)
  for $i = 2, 3, \ldots, N-1$ **do**
    for $\theta \in \Theta$ **do**
      Compute E according to (5.12)
      if Condition (5.30) holds **then**
        Compute Var according to (5.13)
      else
        Compute a according to (5.34)
        Compute Var according to (5.31)
      end if
      Set $s_o = s_i$ and compute s_-, s_+ using (5.21)-(5.22) or (5.23)-(5.24)
      Compute p_-, p_o, p_+ using (5.14)-(5.16)
      Compute $w_i^j(\theta)$ using (5.39) and (5.42)
    end for
    Compute v_i^j according to (5.41)
  end for
end for

5.2.6 Relationship to other numerical methods

In this section we outline (very informally) the interesting relationships between our new method and standard approaches, as well as point out the most relevant differences.

Forward shooting grid methods. We already discussed FSG methods in Section 5.2.2 in order to motivate the Tree-Grid approach. We can see the Tree-Grid method in its simplest version (no artificial diffusion) as modification of the FSG method with 3 "branches" instead of 2, non-constant probabilities, and, most importantly with no need to perform any interpolation. This differences however make the method convergent in contrast to the (general) FSG.

Tree methods. The Tree-Grid method inherits many similarities with binomial and especially trinomial tree methods mostly used in option pricing. Both approaches are based on approximating the continuum of possible outcomes after 1 time-step by a RV gaining only 3 (in case of Tree-Grid and trinomial tree methods) values, graphically often represented by three new "branches" of the "tree". However, in contrast to the trinomial tree method where the branches are growing only from nodes on the tree, in the Tree-Grid method, 3 branches are "planted" in each gridpoint of an arbitrary grid. Therefore, we also chose the name "Tree-Grid" method. In trinomial tree methods, we get the value of

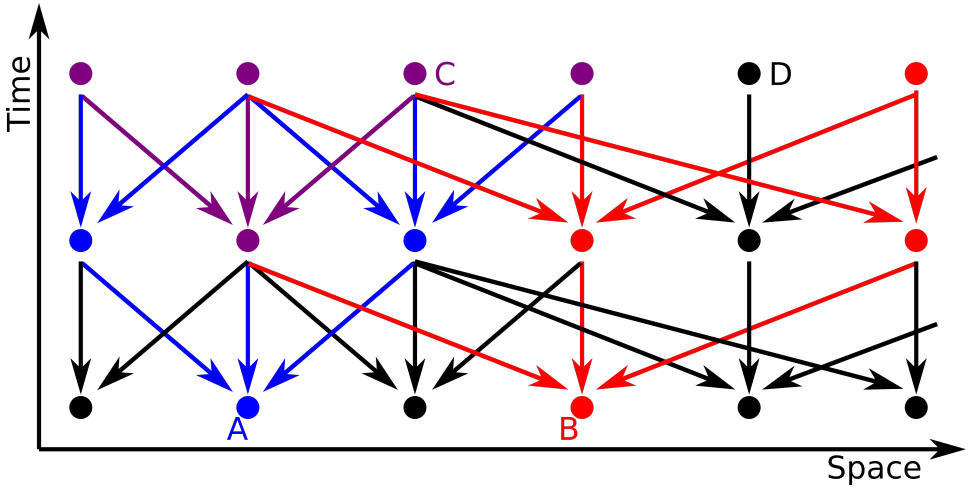

Figure 5.1: Illustration of the flexibility of the Tree-Grid structure. The node A represents a node with lower volatility, whereas node B exhibits higher volatility. Blue nodes are passing its values to node A and red nodes are passing its values to node B. Purple nodes have impact on both A and B. Node C passes its value to four different nodes in the previous time level whereas node D passes its value only to one node in the previous time level.

the value function (e.g. option price) only in 1 space point in the first time layer, whereas in the Tree-Grid method, we get the value of value function on a whole set of space points. One may correctly comment, that also trinomial tree method works on some lattice that can be easily extended so that more values are computed in the first time-layer. This is true, however this lattice is constructed depending on the problem, space-steps are already determined by timestep (can't be chosen according to terminal condition) and the time-step size also can't be determined for each layer arbitrarily. On the other hand the Tree-Grid method works on an arbitrary grid. Besides artificial diffusion, one of the most obvious technical differences is the following: in trinomial tree methods from each node grow 3 branches, and each ("inner") node also passes its value into 3 earlier nodes. However in tree grid method only the first statement holds: each node may pass it's value to different number of earlier nodes –depending on the problem and on the grid. The "tree structure" in Tree-Grid method is then much more flexible. This flexibility is illustrated in Figure 5.1.

Finite difference methods. FDMs are used to solve the HJB equation (5.4),(5.5) instead of the original SCP. However, in the next section, we will also prove the convergence of the approximation computed by the Tree-Grid method to the solution of the HJB equation, despite the fact that the derivation of the method was based on the original formulation (5.1),(5.2). This motivates us to try to look at the method through a finite difference perspective. As $s_o = s_i$ (and therefore $v_o^{j+1} = v_i^{j+1}$) it can be shown:

$$p_- v_-^{j+1} + p_o v_o^{j+1} + p_+ v_+^{j+1} = v_o^{j+1} + \mu \Delta_j t D_1 v_i^{j+1} + 1/2 \left(Var + (\mu \Delta_j t)^2 \right) D_2 v_i^{j+1}, \quad (5.44)$$

where D_1 and D_2 denote standard finite difference approximations of first and second

derivative on nonuniform grids:

$$D_1 v_i^{j+1} = \left(\frac{s_+ - s_i}{s_+ - s_-}\right) \frac{v_i^{j+1} - v_-^{j+1}}{s_i - s_-} + \left(\frac{s_i - s_-}{s_+ - s_-}\right) \frac{v_+^{j+1} - v_i^{j+1}}{s_+ - s_-}, \tag{5.45}$$

$$D_2 v_i^{j+1} = \left(\frac{v_+^{j+1} - v_i^{j+1}}{s_+ - s_i} - \frac{v_i^{j+1} - v_-^{j+1}}{s_i - s_-}\right) \Big/ \left(\frac{s_+ - s_-}{2}\right). \tag{5.46}$$

Now if we substitute (5.44) into (5.40) and suppose that no artificial diffusion is used, and for the discount rate holds $r_i^j(\theta) = 0$, the only difference between the Tree-Grid scheme and an explicit finite difference approximation with a *wide stencil* on the nodes s_-, s_i, s_+ is the term $1/2(\mu\Delta_j t)^2 D_2 v_i^{j+1}$. This term can be interpreted as some *inherent artificial diffusion* that comes into the scheme directly from numerical modeling, and is also making the scheme more stable than explicit FDMs (it in fact makes the derivation of condition (5.30) possible). Therefore, even in this simplest case this scheme can't be viewed as just FDM on specifically chosen nodes, although the similarity is clear. Let us note that such *inherent artificial diffusion term* is also present in tree methods (therefore they are not equivalent to FDMs as sometimes stated in literature, only equivalent up to certain order), but not present in Markov chain approximation methods from [31].

Markov chain approximation methods. The basic idea of Markov chain approximation methods is to construct a Markov chain (in discrete time) approximating a Markov process (5.2) (in continuous time) and then using this chain to find an approximation of the solution to the SCP -an idea very similar to Tree methods, however in literature used in more general frameworks as tree-methods are used mostly only in option pricing. From this viewpoint Tree-Grid method can be also seen as Markov chain approximation method as by constructing variable $\tilde{S}_{t_{j+1}}$ gaining values s_-, s_o, s_+ with probabilities p_-, p_o, p_+. In Section 5.2.3 we in fact constructed a Markov chain approximating (5.2). However, in standard Markov chain methods [31], (as well as in tree methods), the grid can't be chosen arbitrarily, is problem-dependent and the space step is determined by the time step. Moreover, these methods can be often linked to explicit FDMs and (as stated earlier) also do not possess the inherent artificial diffusion term.

5.3 Convergence of the Tree-Grid method

In the previous section we directly discretized the SDE (5.2) and the dynamic programming equation (5.3) to find the approximation of the solution of the SCP. However, in order to show the convergence of this approximation to the viscosity solution as the stepsizes tend to zero we will look at the above algorithm as on a method for solving the HJB equation (5.4),(5.5). Figure 5.2 illustrates these relationships between Tree-Grid method and stochastic control problem, dynamic programming equation and HJB equation.

At first, we will present the required definitions and the convergence theorem for general nonlinear problems from [4].

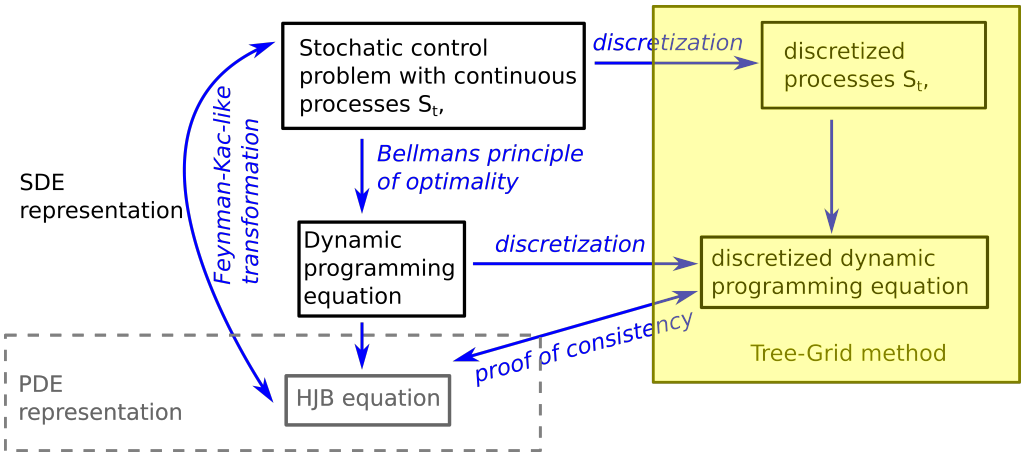

Figure 5.2: The Tree-Grid method is derived from the SCP where the dynamics is represented by stochastic differential equations (SDEs). The core of the method is the discretized stochastic process S_t plugged into the discretized dynamic programming equation, that follows from the SCP by application of Bellmans principle of optimality. Using the dynamic programming equation, the HJB PDE can be derived (via Feynman-Kac-like transformation), which solution is equivalent to the solution of the original SCP. We employ this fact by proving the convergence of the approximation computed using the Tree-Grid method to the solution of this HJB equation.

5.3.1 Consistency of the scheme

For the purposes of following analysis we rewrite (5.40) it in the form $Gv(s_i, t_j) = 0$:

$$\begin{aligned} Gv(s_i,t_j) =& G(v(s_i,t_j), v(s_-,t_{j+1}), v(s_o,t_{j+1}), v(s_+,t_{j+1})) \\ =& \frac{1}{\Delta_j t}\Big(v_i^j - \max_{\theta\in\Theta}\Big(f_i^j(\theta)\Delta_j t + (1 + r_i^j(\theta)\Delta_j t) \\ & \cdot \left(p_- v_-^{j+1} + p_o v_o^{j+1} + p_+ v_+^{j+1}\right)\Big)\Big) = 0. \end{aligned} \tag{5.47}$$

Using the theory from Section 2.2, our goal is to show that equation (5.47) is a monotone, consistent, and stable approximation of the nonlinear differential operator F defined by the PDE (5.4):

$$FV(s,t) = -\frac{\partial V}{\partial t} - \max_{\theta\in\Theta}\left(\frac{\sigma(\cdot)^2}{2}\frac{\partial^2 V}{\partial s^2} + \mu()\frac{\partial V}{\partial s} + r(\cdot)V + f(\cdot)\right). \tag{5.48}$$

Let us note that we multiplied both sides of (5.4) with -1 so that the operator F is elliptic as in the theory of Barles and Souganidis [4]. The variable x from Section 2.2 is here represented by a 2-dimensional vector $[s,t]$. Let us recall $\Delta s = \max_{i\in\{1,2,\dots,N-1\}} \Delta_i s$ and $\Delta t = \max_{j\in\{1,2,\dots,M-1\}} \Delta_j t$. Then the stepsize parameter h from the Section 2.2 is in our case defined as $h = \min(\Delta s, \Delta t)$.

At first we prove the consistency of the scheme (5.47) with the HJB equation (5.4). We define $\Delta_{i-}s = s_- - s_i = -\Delta_{-i}s$, $\Delta_{io}s = s_o - s_i$, $\Delta_{i+}s = s_+ - s_i$ and rewrite (5.14)-(5.16)

equivalently as

$$p_- = \frac{(\Delta_{io}s - \mu\Delta_j t)(\Delta_{i+}s - \mu\Delta_j t) + Var}{(\Delta_{i-}s - \Delta_{io}s)(\Delta_{i-}s - \Delta_{i+}s)}, \tag{5.49}$$

$$p_o = \frac{(\Delta_{i-}s - \mu\Delta_j t)(\Delta_{i+}s - \mu\Delta_j t) + Var}{(\Delta_{io}s - \Delta_{i-}s)(\Delta_{io}s - \Delta_{i+}s)}, \tag{5.50}$$

$$p_+ = \frac{(\Delta_{i-}s - \mu\Delta_j t)(\Delta_{io}s - \mu\Delta_j t) + Var}{(\Delta_{i+}s - \Delta_{i-}s)(\Delta_{i+}s - \Delta_{io}s)}. \tag{5.51}$$

Now we can see that p_-, p_o, p_+ is also a solution of the system of equations

$$p_- + p_o + p_+ = 1, \tag{5.52}$$

$$p_-\Delta_{i-}s + p_o\Delta_{io}s + p_+\Delta_{i+}s = \mu\Delta_j t, \tag{5.53}$$

$$p_-(\Delta_{i-}s)^2 + p_o(\Delta_{io}s)^2 + p_+(\Delta_{i+}s)^2 = Var + (\mu\Delta_j t)^2. \tag{5.54}$$

Now a lemma about a remainder-terms that will be used in our consistency proof follows.

Lemma 4 (Rest terms)**.** *Let $\Delta_{io}s = 0$ ($s_o = s_i$). Then for s_-, s_+ computed according to* (5.21),(5.22) *or* (5.23),(5.24) *we have*

$$R_3 := p_-(\Delta_{i-}s)^3 + p_o(\Delta_{io}s)^3 + p_+(\Delta_{i+}s)^3 = \mathcal{O}\left(\Delta t\left(\Delta t + \Delta s\right)\right), \tag{5.55}$$

and for s_-, s_+ computed according to (5.23),(5.24) *on equidistant grid, we have*

$$R_3 = \mathcal{O}\left(\Delta t\left(\Delta t + (\Delta s)^2\right)\right). \tag{5.56}$$

Moreover, for s_-, s_+ computed either by formulas (5.21),(5.22)*, or by* (5.23),(5.24) *it holds*

$$\begin{aligned} R_4^b &:= b_-p_-(\Delta_{i-}s)^4 + b_op_o(\Delta_{io}s)^4 + b_+p_+(\Delta_{i+}s)^4 \\ &= \mathcal{O}\left(\Delta t\left(\Delta t + (\Delta s)^2\right)\right), \end{aligned} \tag{5.57}$$

$$R_2^b := b_-p_-(\Delta_{i-}s)^2 + b_op_o(\Delta_{io}s)^2 + b_+p_+(\Delta_{i+}s)^2 = \mathcal{O}\left(\Delta t\right), \tag{5.58}$$

where $b_-, b_o, b_+ \in \mathbb{R}$ are arbitrary constants.

Proof. From (5.49)-(5.51) follows:

$$\begin{aligned} R_3 = \Delta_{i-}s\Delta_{io}s\Delta_{i+}s - \mu\Delta_j t\Delta_{io}s\Delta_{i+}s - \mu\Delta_j t\Delta_{i-}s\Delta_{i+}s - \mu\Delta_j t\Delta_{i-}s\Delta_{io}s \\ + Var\Delta_{i+}s + Var\Delta_{io}s + Var\Delta_{i-}s \\ + \mu^2(\Delta_j t)^2\Delta_{i+}s + \mu^2(\Delta_j t)^2\Delta_{io}s + \mu^2(\Delta_j t)^2\Delta_{i-}s. \end{aligned} \tag{5.59}$$

According to (5.38) $Var = \mathcal{O}(\Delta t)$. Substituting this into (5.21),(5.22) or (5.23),(5.24) using the inequality $s_i < s_o = s_i < s_+$ and using definitions of $\Delta_{i-}s, \Delta_{i+}s$, we get

$$\Delta_{i+}s = \mathcal{O}(\sqrt{\Delta t} + \Delta s), \quad \Delta_{i-}s = \mathcal{O}(\sqrt{\Delta t} + \Delta s), \quad \Delta_{i-}s + \Delta_{i+}s = \mathcal{O}(\Delta s). \tag{5.60}$$

As $\Delta_{io}s = 0$ for $s_o = s_i$, by using formulas (5.21),(5.22) we get

$$\begin{aligned} R_3 = &- \mu\Delta_j t\Delta_{i-}s\Delta_{i+}s + Var\Delta_{i+}s + Var\Delta_{i-}s \\ &+ \mu^2(\Delta_j t)^2\Delta_{i+}s + \mu^2(\Delta_j t)^2\Delta_{i-}s = \mathcal{O}\big(\Delta t\,(\Delta t + \Delta s)\big), \end{aligned} \tag{5.61}$$

where we used (5.38), (5.60). By using formulas (5.23),(5.24) on an equidistant grid, it holds $\Delta_{i-}s = -\Delta_{i+}s$ and therefore

$$R_3 = - \mu\Delta_j t\Delta_{i-}s\Delta_{i+}s = \mu\Delta_j t(\Delta_{i+}s)^2 = \mathcal{O}\left(\Delta t\left(\Delta t + (\Delta s)^2\right)\right). \tag{5.62}$$

Let us define $R_4 := p_-(\Delta_{i-}s)^4 + p_o(\Delta_{io}s)^4 + p_+(\Delta_{i+}s)^4$. Now by using either (5.21),(5.22) or (5.23),(5.24) we can prove that $R_4 = \mathcal{O}\left(\Delta t\left(\Delta t + (\Delta s)^2\right)\right)$ in the same manner as in the case of R_3. Now it holds

$$\begin{aligned} \mathcal{O}\big(\Delta t\left(\Delta t + (\Delta s)^2\right)\big) &= \min(b_-, b_o, b_+)R_4 \le R_4^b \\ &\le \max(b_-, b_o, b_+)R_4 = \mathcal{O}\left(\Delta t\left(\Delta t + (\Delta s)^2\right)\right), \end{aligned}$$

and therefore also $R_4^b = \mathcal{O}\left(\Delta t\left(\Delta t + (\Delta s)^2\right)\right)$. Finally, following (5.54), (5.38) it holds

$$\begin{aligned} \mathcal{O}\left(\Delta t\right) &= \min(b_-, b_o, b_+)\left(Var + (\mu\Delta_j t)^2\right) \le R_2^b \\ &\le \max(b_-, b_o, b_+)\left(Var + (\mu\Delta_j t)^2\right) = \mathcal{O}\left(\Delta t\right), \end{aligned}$$

and therefore $R_2^b = \mathcal{O}\left(\Delta t\right)$. □

The lemma establishing the consistency of our scheme follows.

Lemma 5 (Consistency)**.** *If the parameters p_-, p_o, p_+ in the scheme* (5.47) *satisfy the conditions* (5.9)-(5.11)*, $s_o = s_i$ and s_-, s_+ are computed according to* (5.21),(5.22) *then the scheme* (5.47) *is consistent with the PDE* (5.48)*.*

Proof. Let $\phi : \mathbb{R} \times [0,T] \to \mathbb{R}$ be a C^∞-smooth function. Let us define $\phi_i^j = \phi(s_i, t_j)$, $\phi_-^j = \phi(s_-, t_j)$, $\phi_o^j = \phi(s_o, t_j)$, $\phi_+^j = \phi(s_+, t_j)$. Now it holds

$$\begin{aligned} \phi_*^{j+1} = \phi_i^j + \frac{\partial\phi_i^j}{\partial s}\Delta_{i*}s + \frac{\partial\phi_i^j}{\partial t}\Delta_j t + \frac{1}{2}\frac{\partial^2\phi_i^j}{\partial s^2}(\Delta_{i*}s)^2 + \frac{\partial^2\phi_i^j}{\partial s\partial t}\Delta_{i*}s\Delta_j t \\ + \frac{1}{6}\frac{\partial^3\phi_i^j}{\partial s^3}(\Delta_{i*}s)^3 + \frac{1}{2}\frac{\partial^3\phi_i^j}{\partial s^2\partial t}(\Delta_{i*}s)^2\Delta_j t \\ + \frac{1}{24}\frac{\partial^4\phi_{i+\delta*}^{j+\epsilon*}}{\partial s^4}(\Delta_{i*}s)^4 + \frac{1}{6}\frac{\partial^4\phi_{i+\delta*}^{j+\epsilon*}}{\partial s^3\partial t}(\Delta_{i*}s)^3\Delta_j t + \mathcal{O}\left((\Delta_j t)^2\right), \end{aligned} \tag{5.63}$$

where the index $*$ should be substituted by either $-$ or o or $+$, and $\phi_{i+\delta*}^{j+\epsilon*} = \phi(s_i + \delta_*\Delta_{i*}s, t_j + \epsilon_*\Delta_j t)$, and $\delta_*, \epsilon_* \in [0,1]$. Now, using (5.52)-(5.54) and the definitions from

Lemma 4 we get:

$$\begin{aligned} p_-\phi_-^{j+1} + p_o\phi_o^{j+1} + p_+\phi_+^{j+1} &= \phi_i^j + \frac{\partial\phi_i^j}{\partial s}\mu\Delta_j t + \frac{\partial\phi_i^j}{\partial t}\Delta_j t \\ &+\frac{1}{2}\frac{\partial^2\phi_i^j}{\partial s^2}\left(Var + (\mu\Delta_j t)^2\right) + \frac{\partial^2\phi_i^j}{\partial s\partial t}\mu(\Delta_j t)^2 + \frac{1}{6}\frac{\partial^3\phi_i^j}{\partial s^3}R_3 \\ &+\frac{1}{2}\frac{\partial^3\phi_i^j}{\partial s^2\partial t}\left(Var + (\mu\Delta_j t)^2\right)\Delta_j t + R_4^b + R_2^{b'}\Delta_j t + \mathcal{O}\left((\Delta_j t)^2\right), \end{aligned} \tag{5.64}$$

where

$$b_* = \frac{1}{24}\frac{\partial^4\phi_{i+\delta*}^{j+\epsilon*}}{\partial s^4}, \quad b'_* = \frac{1}{6}\frac{\partial^4\phi_{i+\delta*}^{j+\epsilon*}}{\partial s^3\partial t}\Delta_{i*}s, \quad \text{for } * = -, o, +. \tag{5.65}$$

Now, using Lemma 4, and the definition of Var (5.31) we can rewrite (5.64) as

$$\begin{aligned} p_-\phi_-^{j+1} + p_o\phi_o^{j+1} + p_+\phi_+^{j+1} &= \phi_i^j + \frac{\partial\phi_i^j}{\partial s}\mu\Delta_j t + \frac{\partial\phi_i^j}{\partial t}\Delta_j t \\ &+ \frac{1}{2}\frac{\partial^2\phi_i^j}{\partial s^2}\left(\sigma^2\Delta_j t + (a^2+\mu^2)(\Delta_j t)^2\right) + \frac{\partial^2\phi_i^j}{\partial s\partial t}\mu(\Delta_j t)^2 + \frac{1}{6}\frac{\partial^3\phi_i^j}{\partial s^3}R_3 \\ &+ \frac{1}{2}\frac{\partial^3\phi_i^j}{\partial s^2\partial t}\left(\sigma^2\Delta_j t + (a^2+\mu^2)(\Delta_j t)^2\right)\Delta_j t + \mathcal{O}\left(\Delta t\left(\Delta t + (\Delta s)^2\right)\right) \\ &= \phi_i^j + \frac{\partial\phi_i^j}{\partial s}\mu\Delta_j t + \frac{\partial\phi_i^j}{\partial t}\Delta_j t + \frac{1}{2}\frac{\partial^2\phi_i^j}{\partial s^2}\sigma^2\Delta_j t \\ &\quad + \frac{1}{6}\frac{\partial^3\phi_i^j}{\partial s^3}R_3 + R_a + \mathcal{O}\left(\Delta t\left(\Delta t + (\Delta s)^2\right)\right), \end{aligned} \tag{5.66}$$

with the remainder term

$$R_a = \frac{1}{2}\frac{\partial^2\phi_i^j}{\partial s^2}a^2(\Delta_j t)^2 + \frac{1}{2}\frac{\partial^3\phi_i^j}{\partial s^2\partial t}a^2(\Delta_j t)^2, \tag{5.67}$$

and according to (5.37) $R_a = \mathcal{O}\big(\Delta t\,(\Delta t + \Delta s)\big)$ if $a > 0$ and $R_a = 0$ if $a = 0$ (artificial diffusion not needed). Now, substituting (5.66) to $G\phi(s_i, t_j)$ (defined according to (5.47)) we get

$$\begin{aligned} G\phi(s_i,t_j) &= \frac{1}{\Delta_j t}\Big(\phi_i^j - \max_{\theta\in\Theta}\Big(f_i^j(\theta)\Delta_j t + (1 + r_i^j(\theta)\Delta_j t)\Big(\phi_i^j + \frac{\partial\phi_i^j}{\partial s}\mu\Delta_j t \\ &+ \frac{\partial\phi_i^j}{\partial t}\Delta_j t + \frac{1}{2}\frac{\partial^2\phi_i^j}{\partial s^2}\sigma^2\Delta_j t + \frac{1}{6}\frac{\partial^3\phi_i^j}{\partial s^3}R_3 + R_a + \mathcal{O}\left(\Delta t\left(\Delta t + (\Delta s)^2\right)\right)\Big)\Big)\Big) \\ &= -\frac{\partial\phi_i^j}{\partial t} - \max_{\theta\in\Theta}\Big(f_i^j(\theta) + r_i^j(\theta)\phi_i^j + \frac{\partial\phi_i^j}{\partial s}\mu + \frac{1}{2}\frac{\partial^2\phi_i^j}{\partial s^2}\sigma^2 \\ &\quad + \frac{1}{6}\frac{\partial^3\phi_i^j}{\partial s^3}\left(r_i^j(\theta)R_3 + \frac{R_3}{\Delta_j t}\right) + R_a + \mathcal{O}\left(\Delta t + (\Delta s)^2\right)\Big) \\ &= -\frac{\partial\phi_i^j}{\partial t} - \max_{\theta\in\Theta}\left(\frac{\sigma^2}{2}\frac{\partial^2\phi_i^j}{\partial s^2} + \mu\frac{\partial\phi_i^j}{\partial s} + r_i^j(\theta)\phi_i^j + f_i^j(\theta)\right) + R, \end{aligned} \tag{5.68}$$

where, according to the estimation of R_3 in Lemma 4 and according to estimation of R_a, $R = \mathcal{O}\left(\Delta t + (\Delta s)^2\right)$ if artificial diffusion is not present ($a = 0$), formulas (5.23), (5.24) were used and the space-grid is equidistant.

According to (5.48), we have

$$F\phi(s_i,t_j) = -\frac{\partial \phi_i^j}{\partial t} - \max_{\theta\in\Theta}\left(\frac{\sigma^2}{2}\frac{\partial^2\phi_i^j}{\partial s^2} + \mu\frac{\partial\phi_i^j}{\partial s} + r_i^j(\theta)\phi_i^j + f_i^j(\theta)\right). \tag{5.69}$$

Comparing (5.68) and (5.69) we see that $|F\phi(s_i,t_j) - G\phi(s_i,t_j)|$ is of order $\mathcal{O}(\Delta t + \Delta s)$ resp. $\mathcal{O}\left(\Delta t + (\Delta s)^2\right)$ and therefore vanishing with $h = \min(\Delta s, \Delta t) \to 0$. □

Remark 7 (Order of consistency)**.** *The original paper of Barles and Souganidis [4] does not define the order of convergence to the viscosity solution, it just presents a theory for convergence. Therefore, the impact of the order of the scheme (or "consistency order") on the convergence rate is not clear. Moreover, as shown in [26] the maximal order of a monotone scheme is 2. However, in the work of Wang and Forsyth [48] it is experimentally shown that a higher order of the scheme leads to faster convergence for a particular problem. Therefore following the proof of the previous Lemma on an equidistant space-grid it may be advantageous to use formulas* (5.23)*,* (5.24) *leading to an order of the scheme* $\mathcal{O}\left(\Delta t + (\Delta s)^2\right)$ *(if artificial diffusion is not needed) rather than formulas* (5.21)*,* (5.22) *leading to an order of the scheme* $\mathcal{O}(\Delta t + \Delta s)$*. On the other hand, formulas* (5.21)*,* (5.22) *may lead to smaller space-steps* Δ_{i-}, Δ_{i+} *which may theoretically also lead to a higher convergence rate. Therefore, the optimal choice between formulas* (5.21)*,* (5.22) *and* (5.23)*,* (5.24) *needs deeper examination.*

5.3.2 Monotonicity, stability, convergence

Next, we will prove the monotonicity and stability of the method (5.47). Together with the already proven consistency we get the convergence result for the Tree-Grid method. The lemma establishing the monotonicity property of our method follows.

Lemma 6 (Monotonicity)**.** *If the parameters* p_-*,* p_o*,* p_+ *in the scheme* (5.47) *satisfy the condition* (5.8) *and if* $1 + r_i^j(\theta)\Delta_j t \geq 0$ *for all* $\theta \in \Theta$ *then this scheme is monotone.*

Proof. In our case, monotonicity means that (5.47) is non-increasing in v_-^{j+1}, v_o^{j+1}, v_+^{j+1}. This follows directly from the non-negativity of the p_-, p_o, p_+ -condition (5.8). □

Remark 8. *Even if* $1 + r_i^j(\theta)\Delta_j t < 0$ *for some* θ*, we can get a monotone scheme if we substitute* $1 + r_i^j(\theta)\Delta_j t$ *by* $1/(1 - r_i^j(\theta)\Delta_j t)$ *in* (5.47) *for these parameters* θ*. Note that this change does not harm the consistency, nor the stability of the scheme.*

The next step is to prove the stability of the method. At first we will pose some conditions on the problem that will be needed for the stability proof.

Property 1 (Stability condition on the problem)**.** *We assume that:*

1. *There exist constants* C_f, C_r*,* $\forall s,t,\theta : |f(s,t,\theta)| < C_f$*,* $|r(s,t,\theta)| < C_r$*.*
2. *There exist a constant* C_L*,* $|BC_L(s,t)| < C_L$*,* $\forall t$ *and for all possible values* $s < s_1$ *of the variables* s_-, s_o, s_+ *for any grid.*
3. *There exist a constant* C_R*,* $|BC_R(s,t)| < C_R$*,* $\forall t$ *and for all possible values* $s > s_N$ *of the variables* s_-, s_o, s_+ *for any grid.*

The first condition simply establish boundedness of functions $r(\cdot), f(\cdot)$, and the second and third condition establish boundedness of the values that can flow into the model from behind the boundary. Checking if this condition is fulfilled is in most cases trivial. Now we state an inequality that will be used for the stability proof.

Lemma 7 (Inequality recurrence)**.**

$$x_j \leq a_j x_{j+1} + b_j, j < M \Rightarrow x_j \leq \left(\prod_{k=j}^{M-1} a_k\right) x_M + \sum_{k=j}^{M-1} \left(\left(\prod_{l=j}^{k-1} a_l\right) b_k\right).$$

Proof. Proof can be done easily by induction, therefore we omit here the details. □

Lemma about the stability of the method follows.

Lemma 8 (Stability)**.** *If the parameters p_-, p_o, p_+ in the scheme* (5.47) *satisfy the conditions* (5.8),(5.9)*, and the problem satisfies Property 1 then this scheme is stable.*

Proof. Let us define

$$C_j = \max\left(C_L, C_R, \max_{i\in\{1,2,\ldots,N\}}\left(v_i^j\right)\right).$$

Then, it holds:

$$\begin{aligned}
C_j =& \max\Big(C_L, C_R, \max_{i\in\{1,2,\ldots,N\}}\Big(f_i^j(\theta)\Delta_j t + (1 + r_i^j(\theta)\Delta_j t) \\
&\cdot \left(p_- v_-^{j+1} + p_o v_o^{j+1} + p_+ v_+^{j+1}\right)\Big)\Big) \\
\leq& \max\Big(C_L, C_R, \max_{i\in\{1,2,\ldots,N\}}\Big(C_f\Delta_j t + (1 + C_r\Delta_j t) \\
&\cdot \max\left(C_L, C_R, \max_{i\in\{1,2,\ldots,N\}}\left(v_i^{j+1}\right)\right)\Big)\Big) \\
=& C_f\Delta_j t + (1 + C_r\Delta_j t)C_{j+1} \\
\leq& C_f\Delta_j t + \exp(C_r\Delta_j t)C_{j+1}.
\end{aligned}$$

Using Lemma 7 we obtain

$$\begin{aligned}
C_j &= \left(\prod_{k=j}^{M-1} \exp(C_r\Delta_k t)\right) C_M + \sum_{k=j}^{M-1}\left(\left(\prod_{l=j}^{k-1}\exp(C_r\Delta_l t)\right) C_f\Delta_k t\right) \\
&= \exp\left(C_r(t_M - t_j)\right) C_M + \sum_{k=j}^{M-1}\left(\exp\left(C_r(t_k - t_j)\right) C_f\Delta_k t\right) \\
&\leq \exp\left(C_r T\right) C_M + \exp\left(C_r T\right)\sum_{k=j}^{M-1}\left(C_f\Delta_k t\right) \\
&= \exp\left(C_r T\right)\left(C_M + C_f T\right) =: C.
\end{aligned}$$

Let v be the vector of all values v_i^j. Then it holds

$$\|v\|_\infty = \max_{j\in\{1,2,\dots,M\}} \max_{i\in\{1,2,\dots,N\}} |v_i^j| \leq \max_{j\in\{1,2,\dots,M\}} C_j \leq C.$$

As this estimation is independent of the grid spacing, the scheme (5.47) is stable. □

Theorem 9 (Convergence of Tree-Grid method)**.** *The approximation computed by the Tree-Grid method defined by Algorithm 5 for solving the SCP* (5.1),(5.2) *and the corresponding HJB equation* (5.4),(5.5) *satisfying the strong uniqueness property (see [4]) and stability conditions defined in Property 1 converges to the viscosity solution of this SCP (and HJB equation).*

Proof. The proof follows from Theorem 1 and Lemmas 5, 6, 8. □

Remark 9. *Actually the approximation converges to the viscosity solution of the HJB equation restricted to the domain* $[s_L, s_R] \times [0, T]$ *with boundary conditions defined by functions* $BC_L(s,t)$, $BC_R(s,t)$. *We silently assumed that these boundary functions are chosen consistently with the viscosity solution - an assumption that is frequently done in literature, as choosing boundary conditions is problem specific.*

5.4 Numerical example: uncertain volatility model

In this as well as in the following section we will compare the performance of the Tree-Grid method with the classic implicit finite difference method for the HJB equation presented in [16] on examples from finance. In the first example no artificial diffusion is needed in contrast to the second example in Section 5.5. We note that in order to validate the Tree-Grid method, we tested it also with an uncontrolled Black-Scholes equation, which posses an closed-form formula for solution. The method was convergent, however standard FDMs provided better results in this case. The numerical methods were implemented in Matlab and tested on an Intel Core i7-4770 CPU 3.40GHz computer with 8 GB RAM.

Our first example of the usefulness of Tree-Grid method is the problem of option pricing under the uncertain volatility model:

Example 3 (Uncertain volatility model)**.** *The setting of the uncertain volatility model is similar to the famous Black-Scholes model, the only difference is that the volatility is uncertain, only known to lie in some interval. Using this model, we can compute maximal (best case) and minimal (worst case) option prices. Here we present the results for the best case option price* V *that can be, according to [17], computed using the HJB equation*

$$\frac{\partial V}{\partial t} + \max_{\theta\in\Theta}\Big(\frac{\theta^2 S^2}{2}\frac{\partial^2 V}{\partial S^2} + S\frac{\partial V}{\partial S} - rV\Big) = 0. \tag{5.70}$$

Here, t *represents time,* S *is asset price,* θ *(control variable) is volatility,* $\Theta = \{\sigma_{min}, \sigma_{max}\}$ *are the minimal and maximal values of the volatility and* r *is the risk-free interest rate. For comparison reasons we used the parameters from [17]:* $r = 0.04$ *and* $\Theta = \{0.3, 0.45\}$.

Computational domain: The maturity of the option will be six months ($T = 0.5$), the space domain will be restricted to $S \in [0, 500]$. The grid will be uniformly spaced in time,

and non-uniformly in space (nodes will be more dense near to "edges" of terminal condition and less dense near to boundaries of the computational domain)

Terminal and boundary conditions: Terminal and boundary conditions will be also set as in [17]. We will use a butterfly-spread payoff around 100 as the terminal condition:

$$V(T,S) = V_T(S) = \begin{cases} S - 95 & \text{if } 95 < x \leq 100 \\ 105 - S & \text{if } 100 < S \leq 105 \\ 0 & \text{else.} \end{cases},$$

and the Dirichlet boundary conditions:

$$V(S_{min}, t) = BC_L(S) = 0, \quad V(S_{max}, t) = BC_R(S) = 0,$$
$$[S_{min}, S_{max}] = [0, 500].$$

Numerical results: Now we will present results of numerical solutions of the option pricing problem in uncertain volatility model computed on grids with different levels of refinement. With A^k, let us denote the approximation computed on the k-th refinement level. N_t^k will denote number of time-nodes on the $k-th$ refinement level, and N_s^k will denote number of space-nodes on the $k-th$ refinement level. The error of the approximation on the k-th space- and time-refinement level is denoted as $Err\ A^k$ and estimated by the formula

$$Err\ A^k = \|A^k - A^{ref}\|_1, \tag{5.71}$$

where A^{ref} denotes a reference solution. The experimental order of convergence on the k-th space- and time-refinement level is denoted denoted as $EOC\ A^k$ and computed using the formula (4.10).

Table 5.1: Uncertain volatility model, $\Delta t = c \cdot \Delta s$. Error, EOC and computational time of the approximation $A^{k,k}$ for the classic implicit and the Tree-Grid methods, for different numbers of nodes.

			Classic Implicit			Tree-Grid		
k	N_t^k	N_s^k	Err	EOC	Time	Err	EOC	Time
1	51	100	6.03E-005	-	0.0398	4.29E-005	-	0.0031
2	101	199	1.50E-005	2.01	0.0854	2.09E-006	4.36	0.0033
3	201	397	4.07E-006	1.88	0.1990	2.19E-006	-0.07	0.0068
4	401	793	1.10E-006	1.88	0.5138	4.56E-007	2.26	0.0157
5	801	1585	2.89E-007	1.93	1.4758	1.77E-007	1.36	0.0386
6	1601	3169	7.11E-008	2.02	5.3508	2.43E-008	2.87	0.1057
7	3201	6337	1.60E-008	2.15	19.7059	1.32E-008	0.87	0.3350
8	6401	12673	2.99E-009	2.42	78.0347	2.30E-009	2.53	1.1445
9	12801	25345	3.36E-010	3.16	312.6208	2.88E-010	3.00	4.3767

As reference solution we will use an approximation computed on a grid with 25601 time-nodes and 50689 space nodes using the classic implicit method with maximal use of central differences [48]. The Tables 5.1 and 5.2 present the results for approximations computed with fixed ratio between time-step size and space-step size ($\Delta t = c \cdot \Delta s$) and with fixed ratio between time-step size and square of space-step size ($\Delta t = c \cdot (\Delta s)^2$). Figure 5.3 illustrates the results.

Table 5.2: Uncertain volatility model, $\Delta t = c \cdot (\Delta s)^2$. Error, EOC and computational time of the approximation $A^{k,k}$ for the classic implicit and the Tree-Grid methods, for different numbers of nodes.

			Classic Implicit			Tree-Grid		
k	N_t^k	N_s^k	Err	EOC	Time	Err	EOC	Time
1	51	1585	3.43E-005	-	0.1406	1.16E-005	-	0.0204
2	201	3169	3.63E-006	3.24	0.7650	4.95E-007	4.55	0.0515
3	801	6337	2.82E-007	3.69	5.0952	4.84E-008	3.35	0.1524
4	3201	12673	1.59E-008	4.15	40.3774	7.27E-009	2.74	0.6655
5	12801	25345	3.36E-010	5.57	312.6208	2.88E-010	4.66	4.3767

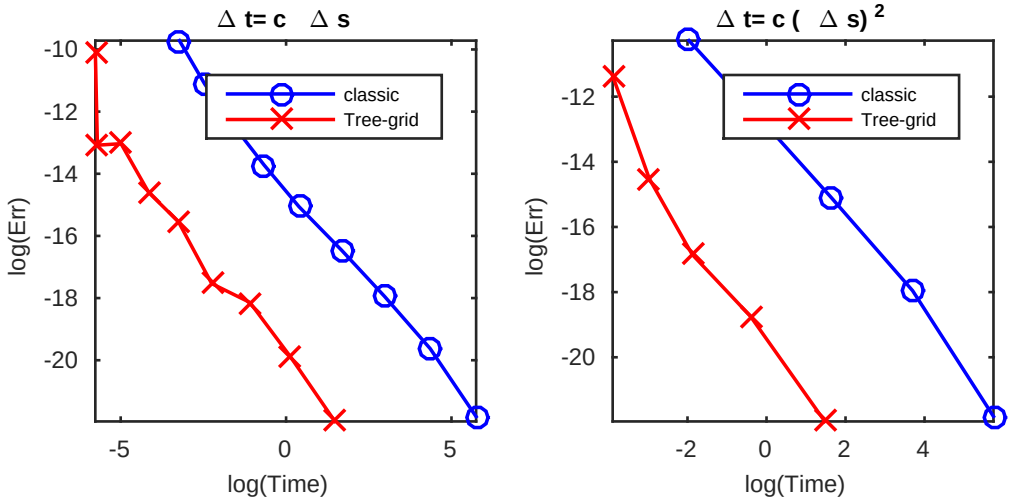

Figure 5.3: Uncertain volatility model. Comparison of natural logarithm of estimated absolute error of the approximation of solution against natural logarithm of computational time (in seconds) for the classic implicit and Tree-Grid method (Illustration of the results from Tables 5.1, 5.2).

From Tables 5.1 and 5.2 and Figure 5.3, it is clear that the Tree-Grid method was not only significantly faster than the classic implicit FDM, but also its error was slightly smaller. Therefore the Tree-Grid method is clearly superior for this model. We note that in this example the condition (5.30) was always met and therefore no artificial diffusion was needed.

5.5 Numerical example: passport option pricing problem

In this second example, we will test the Tree-Grid method and the classic implicit method on the HJB equation for passport option pricing. This setting is described in Example 2 from Chapter 4.

Computational domain: The maturity of the option will be one year ($T = 1$), the space domain will be restricted to $[-3, 4]$. The grid will be uniformly spaced in time, and non-uniformly in space (nodes will be more dense near to zero and less dense near to the

boundaries of the computational domain)

Terminal and boundary conditions: As terminal condition we will use the "capped" payoff:

$$V(T,x) = V_T(x) = \begin{cases} 0 & \text{if } x \leq 0 \\ x & \text{if } 0 < x \leq 1 \\ 1 & \text{if } x > 1 \end{cases},$$

and the Dirichlet boundary conditions:

$$V(x_{min},t) = BC_L(x) = 0, \quad V(x_{max},t) = BC_R(x) = 1,$$
$$[x_{min}, x_{max}] = [-3, 4].$$

Numerical results: In this part we use the same definitions of A^k, Err, EOC and as in previous numerical model.

Table 5.3: Passport option pricing, $\Delta t = c \cdot \Delta s$. Error, experimental order of convergence and computational time of the approximation $A^{k,k}$ for the classic implicit and the Tree-Grid methods, for different numbers of nodes.

			Classic Implicit			Tree-Grid		
k	N_t^k	N_s^k	Err	EOC	Time	Err	EOC	Time
1	51	24	3.42E-007	-	0.0301	1.65E-005	-	0.0037
2	101	47	1.03E-006	-1.59	0.0569	1.64E-006	3.33	0.0051
3	201	93	3.41E-007	1.59	0.1205	1.00E-006	0.71	0.0089
4	401	185	6.26E-007	-0.88	0.2825	2.73E-006	-1.45	0.0177
5	801	369	4.32E-007	0.53	0.8047	2.04E-006	0.42	0.0393
6	1601	737	1.89E-007	1.19	2.0047	9.29E-007	1.13	0.0983
7	3201	1473	8.75E-008	1.11	5.8793	3.99E-007	1.22	0.2735
8	6401	2945	3.76E-008	1.22	21.2013	1.62E-007	1.30	0.8108
9	12801	5889	1.44E-008	1.38	77.8576	6.23E-008	1.38	2.8490
10	25601	11777	4.48E-009	1.68	312.0041	2.15E-008	1.53	11.2965
11	51201	23553	7.83E-010	2.52	1080.4906	6.01E-009	1.84	38.0828

Table 5.4: Passport option pricing, $\Delta t = c \cdot (\Delta s)^2$. Error, experimental order of convergence and computational time of the approximation $A^{k,k}$ for the classic implicit and the Tree-Grid methods, for different numbers of nodes.

			Classic Implicit			Tree-Grid		
k	N_t^k	N_s^k	Err	EOC	Time	Err	EOC	Time
1	51	737	2.01E-007	-	0.0759	1.50E-006	-	0.0073
2	201	1473	9.01E-008	1.16	0.3969	5.00E-007	1.59	0.0239
3	801	2945	3.81E-008	1.24	2.6518	1.83E-007	1.45	0.1148
4	3201	5889	1.45E-008	1.39	19.4999	6.65E-008	1.46	0.7204
5	12801	11777	4.50E-009	1.69	157.2967	2.22E-008	1.58	5.0818
6	51201	23553	7.83E-010	2.52	1080.4906	6.01E-009	1.89	38.0828

As reference solution we will use the approximation computed on a grid with 102401

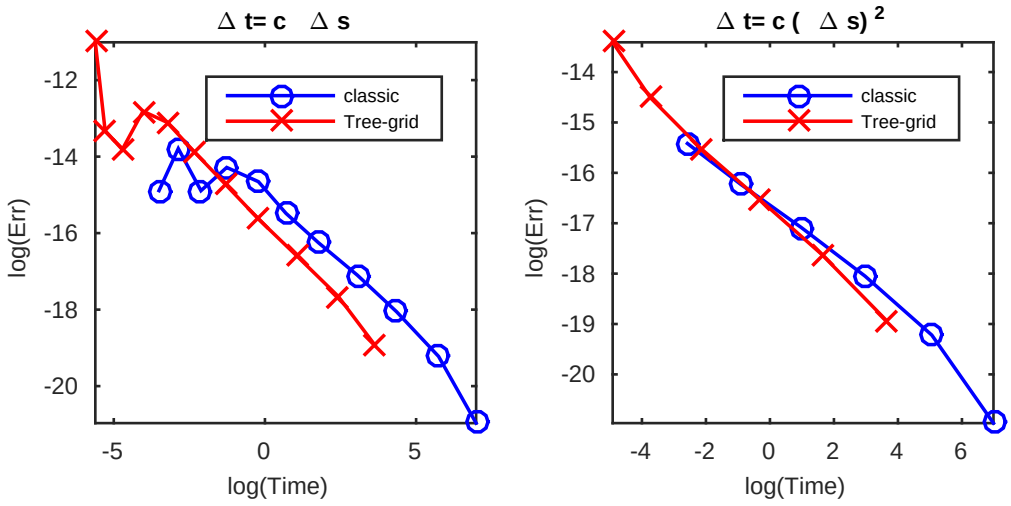

Figure 5.4: Passport option pricing, "capped" payoff. Comparison of natural logarithm of estimated absolute error of the approximation of solution against natural logarithm of computational time (in seconds) for the classic implicit, and Tree-Grid method. (Illustration of results from Tables 5.3, 5.4)

time-nodes and 47105 space nodes using again the classic implicit method with maximal use of central differences. As in the previous numerical example, the Tables 5.3 and 5.4 and Figure 5.4 illustrate the results. However, as the diffusion is vanishing in this case, artificial diffusion was needed in Tree-Grid method in contrast to the uncertain volatility model. The error was smaller in the case of classic implicit FDM, however, taking into account the low computational time of the Tree-Grid method, the Tree-Grid method was still superior on some grids. Moreover one should note that both examples were done with a reference solution computed with implicit FDM, so the implicit method was favored. This larger error of the Tree-Grid method can be probably explained by the additional artificial diffusion term needed to stabilize the scheme. The growth of experimental order of convergence with grid refinement results from using solution computed on finest grid as reference solution. For solutions computed on fine grid, the error computed using such reference solution is smaller than in case of using the exact solution as a reference solution. However this difference is not so huge in case of the coarse grid solution for which the finest grid solution is a good approximation of exact solution.

In this chapter we presented the Tree-Grid method for solving one-dimensional stochastic control-problems and related HJB equations. Being explicit, this method is much faster than the standard implicit FDMs, while still remaining unconditionally stable and convergent on any time-space-grid. However, the method relays on brute-force search of the optimal control, and therefore may be slow for very large numbers of control. We will address and solve this issue in the next chapter.

Chapter 6

Tree-Grid method with control independent stencil

The advantages of the Tree-Grid method is its independence on the space-stepping of the grid, as well as its unconditional convergence and explicitness. However, as well as in FDMs and Markov chain methods, an optimization problem needs to be solved in each step. In the original Tree-Grid method from the previous Chapter 5, solving this optimization problem is done by a brute-force search in the control space, what may be time consuming if this control space is large. In this chapter, we present a modification of the Tree-Grid method, that will allow us to solve the optimization problem more effectively. This chapter is based on the paper [27].

6.1 Tree-Grid method revisited

At first we will quickly recapitulate the Tree-Grid method algorithm. We compute the approximation of the solution on a rectangular domain $[s_L, s_R] \times [0, T]$ with some grid as in usual finite difference schemes for PDEs. The grid-points are denoted as $[s_i, t_j]$, $i \in \{1, 2, \ldots, N\}$, $j \in \{1, 2, \ldots, M\}$, $k < l \Rightarrow s_k < s_l, t_k < t_l$, $t_1 = 0, t_M = T$, $s_1 = s_L, s_N = s_R$. The grid is possibly non-equidistant in space with space-steps $\Delta_i s = s_{i+1} - s_i$ and $\Delta s = \max_i \Delta_i s$. We will use an equidistant discretization in time with a time-step Δt. A generalization to non-equidistant time-stepping is straightforward, however the implementation is less effective in means of computational time in that case. The numerical approximation of $V(s_i, t_j)$ will be denoted by v_i^j. To underline the dependence of stencil nodes, probabilities and values in these stencil nodes defined in the previous chapter on the control variable and current state s_i, we will denote:

- $s_+, s_o s_-$ as $s_{(i+,\theta)}, s_i, s_{(i-,\theta)}$,
- p_+, p_o, p_- as $p_{(i+,\theta)}, p_{(i,\theta)}, p_{(i-,\theta)}$,
- v_-^{j+1}, v_o^{j+1}, v_+^{j+1} as $v_{(i-,\theta)}^{j+1}, v_i^{j+1}, v_{(i+,\theta)}^{j+1}$.

Here we use $s_o = s_i$ to get a consistent scheme according to Lemma 5.

The whole scheme is then defined by the discrete approximation of the dynamic programming equation (5.3)

$$v_i^j = \max_{\theta \in \Theta} \Big(f_i^j(\theta)\Delta t + (1 + r_i^j(\theta)\Delta t) \cdot \Big(p_{(i-,\theta)} v_{(i-,\theta)}^{j+1} + p_{(i,\theta)} v_i^{j+1} + p_{(i+,\theta)} v_{(i+,\theta)}^{j+1} \Big) \Big). \tag{6.1}$$

for $i = 2, 3, \ldots, N-1$ and

$$v_1^j = BC_L(s_1, t_j), \quad v_N^j = BC_R(s_N, t_j). \tag{6.2}$$

Here, $f_i^j(\theta) = f(s_i, t_j, \theta)$, $r_i^j(\theta) = r(s_i, t_j, \theta)$ and

$$v_{(i*,\theta)}^{j+1} = \begin{cases} v_k^{j+1} \quad \text{so that } s_k = s_{(i*,\theta)} & \text{if } s_{(i*,\theta)} \in \{s_1, s_2, \ldots, s_N\} \\ BC_L(s_{(i*,\theta)}, t_{j+1}) & \text{if } s_{(i*,\theta)} < s_1 \\ BC_R(s_{(i*,\theta)}, t_{j+1}) & \text{if } s_{(i*,\theta)} > s_N \end{cases}$$

for the $* \in \{-, +\}$. Here $BC_L(s,t)$ and $BC_R(s,t)$ are functions defining an approximation of the value function behind the boundaries and $s_{(i-,\theta)}$, s_i, $s_{(i+,\theta)}$ are states that the discretized process may attain with the probabilities $p_{(i-,\theta)}$, p_i, $p_{(i+,\theta)}$ under the control θ after the time-step Δt if the previous state was s_i. It holds $s_{(i-,\theta)} < s_i < s_{(i+,\theta)}$. In order to match the moments of this discretized process with the original time-continuous process (5.2) the probabilities are chosen in the following manner:

$$p_{(i-,\theta)} = \frac{-\mu\Delta t(\Delta_{i+}s - \mu\Delta t) + Var}{\Delta_{-i}s(\Delta_{-i}s + \Delta_{i+}s)}, \tag{6.3}$$

$$p_{(i,\theta)} = \frac{(-\Delta_{-i}s - \mu\Delta t)(\Delta_{i+}s - \mu\Delta t) + Var}{-\Delta_{-i}s\Delta_{i+}s}, \tag{6.4}$$

$$p_{(i+,\theta)} = \frac{(-\Delta_{-i}s - \mu\Delta t)(-\mu\Delta t) + Var}{(\Delta_{i+}s + \Delta_{-i}s)\Delta_{i+}s}. \tag{6.5}$$

Here, $\Delta_{i+}s = s_{(i+,\theta)} - s_i$, $\Delta_{-i}s = s_i - s_{(i-,\theta)}$, $\mu := \mu(s_i, t_j, \theta)$ and $Var := Var(s_i, t_j, \theta)$ is chosen in such manner, that $Var/\Delta t$ is equal or at least converges to $\sigma^2(s_i, t_j, \theta)$ with $\Delta t, \Delta s \to 0$. As explained in Chapter 5, these probabilities sum up to one. However, we need to choose states $s_{(i-,\theta)}$, $s_{(i+,\theta)}$ such that all probabilities are positive. Let us assume that the drift μ is positive. Then $p_{(i+,\theta)}$ is positive, and $p_{(i-,\theta)}$, $p_{(i,\theta)}$ are positive if the following condition holds:

$$\Delta_{-i}s\Delta_{i+}s + \mu\Delta t(\Delta_{i+}s - \Delta_{-i}s) \geq (\mu\Delta t)^2 + Var \geq \mu\Delta t\Delta_{i+}s. \tag{6.6}$$

We choose

$$s_{(i-,\theta)} = \left\lfloor s_i - \sqrt{(\mu(s_i, t_j, \theta)\Delta t)^2 + Var(s_i, t_j, \theta)} \right\rfloor_s, \tag{6.7}$$

$$s_{(i+,\theta)} = \left\lceil s_i + \sqrt{(\mu(s_i, t_j, \theta)\Delta t)^2 + Var(s_i, t_j, \theta)} \right\rceil_s, \tag{6.8}$$

where $\lceil\rceil_s$ and $\lfloor\rfloor_s$ were defined in Section 5.2.3. Now it holds

$$\sqrt{(\mu\Delta t)^2 + Var} \leq \Delta_{-i}s, \Delta_{i+}s \leq \sqrt{(\mu\Delta t)^2 + Var} + \Delta s \tag{6.9}$$

and the first inequality in (6.6) holds. For the second inequality in (6.6) it is sufficient if

$$(\mu\Delta t)^2 + Var \geq \left(\sqrt{(\mu\Delta t)^2 + Var} + \Delta s\right)\mu\Delta t. \tag{6.10}$$

For $Var = A(s_i, t_j, \theta)$ with

$$A(s_i, t_j, \theta) = 1/2\left(-(\mu\Delta t)^2 + 2|\mu|\Delta t\Delta s + |\mu|\Delta t\sqrt{(\mu\Delta t)^2 + 4|\mu|\Delta t\Delta s}\right) \tag{6.11}$$

condition (6.10) is fulfilled as equality, for larger Var as inequality. Therefore we set

$$Var = \max\left(\sigma^2(s_i, t_j, \theta)\Delta t, A(s_i, t_j, \theta)\right) \tag{6.12}$$

and compute $s_{(i-,\theta)}$, $s_{(i+,\theta)}$ according to (6.7), (6.8) using this value. We should note, that in (6.11) we replaced μ with $|\mu|$ to cover also the analogous case of a negative drift μ. Now, also the second part of the inequality (6.6), is fulfilled. It holds $Var/\Delta t \to \sigma^2(s_i, t_j, \theta)$ with $\Delta t, \Delta s \to 0$ and it is easy to check that the difference $|Var - \sigma^2(s_i, t_j, \theta)\Delta t|$ is smaller or equal than in Chapter 5. Following Chapter 5, the scheme is then consistent and formula (6.12) is even better than the original version from [28] presented in Chapter 5, as potentially less artificial diffusion is added.

6.2 Modification: control-independent stencil

The dependence of the possible states $s_{(i-,\theta)}$, $s_{(i+,\theta)}$ on the control variable θ implies also a dependence of $v^{j+1}_{(i-,\theta)}$, $v^{j+1}_{(i+,\theta)}$ on θ and makes the optimization problem in (6.1) harder to solve. Therefore, our goal now is to find a θ-independent choice of possible states s_{i-}, s_{i+}, while preserving condition (6.6) (and its analogue for negative drift).

6.2.1 Derivation of the modified scheme

We will assume a positive drift $\mu(s_i, t_j, \theta)$, the case of negative drift is treated analogously.

Let us define

$$\begin{aligned} W_M &= \max_{\theta\in\Theta}\left(\sigma^2(s_i, t_j, \theta)\Delta t + (\mu(s_i, t_j, \theta)\Delta t)^2\right) \\ &= \sigma^2(s_i, t_j, \theta_M)\Delta t + (\mu(s_i, t_j, \theta_M)\Delta t)^2, \end{aligned} \tag{6.13}$$

$$E = \max_{\theta\in\Theta} |\mu(s_i, t_j, \theta)\Delta t|, \tag{6.14}$$

$$W_E = 1/2\left(E^2 + 2\Delta s E + E\sqrt{E^2 + 4\Delta s E}\right). \tag{6.15}$$

It holds $W_E = E(\sqrt{W_E} + \Delta s)$ and for all $W \geq W_E : W > E(\sqrt{W} + \Delta s)$. Finally, let us define

$$W = \max(W_E, W_M) \tag{6.16}$$

and

$$s_{i-} = \left\lfloor s_i - \sqrt{W} \right\rfloor_s \geq s_i - (\sqrt{W} + \Delta s), \tag{6.17}$$

$$s_{i+} = \left\lceil s_i + \sqrt{W} \right\rceil_s \leq s_i + (\sqrt{W} + \Delta s). \tag{6.18}$$

Moreover, we redefine also the variance $Var(s_i, t_j, \theta)$:

$$Var = \max\left(\sigma^2\Delta t, \quad |\mu\Delta t|\left(\sqrt{W} + \Delta s\right) - (\mu\Delta t)^2\right), \tag{6.19}$$

where $\sigma = \sigma(s_i, t_j, \theta)$, $\mu = \mu(s_i, t_j, \theta)$. It is easy to check that $Var/\Delta t \to \sigma^2$ as $\Delta t, \Delta s \to 0$

and therefore the consistency is preserved. Now it holds

$$\Delta_{-i}s,\ \ \Delta_{i+}s \geq \sqrt{W} \geq \sqrt{W_M} = \sqrt{\sigma^2(s_i,t_j,\theta_M)\Delta t + (\mu(s_i,t_j,\theta_M)\Delta t)^2}.$$

Therefore it also holds

$$\begin{aligned}\Delta_{-i}s\Delta_{i+}s + \mu\Delta t(\Delta_{-i}s - \Delta_{i+}s) &\geq \sigma^2(s_i,t_j,\theta_M)\Delta t + (\mu(s_i,t_j,\theta_M)\Delta t)^2 \\ &\geq \sigma^2(s_i,t_j,\theta)\Delta t + (\mu(s_i,t_j,\theta)\Delta t)^2. \end{aligned} \tag{6.20}$$

It also holds

$$\Delta_{-i}s\Delta_{i+}s + \mu\Delta t(\Delta_{-i}s - \Delta_{i+}s) \geq W \geq E(\sqrt{W} + \Delta s) \geq |\mu\Delta t|(\sqrt{W} + \Delta s). \tag{6.21}$$

From (6.20) and (6.21) the first inequality of (6.6) holds. The second inequality of (6.6) holds, because

$$Var + (\mu(s_i,t_j,\theta)\Delta t)^2 \geq \mu\Delta t\Delta_{i+}s. \tag{6.22}$$

Equation (6.22) also holds if we replace $\mu\Delta t\Delta_{i+}s$ with $|\mu\Delta t|\Delta_{-i}s$ which is important for the case of a negative drift. Now substituting $s_{(i-,\theta)}, s_{(i+,\theta)}$ with s_{i-}, s_{i+} for all values of θ, we get also θ-independent values $v^{j+1}_{(i-,\theta)}, v^{j+1}_{(i+,\theta)}$ (that can be written as v^{j+1}_{i-}, v^{j+1}_{i+}, and the scheme (6.1) still remains consistent and monotone ($p_{(i-,\theta)}, p_i, p_{(i-,\theta)} \geq 0$). In the next section, we employ this *"modified scheme"* to effectively solve the control problem arising in each node in equation (5.3).

6.2.2 Analytical solution of the control problem in the modified scheme

According to Section 5.2.6 where also relationship of the Tree-Grid method with the FDMs is discussed, the numerical scheme (6.1) can be written as

$$\begin{aligned} v_i^j =& \max_{\theta\in\Theta}\Big(f_i^j(\theta)\Delta t + (1 + r_i^j(\theta)\Delta t) \\ &\cdot\Big(v_i^{j+1} + \mu_i^j(\theta)\Delta_j t D_1 v_i^{j+1} + 1/2\left(Var_i^j(\theta) + (\mu_i^j(\theta)\Delta_j t)^2\right)D_2 v_i^{j+1}\Big)\Big) \\ :=& \max_{\theta\in\Theta} F_i^j(\theta), \end{aligned} \tag{6.23}$$

where $\mu_i^j(\theta) = \mu(s_i,t_j,\theta)$, $Var_i^j(\theta) = Var(s_i,t_j,\theta)$ and D_1, D_2 are standard finite difference approximations of the first and second derivative on nonuniform grids:

$$D_1 v_i^{j+1} = \left(\frac{s_{i+}-s_i}{s_{i+}-s_{i-}}\right)\frac{v_i^{j+1}-v_{i-}^{j+1}}{s_i - s_{i-}} + \left(\frac{s_i - s_{i-}}{s_{i+}-s_{i-}}\right)\frac{v_{i+}^{j+1}-v_i^{j+1}}{s_{i+}-s_{i-}}, \tag{6.24}$$

$$D_2 v_i^{j+1} = \left(\frac{v_{i+}^{j+1}-v_i^{j+1}}{s_{i+}-s_{i-}} - \frac{v_i^{j+1}-v_{i-}^{j+1}}{s_i - s_{i-}}\right) \Big/ \left(\frac{s_{i+}-s_{i-}}{2}\right). \tag{6.25}$$

Now, under the modification presented in the previous section, s_{i+} and s_{i-} are control-independent and hence also $D_1 v_i^{j+1}$ and $D_2 v_i^{j+1}$ are control independent. Then, for a fixed node (s_i,t_j) the function $F_i^j(\theta)$ is some combination of the functions $f_i^j(\theta)$, $r_i^j(\theta)$, $\mu_i^j(\theta)$ and $Var_i^j(\theta)$. As these functions are typically in closed form, it should be possible to search for the $\max_{\theta\in\Theta} F_i^j(\theta)$ analytically, and it is not necessary to discretize Θ (if it is

for example an interval).

However, $Var_i^j(\theta)$ is defined as the maximum of two different functions in (6.19) and therefore may switch its form in several points of the interval Θ. This can make the analytical computation of $\max_{\theta\in\Theta} F_i^j(\theta)$ quite difficult. This problem is not present, if we can assure $Var_i^j(\theta) = \sigma(s_i, t_j, \theta)^2\Delta t$. That condition is typically fulfilled for a relatively large diffusion coefficient σ compared to the drift coefficient μ.

6.2.3 The Fibonacci algorithm for finding the optimal control

Because of the possible complications arising by the search for the analytical solution of the control problem $\max_{\theta\in\Theta} F_i^j(\theta)$ presented in the previous section, our aim is now to present another, more straightforward approach. Let us suppose:

1. Θ is a one-dimensional interval.
2. Discount rate $r_i^j(\theta)$ is constant in θ.
3. Increment rate $f_i^j(\theta)$ and drift $\mu_i^j(\theta)$ are linear in θ.
4. Volatility $\sigma^2(s_i, t_j, \theta)$ is convex in θ.

These conditions are fulfilled in many applications. Under these conditions, it is easy to verify, that also $1/2(Var_i^j(\theta) + (\mu_i^j(\theta)\Delta_j t)^2)$ is convex. Then, $F_i^j(\theta)$ is convex or concave and therefore has at most one local (and global) extreme inside the interval Θ and has at least one extreme on the boundary. This makes the problem $\max_{\theta\in\Theta} F_i^j(\theta)$ suitable for the Fibonacci algorithm for maximum search [15]:

Algorithm 6 Fibonacci algorithm for finding the optimal control

1: Discretize the interval Θ into Φ_n points $\theta_1, \theta_2, \ldots \theta_{\Phi_n}$ where Φ_n is the n-th Fibonacci number.
2: Set $a = 1$, $b = \Phi_n$, $c_1 = \Phi_{n-2}$, $c_2 = \Phi_{n-1}$
3: **for** $j = n-1, n-2, \ldots, 3$ **do**
4: **if** $F_i^j(\theta_{c_1}) > F_i^j(\theta_{c_1})$ **then**
5: $b := c_2$;
6: $c_2 := c_1$;
7: $c_1 := a - 1 + \Phi_{j-2}$;
8: **else**
9: $a := c_1$;
10: $c_1 := c_2$;
11: $c_2 := a - 1 + \Phi_{j-1}$;
12: **end if**
13: **end for**
14: $\max_{\theta\in\Theta} F_i^j(\theta) \approx \max(F_i^j(\theta_a), F_i^j(\theta_{c_1}), F_i^j(\theta_{c_2}), F_i^j(\theta_b), F_i^j(\theta_1), F_i^j(\theta_{\Phi_n}))$

In the last step of the algorithm we included for testing also values $F_i^j(\theta_1), F_i^j(\theta_{\Phi_n})$ for the case that the function $F_i^j(\theta)$ is convex and the maximum is on the boundary. The

computational time of the Fibonacci algorithm is $\mathcal{O}(n) = \mathcal{O}(\log(\Phi_n))$ which is much better than the computational time of the brute-force search approach from Chapter 5 that is $\mathcal{O}(\Phi_n)$ for Φ_n controls.

6.3 Numerical example: passport option pricing problem

We will test this modified Tree-Grid method with a control-independent stencil and the Fibonacci algorithm for control search on a Passport option pricing problem with the same parameters, terminal and boundary conditions as in previous chapter. The equation for passport option pricing is described in the Example 2 in Chapter 4.

Computational domain: The maturity of the option will be one year ($T = 1$), the spatial domain will be restricted to $[-3, 4]$. The grid will be uniformly spaced in time, and non-uniformly in space. On the coarsest grid, the time-step size is 0.01. At each refinement, a four-times smaller time-step is taken. The basis for the space grid is the vector of nodes:

$$\begin{aligned} S_0 = [&-3, -2, -1.5, -1, -0.75, -0.5, -0.375, -0.25, -0.1875, -0.125, \\ &-0.0625, 0, 0.0625, 0.125, 0.1875, 0.25, 0.375, 0.5, 0.75, 1, 1.5, 2, 3, 4] \end{aligned} \tag{6.26}$$

On the coarsest grid 15 another nodes are equidistantly inserted between each two neighbouring nodes of S_0. Moreover, at each refinement a new space-node is inserted between each two neighbouring space-nodes.

Terminal and boundary conditions: As terminal condition we use the "capped" payoff:

$$V(T, x) = V_T(x) = \begin{cases} 0 & \text{if } x \leq 0 \\ x & \text{if } 0 < x \leq 1 \\ 1 & \text{if } x > 1 \end{cases},$$

and the Dirichlet boundary conditions:

$$\begin{aligned} &V(x_{min}, t) = BC_L(x_{min}) = 0, \quad V(x_{max}, t) = BC_R(x_{max}) = 1, \\ &x_{min} = -3, \quad x_{max} = 4. \end{aligned}$$

Results: In Figure 6.1 we illustrate the results of numerical simulations. The left figure presents a comparison of error and computational time of the original Tree-Grid method with the modified Tree-Grid method with control-independent stencil for different discretizations. To compute the error, we used as a benchmark solution a solution computed on a very fine grid (having twice as much space and time nodes as the grid at the last refinement level) with an implicit FDM from [48]. In both cases, the control interval was discretized into 9 different controls, and we used brute-force search for the optimal control. We see that the modified Tree-Grid (TG) method converges, however the original method performs better. This may be of course compensated for finer discretizations of the control interval, if the optimal control is searched analytically or with a Fibonacci search algorithm in the modified scheme.

This illustrates the right figure. Here we used a coarse grid with 24 space-nodes defined by (6.26), 100 (equidistant) time-steps and a varying number of controls. As number of

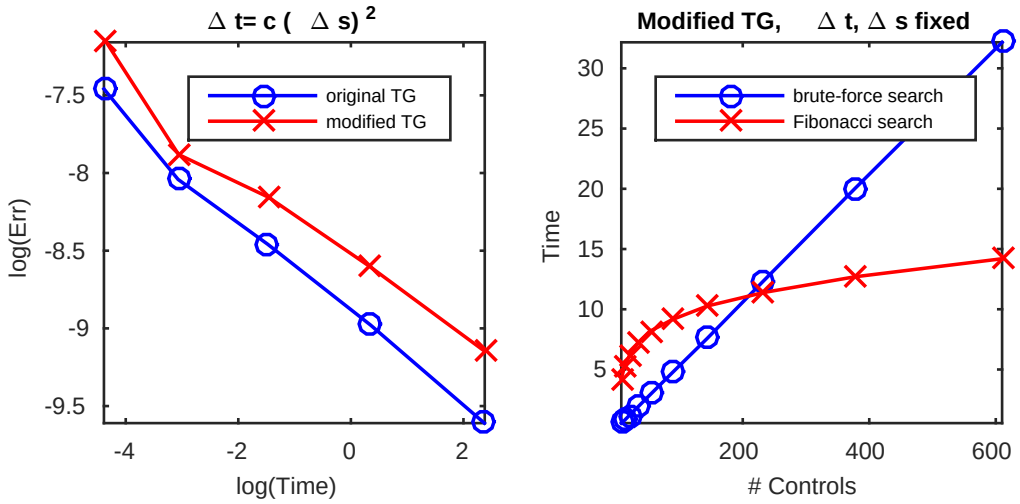

Figure 6.1: Left: Comparison of the natural logarithm of estimated absolute error of numerical solution against natural logarithm of computational time (in seconds) for the original Tree-Grid (TG) method and the modified Tree-Grid method with control independent stencil. Brute-force search for optimal control is done in both cases. Right: Computational time (in seconds) of the modified Tree-Grid method with control independent stencil for different number of controls in cases of brute-force search and Fibonacci search for optimal control.

controls (on the x-axis), we used the Fibonacci numbers from the fifth (8) to the 14th (610). We compared the computational time of the modified Tree-Grid method with a brute-force search for control and with a Fibonacci search for control. We observe that for a large number of controls the Fibonacci search performs better due to its logarithmic time-complexity (in contrast to the linear time complexity of brute-force search). We note that the actual values presented here in the figure are strongly implementation dependent, but they are sufficient in illustrating the proof of concept.

Chapter 7

Two-dimensional Tree-Grid method

Up to now we were concerned only with numerical methods for problems with one space dimension. In this chapter we will present our Tree-Grid method for two space dimensions. This method is again explicit and unconditionally convergent for any grid. This chapter is based on the paper [29].

7.1 Recapitulation: problem formulation

For convenience, we repeat here the problem formulation from Chapter 2. We are concerned with searching for the value function $V(x, y, t)$ of the following *2-dimensional general stochastic control problem* (SCP):

$$V(x,y,t) = \max_{\theta(x,y,t)\in\bar{\Theta}} \mathbb{E}\left(\int_t^T \exp\left(\int_t^k r(*_l)dl\right) f(*_k)dk \right.$$
$$\left. + \exp\left(\int_t^T r(*_k)dk\right) V_T(X_T, Y_T)\Big| X_t = x, Y_t = y\right), \quad (7.1)$$
$$dX_t = \mu_x(*_t)dt + \sigma_x(*_t)dW_t^x, \quad (7.2)$$
$$dY_t = \mu_y(*_t)dt + \sigma_y(*_t)dW_t^y, \quad (7.3)$$
$$dW_t^x dW_t^y = \sigma_{xy}(*_t)/(\sigma_x(*_t)\sigma_y(*_t)) \quad (7.4)$$
$$*_t = (X_t, Y_t, t, \theta(X_t, Y_t, t)), \quad 0 < t < T, \quad x, y \in \mathbb{R},$$

where x, y are state variables and t is time. Here, $\bar{\Theta}$ is the space of all *suitable* control functions (see e.g. [31, 50]) from $\mathbb{R}^2 \times [0, T]$ to a set Θ. For our purpose, we will assume Θ to be discrete. If this is not the case, we can easily achieve this property by its discretization. Now following the Bellman's principle, the *dynamic programming equation* holds:

$$V(x,y,t_j) = \max_{\theta(x,y,t)\in\bar{\Theta}_{t_j}} \mathbb{E}\left(\int_{t_j}^{t_{j+1}} \exp\left(\int_{t_j}^k r(*_l)dl\right) f(*_k)dk \right.$$
$$\left. + \exp\left(\int_{t_j}^{t_{j+1}} r(*_k)dk\right) V(X_{t_{j+1}}, Y_{t_{j+1}}, t_{j+1})\Big| X_{t_j} = x, Y_{t_j} = y\right), \quad (7.5)$$

where $0 \le t_j < t_{j+1} \le T$ are some time-points and $\bar{\Theta}_{t_j}$ is a set of control functions from $\bar{\Theta}$ restricted to the $\mathbb{R}^2 \times [t_j, t_{j+1})$ domain. Using this equation (7.5), it can be shown [38], that solving the SCP (7.1)-(7.4) is equivalent to solving the two-dimensional Hamilton-

Jacobi-Bellman (HJB) equation:

$$\frac{\partial V}{\partial t} + \max_{\theta \in \Theta} (LV + r(\cdot)V + f(\cdot)) = 0, \tag{7.6}$$

$$LV = \frac{\sigma_x(\cdot)^2}{2}\frac{\partial^2 V}{\partial x^2} + \sigma_{xy}(\cdot)\frac{\partial^2 V}{\partial x \partial y} + \frac{\sigma_y(\cdot)^2}{2}\frac{\partial^2 V}{\partial y^2} + \mu_x(\cdot)\frac{\partial V}{\partial x} + \mu_y(\cdot)\frac{\partial V}{\partial y} \tag{7.7}$$

$$V(x, y, T) = V_T(x, y), \tag{7.8}$$

$$0 < t < T, \quad x, y \in \mathbb{R}$$

where $\sigma_x(\cdot)$, $\sigma_y(\cdot)$, $\sigma_{xy}(\cdot)$, $\mu_x(\cdot)$, $\mu_y(\cdot)$, $r(\cdot)$, $f(\cdot)$ are functions of x, y, t, θ. We should note that the maximum operator in (7.1) and (7.6) can be replaced by a minimum operator and the whole following analysis will hold analogously.

7.2 Construction of 2D Tree-Grid method

In this section we will derive the two-dimensional Tree-Grid algorithm for solving the problem (7.1)-(7.4). We will use the ideas that were widely explained in chapters 5, 6. We will work on a three-dimensional rectangular domain with two space dimensions and one time-dimension. For a fixed control θ, the candidate for a value in each node $V(x_i, y_j, t_k)$ will be computed from seven values from the next time-layer t_{k+1}. Figure 7.1 illustrates this approach. We will denote these seven values in this context simply as stencil located at (x_i, y_j, t_k). The weights of these seven values can be interpreted as probabilities and therefore we demand, that the moments of such discrete random variable are matching with the moments of the increment of the two-dimensional stochastic process (7.2)-(7.4) with the fixed control θ at least asymptotically. Then, the actual value $V(x_i, y_j, t_k)$ will be computed as a maximum of the candidates. For handling nodes close to the boundary we suppose that we know, how the solution behaves in the outer neighborhood of the boundaries, and that we can describe this behavior with a boundary function $BC(x, y, t)$. The terminal condition in the time-layer $t_M = T$ reads $V(x, y, t_M) = V_T(x, y)$.

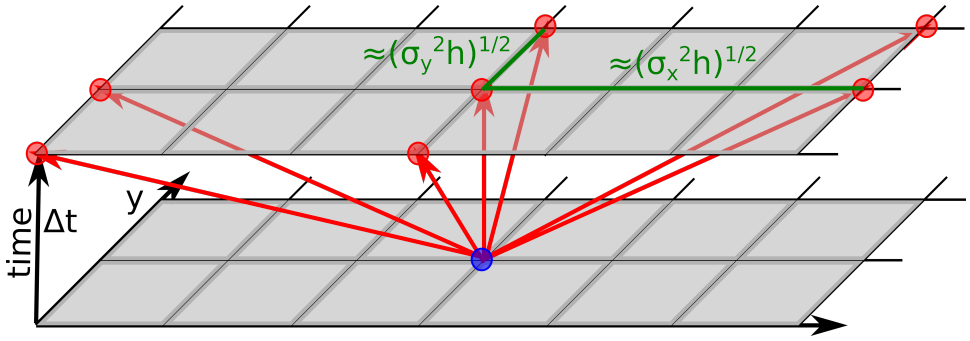

Figure 7.1: Illustration of the two-dimensional Tree-Grid structure. Only the red nodes in later time layer impact the blue node. On the other hand, the red nodes can be interpreted as possible future states if we are in the blue node state. The figure illustrates a situation with positive correlation and a variance that is larger in the x-direction than in the y-direction. The stencil size in each direction is roughly proportional to the square root of the variance coefficient in that direction multiplied by the discretization parameter h.

7.2.1 Notation

At first, before discussing how to choose the stencil nodes around node (x_i, y_j, t_k), and the proper weights, we present here the notation used in the sequel:

- $x_1, x_2, \ldots, x_{N_x}$ -space discretization in the space 1. dimension.
  $y_1, y_2, \ldots, y_{N_y}$ -space discretization in the space 2. dimension.
  $t_1, t_2, \ldots, t_M$ -time discretization .
- $\Delta_k t = t_{k+1} - t_k$ -(current) time-step. We will use equidistant timestepping, $\Delta_k t = \Delta t$ for all $k = 1, 2, \ldots, M-1$. Generalization to non-equidistant timestepping is straightforward.
- $\Delta_i x = x_{i+1} - x_i$, $\Delta_j y = y_{j+1} - y_j$ space-steps in the 1. and the 2. dimension (possibly non-equidistant).
- $\Delta x = \max_i \Delta_i x$, $\Delta y = \max_i \Delta_i y$.
- $h = \max(K \max(\Delta x, \Delta y), \Delta t)$. where $K > 0$ is a parameter used for regulating the stencil size. In the non-equidistant case, Δt should be replaced by $\Delta_k t$.
- $b = h/\Delta t$. In the non-equidistant case, this should be replaced by $b = h/\Delta_k t$ in the following algorithm.
- (x_i, y_j, t_k) -the node for which the stencil is constructed.
- $E_* = \mu_*(x_i, y_j, t_k, \theta)\Delta t$ for $* = x, y$.
- $Var_* = (\sigma_*^2(x_i, y_j, t_k, \theta) + \mathcal{O}(h))\Delta t$ for $* = x, y$, will be determined later.
- $\rho(x_i, y_j, t_k, \theta) = \sigma_{xy}(x_i, y_j, t_k, \theta)/(\sigma_x(x_i, y_j, t_k, \theta)\sigma_y(x_i, y_j, t_k, \theta))$
- $\tilde{\sigma}_{xy}(x_i, y_j, t_k, \theta) = \sigma_{xy}(x_i, y_j, t_k, \theta) + \mathcal{O}\left(\sqrt{h}\right)$, will be determined later.
- $Cov = \left(\sigma_{xy}(x_i, y_j, t_k, \theta) + \mathcal{O}\left(\sqrt{h}\right)\right)\Delta t$, will be determined later.
- $W_* = Var_* + E_*^2$ for $* = x, y$.
- $W_{xy} = Cov + E_x E_y$.
- $v_{i,j}^k$ -numerical approximation of $V(x_i, y_j, t_k)$.

7.2.2 Choosing the stencil nodes

Next, we describe how to choose the stencil nodes around an arbitrary node (x_i, y_j, t_k) for a fixed control θ. The values in these nodes will impact the value in the node (x_i, y_j, t_k).

If $E_x > 0$,

$$x_- = \left\lfloor x_i - \sqrt{2W_x b} \right\rfloor_x, x_+ = \left\lceil \max\left(x_i + \sqrt{2W_x b}, x_i + (x_i - x_-)\right) \right\rceil_x, \tag{7.9}$$

else if $E_x < 0$,

$$x_+ = \left\lceil x_i + \sqrt{2W_x b} \right\rceil_x, x_- = \left\lfloor \min\left(x_i - \sqrt{2W_x b}, x_i - (x_+ - x_i)\right) \right\rfloor_x, \tag{7.10}$$

else ($E_x = 0$),

$$x_+ = \left\lceil x_i + \sqrt{2W_x b} \right\rceil_x, x_- = \left\lfloor x_i - \sqrt{2W_x b} \right\rfloor_x. \tag{7.11}$$

where $\lceil\rceil_x$ resp. $\lfloor\rfloor_x$ denotes rounding to the nearest greater resp. smaller element from $x_1, x_2, \ldots, x_{N_x}$. If such element does not exist, $\lceil z \rceil_x$ resp. $\lfloor z \rfloor_x$ will return just z.

If $E_y > 0$,

$$y_- = \left\lfloor y_j - \sqrt{2W_y b} \right\rfloor_y, y_+ = \left\lceil \max\left(y_j + \sqrt{2W_y b}, y_j + (y_j - y_-)\right) \right\rceil_y, \tag{7.12}$$

else if $E_y < 0$,

$$y_+ = \left\lceil y_j + \sqrt{2W_y b} \right\rceil_y, y_- = \left\lfloor \min\left(y_j - \sqrt{2W_y b}, y_j - (y_+ - y_j)\right) \right\rfloor_y, \tag{7.13}$$

else ($E_y = 0$),

$$y_+ = \left\lceil y_j + \sqrt{2W_y b} \right\rceil_y, y_- = \left\lfloor y_j - \sqrt{2W_y b} \right\rfloor_y. \tag{7.14}$$

where $\lceil\rceil_y$, $\lfloor\rfloor_y$ are defined analogously.

The following nodes with the respective weights (probabilities) will be used in the stencil located at (x_i, y_j, t_k):

- node (x_i, y_j, t_{k+1}) with the probability p_o,
- nodes (x_+, y_j, t_{k+1}) and (x_-, y_j, t_{k+1}) with the probabilities p_{x+} and p_{x-},
- nodes (x_i, y_+, t_{k+1}) and (x_i, y_-, t_{k+1}) with the probabilities p_{y+} and p_{y-},
- nodes (x_+, y_+, t_{k+1}) and (x_-, y_-, t_{k+1}), both with the probability p_{xy} if W_{xy} is non-negative, and nodes (x_+, y_-, t_{k+1}) and (x_-, y_+, t_{k+1}) both with the probability p_{xy} if W_{xy} is negative. In the following algorithm we will focus on the case of non-negative W_{xy}, the case of a negative W_{xy} is treated analogously.

Moreover, we define the difference operators

$$\Delta_+ x = x_+ - x_i, \quad \Delta_- x = x_i - x_-, \tag{7.15}$$

$$\Delta_+ y = y_+ - y_i, \quad \Delta_- y = y_i - y_-. \tag{7.16}$$

7.2.3 Choosing the stencil weights (probabilities)

To match the first two moments of the approximative increment of the stochastic process and of the increment of the discrete process defined by the "stencil nodes" and their probabilities and to ensure that the probabilities are non-negative and sum up to 1, we

demand:

$$p_o + p_{x+} + p_{x-} + p_{y+} + p_{y-} + 2p_{xy} = 1, \tag{7.17}$$
$$(p_{x+} + p_{xy})\Delta_+ x - (p_{x-} + p_{xy})\Delta_- x = E_x, \tag{7.18}$$
$$(p_{y+} + p_{xy})\Delta_+ y - (p_{y-} + p_{xy})\Delta_- y = E_y, \tag{7.19}$$
$$(p_{x+} + p_{xy})(\Delta_+ x)^2 + (p_{x-} + p_{xy})(\Delta_- x)^2 = W_x, \tag{7.20}$$
$$(p_{y+} + p_{xy})(\Delta_+ y)^2 + (p_{y-} + p_{xy})(\Delta_- y)^2 = W_y, \tag{7.21}$$
$$p_{xy}(\Delta_+ x\Delta_+ y + \Delta_- x\Delta_- y) = W_{xy}, \tag{7.22}$$
$$p_o, p_{x+}, p_{x-}, p_{y+}, p_{y-}, p_{xy} \geq 0. \tag{7.23}$$

For negative W_{xy}, only the condition (7.22) changes to

$$p_{xy}(\Delta_+ x\Delta_- y + \Delta_- x\Delta_+ y) = |W_{xy}|. \tag{7.24}$$

The solution of the six equations (7.17)-(7.22) reads

$$\begin{aligned} p_o =& \frac{\Delta_+ x\Delta_- x\Delta_+ y\Delta_- y - \Delta_+ x\Delta_- xW_y - \Delta_+ y\Delta_- yW_x}{\Delta_+ x\Delta_- x\Delta_+ y\Delta_- y} \\ &+ \frac{E_y(\Delta_+ y - \Delta_- y)}{\Delta_+ y\Delta_- y} + \frac{E_x(\Delta_+ x - \Delta_- x)}{\Delta_+ x\Delta_- x} + \frac{2|W_{xy}|}{\Delta_c}, \end{aligned} \tag{7.25}$$
$$p_{x+} = \frac{W_x + E_x\Delta_- x}{(\Delta_+ x)^2 + \Delta_+ x\Delta_- x} - \frac{|W_{xy}|}{\Delta_c}, \tag{7.26}$$
$$p_{x-} = \frac{W_x - E_x\Delta_+ x}{(\Delta_- x)^2 + \Delta_+ x\Delta_- x} - \frac{|W_{xy}|}{\Delta_c}, \tag{7.27}$$
$$p_{y+} = \frac{W_y + E_y\Delta_- y}{(\Delta_+ y)^2 + \Delta_+ y\Delta_- y} - \frac{|W_{xy}|}{\Delta_c}, \tag{7.28}$$
$$p_{y-} = \frac{W_y - E_y\Delta_+ y}{(\Delta_- y)^2 + \Delta_+ y\Delta_- y} - \frac{|W_{xy}|}{\Delta_c}, \tag{7.29}$$
$$p_{xy} = \frac{|W_{xy}|}{\Delta_c}, \tag{7.30}$$

where $\Delta_c = \Delta_+ x\Delta_+ y + \Delta_- x\Delta_- y$ for non-negative $(\tilde{\sigma}_{xy}(x_i, y_j, t_k, \theta)\Delta t + E_x E_y)$ and $\Delta_c = \Delta_+ x\Delta_- y + \Delta_- x\Delta_+ y$ for negative $(\tilde{\sigma}_{xy}(x_i, y_j, t_k, \theta)\Delta t + E_x E_y)$.

Following the construction of x_+, x_-, y_+, y_-, it is easy to check that p_o is always non-negative. The same holds for p_{xy}. To ensure also the non-negativity of p_{x+}, p_{x-}, p_{y+} and p_{y-} we have to properly choose the variables Var_x, Var_y, and Cov to get non-negative weights, while still remaining consistent with the original problem as $h \to 0$. This is done in the next section.

7.2.4 Artificial diffusion and covariance adjustment

Let us assume $E_x \geq 0$. Now, the first fraction of p_{x+} is positive, however the first fraction of p_{x-} may be negative. We will set Var_x (and hence W_x) in such way, that it will be also positive. It holds

$$W_x - E_x\Delta_+ x > Var_x + E_x^2 - E_x(\sqrt{2b(Var_x + E_x^2)} + 2\Delta x). \tag{7.31}$$

The right-hand side of the inequality (7.31) is 0 for $Var_x = A_x$ and greater than 0 for $Var_x > A_x$ with

$$A_x = 1/2 \left(|E_x| \sqrt{4b^2 E_x^2 + 16b\Delta x |E_x|} - (2b-2)E_x^2 + 4\Delta x |E_x| \right). \tag{7.32}$$

We replaced here E_x with $|E_x|$ to cover also the analogous case $E_x < 0$ and possibly negative p_{x+}. Now, if we set

$$Var_x = \max(\sigma_x^2(x_i, y_j, t_k, \theta)\Delta t, A_x, E_x^2), \tag{7.33}$$

$$Var_y = \max(\sigma_y^2(x_i, y_j, t_k, \theta)\Delta t, A_y, E_y^2), \tag{7.34}$$

where A_y is defined analogously, the first fraction in p_{x+}, p_{x-}, p_{y+} and p_{y-} will be non-negative. The possible difference between Var_x resp. Var_y and the variances from the original problem $\sigma_x^2 \Delta t$ resp. $\sigma_y^2 \Delta t$ is called artificial diffusion. Now, taking into account the correlation coefficient in the current node, following these definitions of variances Var_x, Var_y we define also new covariance

$$\tilde{\sigma}_{xy} = \rho \frac{\sqrt{Var_x Var_y}}{\Delta t} \tag{7.35}$$

and

$$\begin{aligned} C_{xy} := \min \Big(& \frac{W_x + E_x \Delta_- x}{(\Delta_+ x)^2 + \Delta_+ x \Delta_- x} \Delta_c, \frac{W_x - E_x \Delta_+ x}{(\Delta_- x)^2 + \Delta_+ x \Delta_- x} \Delta_c, \\ & \frac{W_y + E_y \Delta_- y}{(\Delta_+ y)^2 + \Delta_+ y \Delta_- y} \Delta_c, \frac{W_y - E_y \Delta_+ y}{(\Delta_- y)^2 + \Delta_+ y \Delta_- y} \Delta_c, \\ & |\tilde{\sigma}_{xy}(x_i, y_j, t_k, \theta)\Delta t + E_x E_y| \Big), \end{aligned} \tag{7.36}$$

Now, we define Cov as

$$Cov = \begin{cases} C_{xy} - E_x E_y & \text{if } (\tilde{\sigma}_{xy}(x_i, y_j, t_k, \theta)\Delta t + E_x E_y) \geq 0, \\ -C_{xy} - E_x E_y & \text{if } (\tilde{\sigma}_{xy}(x_i, y_j, t_k, \theta)\Delta t + E_x E_y) < 0. \end{cases} \tag{7.37}$$

This covariance Cov is consistent with the variances Var_x, Var_y defined by (7.33), (7.34) as for $\tilde{\sigma}_{xy}(x_i, y_j, t_k, \theta)\Delta t + E_x E_y \geq 0$ it holds

$$-\sqrt{Var_x Var_y} \leq -E_x E_y \leq Cov \leq \tilde{\sigma}_{xy}(x_i, y_j, t_k, \theta)\Delta t \leq \sqrt{Var_x Var_y}, \tag{7.38}$$

and for $\tilde{\sigma}_{xy}(x_i, y_j, t_k, \theta)\Delta t + E_x E_y < 0$ it holds

$$-\sqrt{Var_x Var_y} \leq \tilde{\sigma}_{xy}(x_i, y_j, t_k, \theta)\Delta t \leq Cov \leq -E_x E_y \leq \sqrt{Var_x Var_y}. \tag{7.39}$$

Now it also holds $|W_{xy}| = C_{xy}$, what implies that p_{x+}, p_{x-}, p_{y+} and p_{y-} are all positive. It is easy to check that it holds

$$\frac{Var_x}{\Delta t} = \sigma_x^2(x_i, y_j, t_k, \theta) + \mathcal{O}(h), \quad \frac{Var_y}{\Delta t} = \sigma_y^2(x_i, y_j, t_k, \theta) + \mathcal{O}(h). \tag{7.40}$$

and

$$\frac{Var_x}{\Delta t} = \sigma_x^2(x_i, y_j, t_k, \theta), \quad \text{resp.} \quad \frac{Var_y}{\Delta t} = \sigma_y^2(x_i, y_j, t_k, \theta) \tag{7.41}$$

for $\sigma_x^2(x_i, y_j, t_k, \theta) \neq 0$ resp. $\sigma_y^2(x_i, y_j, t_k, \theta) \neq 0$ and h small enough. It holds

$$W_x + E_x\Delta_- x = \left(\sigma_x^2 + \mathcal{O}\left(\sqrt{h}\right)\right)\Delta t, \tag{7.42}$$

$$(\Delta_+ x)^2 + \Delta_+ x\Delta_- x = 4h\sigma_x^2 + \mathcal{O}(h^{3/2}), \tag{7.43}$$

$$\Delta_c = 4h\sigma_x\sigma_y + \mathcal{O}(h^{3/2}). \tag{7.44}$$

It follows that

$$\frac{W_x + E_x\Delta_- x}{(\Delta_+ x)^2 + \Delta_+ x\Delta_- x}\Delta_c = \left(\sigma_x\sigma_y + \mathcal{O}\left(\sqrt{h}\right)\right)\Delta t \tag{7.45}$$

and the same holds also for the second, third and fourth maximum candidate in (7.36). Moreover,

$$|\tilde{\sigma}_{xy}\Delta t + E_x E_y| = |\tilde{\sigma}_{xy}|\Delta t + \mathcal{O}((\Delta t)^2) = \left(|\sigma_{xy}| + \mathcal{O}\left(\sqrt{h}\right)\right)\Delta t. \tag{7.46}$$

Therefore,

$$C_{xy} = \left(|\sigma_{xy}| + \mathcal{O}\left(\sqrt{h}\right)\right)\Delta t, \quad \Rightarrow \quad \frac{Cov}{\Delta t} = \sigma_{xy} + \mathcal{O}\left(\sqrt{h}\right) \tag{7.47}$$

and it is easy to check that

$$\frac{Cov}{\Delta t} = \sigma_{xy} \tag{7.48}$$

for $\sigma_x^2 \neq 0$, $\sigma_y^2 \neq 0$, $|\sigma_{xy}| \neq \sigma_x\sigma_y$ and h small enough. Following (7.40),(7.47), the modified variances and the modified covariance are consistent with the variances and the covariance from the original problem. Moreover, for $\sigma_x, \sigma_y \neq 0$ and $|\sigma_{xy}| < \sigma_x\sigma_y$, the modified variances and the covariance will be equal to the original ones for h small enough.

7.2.5 Setting parameter K and stencil size reduction

In the formula for $h = \max(K\max(\Delta x, \Delta y), \Delta t)$, the part $K\max(\Delta x, \Delta y)$ is responsible for the consistency of the correlation in the numerical model with the correlation of the original problem. Here, the parameter $K > 0$ can be chosen arbitrarily, however for a large (relatively to problem parameters) K, the stencil is large, what typically increases the error. On the other hand, for too small K, the correlation may start being exact (or exact enough) only for very fine grids, while not being sufficiently exact on the coarse grids, resulting in larger errors on these coarse grids. For $K = 0$ we can't guarantee the convergence of the correlation. However, the correlation in the numerical model may match with the correlation from the original problem even for small K or $K = 0$. This motivates the following **multiple K modification** of the Tree-Grid method;
For each control in each node:

1. set $l = 1$ and use $K = K_0$, $K_0 \geq 0$ to compute x_*, y_*, p_*.
2. compute the correlation of the numerical model: $\tilde{\rho} = Cov/\sqrt{Var_x Var_y}$.
3. **if** $\tilde{\rho} \neq \rho$: recompute x_*, y_*, p_* with $K = K_l$, $K_l > K_{l-1}$ and set l:=l+1
   else break.

4. **if** $l < l_{max}$: go to step 2,
   else break.

Using this modification, we will use smaller stencil size as much as possible. This approach can be seen as some analogy to the approach in paper [35] where fixed (and thus small) stencil is used as much as possible. However, here we will not increase the convergence rate, but possibly reduce the error.

Another approach, **the non-constant K modification**, is to use non-constant $K = K(x, y, \theta)$. This can be smaller in nodes with large volatilities σ_x, σ_y and larger in nodes with smaller σ_x, σ_y, regulating the stencil size to not explode in case of large volatilities.

Both modifications can be also combined and it is easy to check, that they do not harm the consistency. In our numerical simulation, these modifications didn't lead to significant improvement, therefore we used just a constant $K = 1/400$. However, they may be useful for other stochastic control problems.

7.2.6 The final 2D Tree-Grid method algorithm

Finally, we can use the stencil nodes and weights to construct the 2D Tree-Grid algorithm.

We define the function v^{k+1}:

$$\begin{aligned} &\text{If } (x,y) \in \{x_1, x_2, \dots, x_{N_x}\} \times \{y_1, y_2, \dots, y_{N_y}\} : \\ &\qquad v^{k+1}(x,y) = v_{i,j}^{k+1} \text{ so that } (x,y) = (x_i, y_j), \\ &\text{else: } v^{k+1}(x,y) = BC(x,y,t_{k+1}). \end{aligned} \tag{7.49}$$

For a given space-node (x_i, y_j) and a given control θ we define

$$v_o^{k+1} = v^{k+1}(x_i, y_j), \quad v_{x+}^{k+1} = v^{k+1}(x_+, y_j), \quad v_{x-}^{k+1} = v^{k+1}(x_-, y_j), \tag{7.50}$$

$$v_{y+}^{k+1} = v^{k+1}(x_i, y_+), \quad v_{y-}^{k+1} = v^{k+1}(x_i, y_-), \tag{7.51}$$

$$\text{If } W_{xy} \geq 0 : \quad v_{xy+}^{k+1} = v^{k+1}(x_+, y_+), \quad v_{xy-}^{k+1} = v^{k+1}(x_-, y_-). \tag{7.52}$$

$$\text{If } W_{xy} < 0 : \quad v_{xy+}^{k+1} = v^{k+1}(x_+, y_-), \quad v_{xy-}^{k+1} = v^{k+1}(x_-, y_+). \tag{7.53}$$

$$f_{i,j}^k(\theta) = f(x_i, y_j, t_k, \theta), \quad r_{i,j}^k(\theta) = r(x_i, y_j, t_k, \theta). \tag{7.54}$$

Now, the discretized version of the dynamic programming equation for $i = 2, 3, \dots, N_x - 1$, $j = 2, 3, \dots, N_y - 1$, $k = 1, 2, \dots, M - 1$ reads

$$v_{i,j}^k = \max_{\theta \in \Theta} w_{i,j}^k(\theta) \tag{7.55}$$

$$\begin{aligned} w_{i,j}^k(\theta) = &f_{i,j}^k(\theta)\Delta t + (1 + r_{i,j}^k(\theta)\Delta t) \\ &\cdot \Big(p_{x+} v_{x+}^{k+1} + p_{x-} v_{x-}^{k+1} + p_{y+} v_{y+}^{k+1} + p_{y-} v_{y-}^{k+1} \\ &+ p_o v_o^{k+1} + p_{xy}(v_{xy+}^{k+1} + v_{xy-}^{k+1}) \Big). \end{aligned} \tag{7.56}$$

For boundary nodes (x_i, y_j) ($i \in \{1, N_x\}$ or $j \in \{1, N_y\}$) we employ the boundary condi-

tion:

$$v_{i,j}^k = BC(x_i, y_j, t_k). \tag{7.57}$$

The terminal condition is defined as

$$v_{i,j}^M = V_T(x_i, y_j). \tag{7.58}$$

Finally we can summarize the whole algorithm of the 2D Tree-Grid method:

Algorithm 7 The 2D Tree-Grid method

Set $v_{i,j}^M = V_T(x_i, y_j)$ for $i = 1, 2, \ldots, N_x$, $j = 1, 2, \ldots, N_y$;
for $k = M-1, M-2, \ldots, 1$ **do**
  for $i = 1, 2, 3, \ldots, N_x$ **do**
    for $j = 1, 2, 3, \ldots, N_y$ **do**
      if $i \in \{1, N_x\}$ or $j \in \{1, N_y\}$ **then**
        Compute $v_{i,j}^k$ according to (7.57);
      else
        for $\theta \in \Theta$ **do**
          Determine E_x, E_y according to Section 7.2.1;
          Compute A_x, A_y according to (7.32);
          Compute Var_x, Var_y according to (7.33), (7.34);
          Determine W_x, W_y according to Section 7.2.1;
          Determine x_+, x_-, y_+, y_- according to (7.9)-(7.14);
          Determine $\Delta_+ x$, $\Delta_- x$, $\Delta_+ y$, $\Delta_- y$ according to (7.15)-(7.16);
          Compute $\tilde{\sigma}_{xy}$ according to (7.35);
          Compute C_{xy} according to (7.36);
          Compute Cov according to (7.37);
          Determine W_{xy} according to Section 7.2.1;
          Compute $p_o, p_{x+}, p_{x-}, p_{y+}, p_{y-}, p_{xy}$ according to (7.25)-(7.30);
          Using (7.49), compute all variables defined by (7.50)-(7.54);
          Compute $w_{i,j}^k(\theta)$ according to (7.56);
        end for
        Compute $v_{i,j}^k$ according to (7.55);
      end if
    end for
  end for
end for

7.2.7 Comparison to other wide stencil methods

As the stencil size gets wider with Δx, Δy, the 2D Tree-Grid method can be classified as an explicit wide stencil method. Wide stencil methods are presented for example in [31] or in the papers [8, 12, 13]. A similar but implicit method leading to wide stencil only in some nodes can be found in [35]. Let us therefore state the two most important advantages of the 2D Tree-Grid method:

1. Unconditional stability. In contrast to other explicit wide stencil method, the 2D Tree-Grid method is unconditionally stable. The unconditional stability is achieved

by making the stencil size dependent not only on the space steps but also on the time step. The method from [35] is also unconditionally stable, however this property is achieved by making the method implicit in that case.

2. No interpolation is needed. In order to get a 7-point wide stencil without interpolation, we possibly need to artificially increase the variance and decrease the covariance of the system (see Section 7.2.4). However, the approximation is still consistent as this artificial increase and decrease vanishes with grid refinement and in standard cases even reaches zero at some point. We should note, that even in standard wide stencil schemes [12], the variance is increased and covariance altered due to the use of interpolation. Moreover in that case, this behavior is present for any grid refinement, in contrast to the case of the Tree-Grid method presented in this work. The comparison of the stencil in 2D Tree-Grid scheme and of the standard wide stencil scheme is illustrated in Figure 7.2.

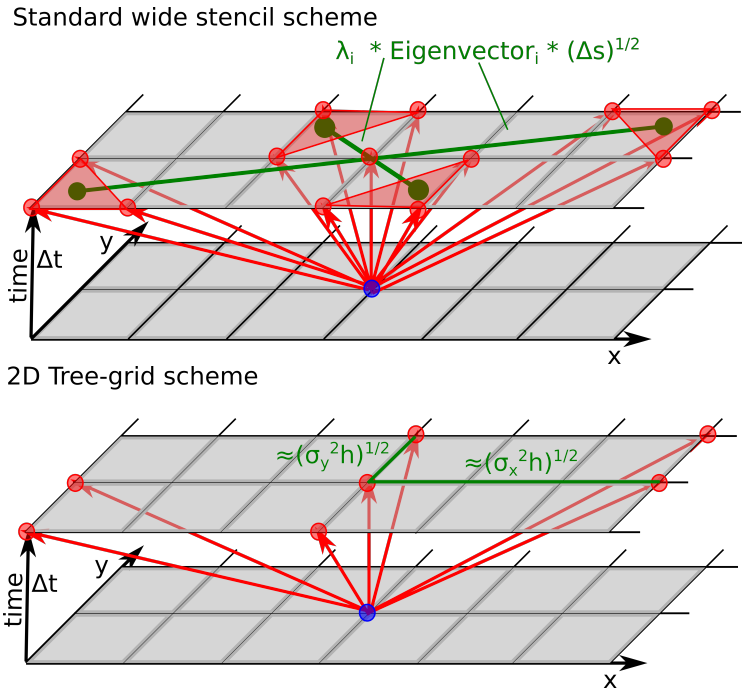

Figure 7.2: Illustration of the stencil of the standard wide-stencil scheme compared to the stencil of the 2D Tree-Grid scheme. In the case of standard wide stencil scheme, value in the green points is computed by interpolation using the red grid-nodes. In this example 13 grid-node values are needed in the computation of the value in the blue node (the upper figure). On the other hand, we only need 7 (red) nodes and no interpolation to compute the value of the blue node using the 2D Tree-Grid method (the bottom figure)

Important advantage of the standard wide-stencil schemes is the possibility of generalization to dimensions higher that two. As we will see in the next chapter, the Tree-Grid method cannot be generalized to higher dimensions for an arbitrary problem setting.

7.3 Convergence of the 2D Tree-Grid method

In this section, we will prove the convergence of the 2D Tree-Grid method. In the first part, we will quickly summarize the convergence theory of Barles and Souganidis [4] and in the second part of this section we will use this theory to prove the convergence of our scheme. Let us note that the Algorithm 7 was derived by discretizing the dynamics in the original SCP (7.1)-(7.4), but we will prove that the scheme is consistent with the HJB equation (7.6)-(7.8).

Now, we will prove the convergence of the 2D Tree-Grid method. For the purpose of this convergence analysis we will rewrite equations (7.55),(7.56) as

$$\begin{aligned} Gv(x_i, y_j, t_k) =& G(v_o^{k+1}, v_{x+}^{k+1}, v_{x-}^{k+1}, v_{y+}^{k+1}, v_{y-}^{k+1}, v_{xy+}^{k+1}, v_{xy-}^{k+1}) \\ =& \frac{1}{\Delta t}\Big(v_{i,j}^k - \max_{\theta\in\Theta}\Big(f_{i,j}^k(\theta)\Delta t + \Big(1 + r_{i,j}^k(\theta)\Delta t\Big) \\ & \cdot \Big(p_{x+}v_{x+}^{k+1} + p_{x-}v_{x-}^{k+1} + p_{y+}v_{y+}^{k+1} + p_{y-}v_{y-}^{k+1} \\ & + p_o v_o^{k+1} + p_{xy}(v_{xy+}^{k+1} + v_{xy-}^{k+1})\Big)\Big)\Big) = 0. \end{aligned} \tag{7.59}$$

Using the theory from Section 2.2, our goal is to show that the equation (7.59) is a monotone, consistent, and stable approximation of the nonlinear differential operator F defined by the PDE (7.6):

$$FV(x, y, t) = -\frac{\partial V}{\partial t} - \max_{\theta\in\Theta}\Big(LV + r(\cdot)V + f(\cdot)\Big). \tag{7.60}$$

Let us define the difference operators

$$\Delta_{o+}z = \Delta_+ z, \quad \Delta_{o-}z = -\Delta_- z, \tag{7.61}$$

for $z \in \{x, y\}$. At first, we will show the consistency of the scheme in an arbitrary point (x_i, y_j, t_k):

Lemma 10 (Consistency)**.** *The discrete scheme* (7.59) *is consistent with the PDE operator* (7.60)*.*

Proof. Let $\phi : \mathbb{R}^2 \times [0, T] \to \mathbb{R}$ be a C^∞-smooth function. Let us define $\phi^k(x, y) = \phi(x, y, t_k)$ and use short notation defined by (7.50)-(7.54) for ϕ^{k+1} instead of v^{k+1}. At first, let us sketch the main idea of the proof of consistency in an arbitrary point (x_i, y_j, t_k): We will write all values ϕ_α^{k+1} for $\alpha \in \{x+, x-, y+, y-, xy+, xy-\}$ as Taylor expansions around $\phi_{i,j}^k$. We will substitute these Taylor expansions into the discrete scheme (7.59), group terms with the same derivatives together and estimate the sum of coefficients in front of them. We will end up with the PDE operator (7.60) and some terms of order $\mathcal{O}(h^\lambda)$, where $\lambda = 1/2$ if $|\rho| = 1$ or $\sigma_x = 0$ or $\sigma_y = 0$ and $\lambda = 1$ else. Let us suppose that $W_{xy} \geq 0$ in (x_i, y_j, t_k). The case of negative W_{xy} can be treated analogously. For

$* \in \{+,-\}$, let us define the operators:

$$\begin{aligned} A_{xy*}\phi =& \frac{\partial \phi}{\partial x}\Delta_{o*}x + \frac{\partial \phi}{\partial y}\Delta_{o*}y + \frac{1}{2}\frac{\partial^2 \phi}{\partial x^2}(\Delta_{o*}x)^2 + \frac{\partial^2 \phi}{\partial x \partial y}\Delta_{o*}x\Delta_{o*}y \\ &+ \frac{1}{2}\frac{\partial^2 \phi}{\partial y^2}(\Delta_{o*}y)^2, \end{aligned} \tag{7.62}$$

$$\begin{aligned} B_{xy*}\phi =& \frac{1}{6}\frac{\partial^3 \phi}{\partial x^3}(\Delta_{o*}x)^3 + \frac{1}{2}\frac{\partial^3 \phi}{\partial x^2 \partial y}(\Delta_{o*}x)^2\Delta_{o*}y + \frac{1}{2}\frac{\partial^3 \phi}{\partial x \partial y^2}\Delta_{o*}x(\Delta_{o*}y)^2 \\ &+ \frac{1}{6}\frac{\partial^3 \phi}{\partial y^3}(\Delta_{o*}y)^3, \end{aligned} \tag{7.63}$$

$$\begin{aligned} C_{xy*}\phi =& \frac{1}{24}\frac{\partial^4 \phi}{\partial x^4}(\Delta_{o*}x)^4 + \frac{1}{6}\frac{\partial^4 \phi}{\partial x^3 \partial y}(\Delta_{o*}x)^3\Delta_{o*}y + \frac{1}{4}\frac{\partial^4 \phi}{\partial x^2 \partial y^2}(\Delta_{o*}x)^2, (\Delta_{o*}y)^2 \\ &+ \frac{1}{6}\frac{\partial^4 \phi}{\partial x \partial y^3}\Delta_{o*}x(\Delta_{o*}y)^3 + \frac{1}{24}\frac{\partial^4 \phi}{\partial y^4}(\Delta_{o*}y)^4. \end{aligned} \tag{7.64}$$

Now we can expand ϕ_{xy*}^{k+1} around $\phi_{i,j}^k$ as follows:

$$\begin{aligned} \phi_{xy*}^{k+1} =& \phi_{i,j}^k + \frac{\partial}{\partial t}\phi_{i,j}^k \Delta t + A_{xy*}\phi_{i,j}^k + \frac{\partial}{\partial t}\left(A_{xy*}\phi_{i,j}^k\right)\Delta t + B_{xy*}\phi_{i,j}^k \\ &+ \frac{\partial}{\partial t}\left(B_{xy*}R_{xy*}\right)\Delta t + C_{xy*}R_{xy*} + \mathcal{O}((\Delta t)^2), \end{aligned} \tag{7.65}$$

where

$$R_{xy*} = \phi(x_i + \epsilon_{xy*}\Delta_{o*}x, y_j + \delta_{xy*}\Delta_{o*}y, t_k + \gamma_{xy*}\Delta t), \tag{7.66}$$

for some $\epsilon_{xy*}, \delta_{xy*}, \gamma_{xy*} \in [0,1]$. We expand $\phi_{x*}^{k+1}, \phi_{y*}^{k+1}$, $* \in \{+,-\}$ in the same manner: for ϕ_{x*}^{k+1} we only need to substitute $\Delta_{o*}y$ with 0 and for ϕ_{y*}^{k+1} we only need to substitute $\Delta_{o*}x$ with 0 in the Taylor expansion (7.65) and change the index $xy*$ to $x*$ resp. $y*$ in all expressions (7.62)-(7.66). Now, by substituting the Taylor expansions into the scheme $G\phi_{i,j}^k$ defined by (7.59) we get:

$$\begin{aligned} G\phi_{i,j}^k =& \frac{1}{\Delta t}\Big(\phi_{i,j}^k - \max_{\theta\in\Theta}\Big(f_{i,j}^k(\theta)\Delta t + \Big(1 + r_{i,j}^k(\theta)\Delta t\Big) \\ &\cdot \Big(\phi_{i,j}^k + \frac{\partial \phi_{i,j}^k}{\partial t}\Delta t + (L\phi_{i,j}^k + \mathcal{O}(h^\lambda))\Delta t\Big)\Big)\Big) \\ =& F\phi_{i,j}^k + \mathcal{O}(h^\lambda), \end{aligned} \tag{7.67}$$

where $\lambda = 1/2$ if $|\rho| = 1$ or $\sigma_x = 0$ or $\sigma_y = 0$ and $\lambda = 1$ else. In the first equation of (7.67) we used the estimates of the linear combinations of higher order terms from the Taylor expansions. The coefficients of these linear combinations are the probabilities $p_o, p_{x+}, p_{x-}, p_{y+}, p_{y-}, p_{xy+}, p_{xy-}$ (according to the definition of G (7.59)). We describe here these estimates:

1. For the linear combinations of the terms included in $A_\alpha\phi_{i,j}^k$, $\alpha \in \{x+, x-, y+, y-, xy+, xy-\}$ we used the properties of the scheme (7.17)-(7.22). After summing all expressions obtained by applying (7.17)-(7.22) we get $\left(L\phi_{i,j}^k + \mathcal{O}(h^{1/2})\right)\Delta t$. Moreover, following Section 7.2, if $|\rho| \neq 1, \sigma_x \neq 0, \sigma_y \neq 0$, for h small enough we end up just with $L\phi_{i,j}^k \Delta t$.

2. In the same way, for the linear combinations of the terms included in $\frac{\partial}{\partial t}\left(A_\alpha \phi_{i,j}^k\right)\Delta t$ we get $\left(L\left(\partial\phi_{i,j}^k/\partial t\right)+\mathcal{O}(h^\lambda)\right)(\Delta t)^2=\mathcal{O}(h)\Delta t$.

3. For the linear combinations of the terms included in $B_\alpha\phi_{i,j}^k$ we used the following estimates:

$$\begin{aligned}&(p_{x+}+p_{xy+})(\Delta_{o+}x)^3+(p_{x-}+p_{xy+})(\Delta_{o-}x)^3\\&=W_x(\Delta_{o+}x+\Delta_{o-}x)-E_x\Delta_{o+}x\Delta_{o-}x=\mathcal{O}(h)\Delta t,\end{aligned}\tag{7.68}$$

$$\begin{aligned}&p_{xy+}(\Delta_{o+}x)^2\Delta_{o+}y+p_{xy-}(\Delta_{o-}x)^2\Delta_{o-}y=\\&\frac{W_{xy}\left((\Delta_{o+}x)^2\Delta_{o+}y+(\Delta_{o-}x)^2\Delta_{o-}y\right)}{\Delta_{o+}x\Delta_{o+}y+\Delta_{o-}x\Delta_{o-}y}=\mathcal{O}(h)\Delta t.\end{aligned}\tag{7.69}$$

   Here we used that $\Delta_{o+}x+\Delta_{o-}x=\mathcal{O}(h)$, $(\Delta_{o+}x)^2\Delta_{o+}y+(\Delta_{o-}x)^2\Delta_{o-}y=\mathcal{O}(h^2)$. We used analogous estimates also for the terms where x and y are switched symmetrically.

4. As $(\Delta_{o*}x)^2=\mathcal{O}(h)$, $(\Delta_{o*}y)^2=\mathcal{O}(h)$ it is clear that all terms in $\frac{\partial}{\partial t}\left(B_\alpha R_\alpha\right)\Delta t$ are of order $\mathcal{O}(h)\Delta t$.

5. For the linear combinations of the terms included in $C_\alpha R_\alpha$ we constructed the estimate in the following way: Let us define

$$\psi=(p_{x+}a_{x+}+p_{xy+}a_{xy+})(\Delta_{o+}x)^2+(p_{x-}a_{x-}+p_{xy-}a_{xy-})(\Delta_{o-}x)^2.$$

   Then, it holds:

$$\begin{aligned}&m\left(\sigma_x^2+\mathcal{O}(h)\right)\Delta t\le\psi\le M\left(\sigma_x^2+\mathcal{O}(h)\right)\Delta t,\\&m=\min\left(a_{x+},a_{x-},a_{xy+},a_{xy-}\right),\quad M=\max\left(a_{x+},a_{x-},a_{xy+},a_{xy-}\right).\end{aligned}$$

   If $m=\mathcal{O}(h)$, $M=\mathcal{O}(h)$ then $\psi=\mathcal{O}(h)\Delta t$. Now for $(a_{x+},a_{x-},a_{xy+},a_{xy-})$ equal to

   - $\frac{1}{24}\frac{\partial^4}{\partial x^4}\left((\Delta_{o+}x)^2R_{x+},(\Delta_{o-}x)^2R_{x-},(\Delta_{o+}x)^2R_{xy+},(\Delta_{o-}x)^2R_{xy-}\right)$,
   - $\frac{1}{6}\frac{\partial^4}{\partial x^3\partial y}\left(0,0,\Delta_{o+}x\Delta_{o+}yR_{xy+},\Delta_{o-}x\Delta_{o-}yR_{xy-}\right)$,
   - $\frac{1}{4}\frac{\partial^4}{\partial x^3\partial y}\left(0,0,(\Delta_{o+}y)^2R_{xy+},(\Delta_{o-}y)^2R_{xy-}\right)$,

   which are all $\mathcal{O}(h)$, we get that the linear combinations of the first three terms in $C_\alpha R_\alpha$ are of order $\mathcal{O}(h)\Delta t$. Using analogous estimates, also the linear combinations of the fourth and fifth term are of order $\mathcal{O}(h)\Delta t$.

Using the estimates above, we proved (7.67) which is the first order consistency of our scheme if $|\rho|<1$, $\sigma_x>0$, $\sigma_y>0$ and consistency of order $1/2$ else. □

Now the lemma establishing monotonicity of the scheme (7.59) follows:

Lemma 11 (Monotonicity)**.** *If $1+r_{i,j}^k(\theta)\Delta t\ge 0$ for all possible i,j,k,θ, then the discrete scheme* (7.59) *is monotone.*

Proof. Monotonicity is in this case a direct implication of the non-negativity of p_o, p_{x+},

p_{x-}, p_{y+}, p_{y-}, p_{xy+}, p_{xy-}. □

Remark 10. *As already mentioned in Chapter 5, even if* $1 + r^k_{i,j}(\theta)\Delta t < 0$ *for some* i, j, k, θ*, we can get a monotone scheme if we substitute* $1 + r^k_{i,j}(\theta)\Delta t$ *by* $1/(1 - r^k_{i,j}(\theta)\Delta t)$ *in* (7.59) *for these parameters* i, j, k, θ*. Note that this change does not harm the consistency, nor the stability of the scheme.*

The stability analysis of the 2D Tree-Grid method is basically identical to the stability analysis of the one-dimensional Tree-Grid method done in Chapter 5. Therefore we state here just the stability condition and the Lemma about stability of the scheme.

Property 2 (Stability condition of the problem)**.** *We suppose that:*

1. *There exist constants* C_f, C_r *such that for all* $x, y, t, \theta \in [x_1, x_{N_x}] \times [y_1, y_{N_y}] \times [t_1, t_M] \times \Theta$ *holds:* $|f(x, y, t, \theta)| < C_f$, $|r(x, y, t, \theta)| < C_r$.
2. *There exist constant* C_{BC} *such that* $|BC(x, y, t)| < C_{BC}$ *holds for all* $t \in [t_1, t_M]$ *and for all possible outer-domain values* (x, y) *of the variables* $(x_+, y_j), (x_-, y_j), (x_i, y_+), (x_i, y_-), (x_+, y_+), (x_+, y_-), (x_-, y_+), (x_-, y_-)$ *for any grid.*

The lemma about stability of the scheme follows:

Lemma 12 (Stability)**.** *If the problem satisfies the stability conditions defined in Property 2, then the scheme* (7.59) *is stable.*

Proof. The proof of this lemma can be found in Chapter 5. □

Finally, the convergence theorem follows:

Theorem 13 (Convergence of the 2D Tree-Grid method)**.** *The approximation computed with the 2D Tree-Grid method (defined by Algorithm 7) for solving the SCP* (7.1)-(7.4) *and the corresponding HJB equation* (7.6)-(7.8) *satisfying the strong uniqueness property (see [4]) and the stability conditions defined in Property 2 converges to the viscosity solution of this SCP (and the HJB equation).*

Proof. The proof follows from the Theorem 1 and Lemmas 10, 11, 12. □

7.4 Numerical example: two-factor uncertain volatility model

In this section, we will use the 2D Tree-Grid method for pricing options on two different risky assets under uncertain volatility:

Example 4 (Two-factor uncertain volatility model)**.** *In this setting, the volatilities and the correlation of the assets are only known to lie in certain bounds. The maximal option*

price can be in this case computed as solution of the HJB equation

$$\frac{\partial V}{\partial t} + \max_{\theta \in \Theta} (LV - rV) = 0, \tag{7.70}$$

$$LV = \frac{\sigma_x^2}{2} x^2 \frac{\partial^2 V}{\partial x^2} + \rho \sigma_x \sigma_y xy \frac{\partial^2 V}{\partial x \partial y} + \frac{\sigma_y^2}{2} y^2 \frac{\partial^2 V}{\partial y^2} + rx \frac{\partial V}{\partial x} + ry \frac{\partial V}{\partial y}, \tag{7.71}$$

$$V(x, y, T) = V_T(x, y), \tag{7.72}$$

$$0 < t < T, \quad x, y \in \mathbb{R}.$$

Here, x, y, t denote the first, the second asset price and the time, r denotes the risk-free interest rate, T is maturity time, the terminal condition $V_T(x, y)$ is payoff function and for the control variable holds $\theta = (\sigma_x, \sigma_y, \rho)$, where σ_x, σ_y, ρ are uncertain volatilities of the first and the second asset, and their correlation. For the set of possible controls holds:

$$\Theta = [\sigma_{x,min}, \sigma_{x,max}] \times [\sigma_{y,min}, \sigma_{y,max}] \times [\rho_{min}, \rho_{max}]. \tag{7.73}$$

To obtain the minimal option price, we have to replace max *by* min *in HJB equation* (7.70)*. This model was discussed for example in [40] and later solved with a hybrid (combining fixed and wide stencils) implicit method in [35]. As explained in [35], the optimal control lies in a subset of Θ,*

$$\tilde{\Theta} = \Big((\{\sigma_{x,min}, \sigma_{x,max}\} \times [\sigma_{y,min}, \sigma_{y,max}]) \\ \cup ([\sigma_{x,min}, \sigma_{x,max}] \times \{\sigma_{y,min}, \sigma_{y,max}\}) \Big) \times \{\rho_{min}, \rho_{max}\}. \tag{7.74}$$

Therefore, we will search for the control in $\bar{\Theta}$, a discretized version of $\tilde{\Theta}$.

To verify the implementation, we will also solve a problem with an one-element set $\bar{\Theta} = \{(\sigma_x, \sigma_y, \rho)\}$. In this case, the equation (7.70) is simply the 2-dimensional Black-Scholes equation [7] for which the analytical solution is known.

Terminal condition: As a terminal condition, we use the payoff function in the form of a butterfly spread on the maximum of two assets:

$$V_T(x, y) = (M - K_1)^+ - 2\,(M - (K_1 + K_2)/2)^+ + (M - K_2)^+ \\ \text{with } M = \max(x, y), \quad K_1 = 34, \quad K_2 = 46.$$

Boundary conditions: We will use Dirichlet boundary conditions. On the upper and the right boundary, we will set the value to 0:

$$BC(x > x_{max}, y, t) = 0, \quad BC(x, y > y_{max}, t) = 0, \quad x_{max} = y_{max} = 144$$

To verify that our computational domain is large enough, we solved the HJB and the Black-Scholes equation also for $x_{max} = y_{max} = 400$ and obtained in node $(x, y, t) = (40, 40, 0)$ the same values as for $x_{max} = y_{max} = 144$. On the lower ($y = 0$) and left ($x = 0$) boundary, the equation (7.70) degenerates to a HJB equation for one-dimensional uncertain volatility model from [17]. We solve it with the one-dimensional Tree-Grid method, from chapter Chapter 5 with artificial diffusion defined as in Chapter 6. For the case that the values from outside of the computational domain $[0, x_{max}] \times [0, y_{max}]$ are needed, we artificially define

the solution for $x < 0, y < 0$ to have the same values as the solution on the boundary:

$$BC(x < 0, y) = BC(0, y), \quad BC(x, y < 0) = BC(x, 0), \quad BC(x < 0, y < 0) = 0.$$

Numerical results: Here we present the experimental convergence results of the 2D Tree-Grid method applied to the Black-Scholes model and the uncertain volatility model. We implemented our method in Python [1] and tested on Intel Core i7-4770 CPU 3.40GHz computer with 16 GB RAM. We performed the simulations on four different sets of parameters:

- Black-Scholes model with moderate volatility and correlation: $\sigma_x = 0.3, \sigma_y = 0.5, \rho = 0.4$. Results of the simulation are in Table 7.1.
- Black-Scholes model with extreme parameters: $\sigma_x = 0.05, \sigma_y = 0.05, \rho = -0.95$. Results of the simulation are in Table 7.2.
- Uncertain volatility model, maximal option price (worst case scenario) with parameters $\sigma_{x,min} = \sigma_{y,min} = \rho_{min} = 0.3$, $\sigma_{x,max} = \sigma_{y,max} = \rho_{max} = 0.5$. Results of the simulation are in Table 7.3.
- Uncertain volatility model, minimal option price (best case scenario). The max operator is replaced by min in equation (7.70), all other parameters are the same as in the worst case scenario. Results of the simulation are in Table 7.4.

In all models we used the parameters $T = 0.25$, $r = 0.05$. For each model we computed the approximations of the solution on different refinement levels. Let us denote the final time-layer (t=0) of the approximation of the solution on the k-th refinement level as A_k, and final time-layer of the reference solution as A^{ref}. We measured the error of the approximation on each refinement level in two different ways:

1. the L_1 error - the error was computed using the formula:

$$Err\ A^k = \|A^k - A^{ref}\|_1, \tag{7.75}$$

2. the error in $(x = 40, y = 40)$, computed using the formula

$$Err\ A^k = |A^k(40, 40) - A^{ref}(40, 40)|. \tag{7.76}$$

   We use this error and the value in $(x = 40, y = 40)$ to compare our results with results from the paper [35], where the same parameters are used for the uncertain volatility model.

To compute the experimental order of convergence (EOC) we used the formula (4.10). In all refinement levels we used a (rectangular) grid with equidistant time-stepping and non-equidistant space-stepping with more nodes near to the non-smooth regions of the terminal conditions. The refinements were done uniformly.

From Table 7.2 we can deduce, that the numerical solution converges also in the case of very small volatility and large correlation, although the convergence is not as smooth as in the case of moderate parameters (Table 7.1). As we can see in Tables 7.1, 7.3, the

[1] The code can be downloaded from https://github.com/igor-vyr/Tree-Grid-method

Table 7.1: Black-Scholes model, $\sigma_x = 0.3, \sigma_y = 0.5, \rho = 0.4$. M -number of timesteps, N -number of space nodes. Value, error and experimental order of convergence (EOC) in the final time layer in the point $(x, y) = (40, 40)$ and error, EOC in L_1 norm in the final time layer. As reference solution the exact solution (computed with R-library fExoticOptions) was used.

		Convergence	in $(x, y) = (40, 40)$		Convergence	in L_1
M	N	Value	Err	EOC	Err	EOC
25	35^2	1.9910	1.77E-01	-	1.21E-02	-
50	69^2	1.8211	7.02E-03	4.66	1.84E-03	2.72
100	137^2	1.8229	8.88E-03	-0.34	7.16E-04	1.36
200	273^2	1.8177	3.67E-03	1.27	3.02E-04	1.25
400	545^2	1.8141	5.31E-05	6.11	1.24E-04	1.28
800	1089^2	1.8138	1.96E-04	-1.89	5.11E-05	1.28
1600	2177^2	1.8139	1.38E-04	0.51	2.54E-05	1.01

Table 7.2: Black-Scholes model, $\sigma_x = 0.05, \sigma_y = 0.05, \rho = -0.95$. M -number of timesteps, N -number of space nodes. Value, error and experimental order of convergence (EOC) in the final time layer in the point $(x, y) = (40, 40)$ and error, EOC in L_1 norm in the final time layer. As reference solution the exact solution (computed with R-library fExoticOptions) was used.

		Convergence	in $(x, y) = (40, 40)$		Convergence	in L_1
M	N	Value	Err	EOC	Err	EOC
25	35^2	3.3619	1.28E+00	-	3.60E-02	-
50	69^2	3.8702	7.71E-01	0.73	1.96E-02	0.88
100	137^2	4.3095	3.31E-01	1.22	7.62E-03	1.37
200	273^2	4.6339	6.91E-03	5.58	5.53E-04	3.78
400	545^2	4.6465	5.73E-03	0.27	3.86E-05	3.84
800	1089^2	4.6468	5.97E-03	-0.06	4.61E-05	-0.26
1600	2177^2	4.6416	8.35E-04	2.84	9.85E-06	2.22

Table 7.3: Uncertain volatility model, worst case scenario (maximization). M -number of timesteps, N -number of space nodes, Q -number of controls. Value, error and experimental order of convergence (EOC) in the final time layer in the point $(x, y) = (40, 40)$ and error, EOC in L_1 norm in the final time layer. As reference solution, a solution computed on a fine grid with 400 timesteps, 545^2 space nodes and 256 controls was used.

			Convergence	in $(x, y) = (40, 40)$		Convergence	in L_1
M	N	Q	Value	Err	EOC	Err	EOC
25	35^2	16	2.8364	1.59E-01	-	1.17E-02	-
50	69^2	32	2.6619	1.53E-02	3.38	3.64E-03	1.68
100	137^2	64	2.6784	1.19E-03	3.68	1.17E-03	1.64
200	273^2	128	2.6784	1.20E-03	-0.01	3.07E-04	1.93

Table 7.4: Uncertain volatility model, best case scenario (minimization). M -number of timesteps, N -number of space nodes, Q -number of controls. Value, error and experimental order of convergence (EOC) in the final time layer in the point $(x, y) = (40, 40)$ and error, EOC in L_1 norm in the final time layer. As reference solution, a solution computed on a fine grid with 400 timesteps, 545^2 space nodes and 256 controls was used.

			Convergence	in $(x, y) = (40, 40)$		Convergence	in L_1
M	N	Q	Value	Err	EOC	Err	EOC
25	35^2	16	0.9847	6.95E-02	-	4.29E-03	-
50	69^2	32	0.9475	3.23E-02	1.11	2.01E-03	1.09
100	137^2	64	0.9270	1.18E-02	1.46	7.59E-04	1.41
200	273^2	128	0.9173	2.07E-03	2.50	1.31E-04	2.53

point-wise convergence may be quite non-smooth, even if the approximation is converging relatively smoothly in L_1. In the uncertain volatility model, the values in $(x = 40, y = 40)$ are similar to the values from paper [35], what verifies our method. The experimental order of convergence in the Tables 7.1, 7.3, 7.4 seems to be slightly better than the theoretical order 1, the experimental order of convergence in Table 7.2 is quite non-smooth.

The increase of the experimental order of convergence in the last refinement level in the uncertain volatility model results from using a solution computed on a fine grid as reference solution (that is disproportionally closer to the solution on the last refinement level in contrast to the solutions on previous refinement levels).

In Figure 7.3, we see the final time layer (t=0) of the approximation of option prices under the uncertain volatility model for both best and worst case scenario.

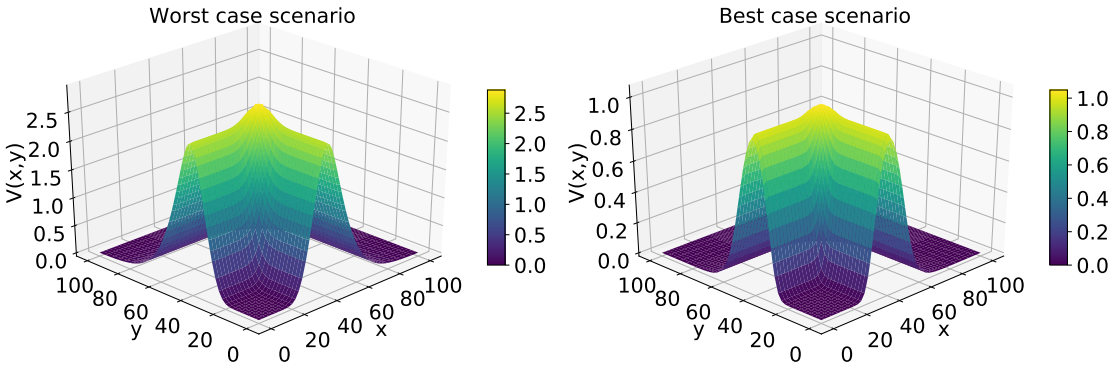

Figure 7.3: Final time layers (t=0) of the approximations of the worst case option price (maximization) and of the best case option price (minimization) from the uncertain volatility model computed with the 2D Tree-Grid method with 50 timesteps, 69^2 space nodes and 32 controls.

8 Chapter 8

Restrictions for the higher dimensional generalization of the Tree-Grid method

In the previous chapter we introduced the Tree-Grid method for solving the stochastic control problems with two space dimensions. The question that clearly arises is, whether this approach could be generalized to more space dimensions. In this chapter we will show that the most natural generalization of the Tree-Grid method to higher space-dimensions using the ideas that were presented in the previous chapter for two dimensions fails in providing non-negative weights for larger correlations. Therefore, this generalization cannot be proven to be convergent according to Theorem 1 in general. To understand the principles behind this chapter we advise the reader to start reading with Chapter 7.

8.1 P-dimensional stochastic control problem

For readers convenience, we repeat here the formulation of the general P-dimensional stochastic control problem from Chapter 2:

$$V(\bar{s},t) = \max_{\theta(s,t)\in\bar{\Theta}} \mathbb{E}\left(\int_t^T \exp\left(\int_t^k r(\bar{S}_l,l,\theta(\bar{S}_l,l))dl \right) f(\bar{S}_k,k,\theta(\bar{S}_k,k))dk \right.$$
$$\left. + \exp\left(\int_t^T r(\bar{S}_k,k,\theta(\bar{S}_k,k))dk \right) V_T(\bar{S}_T) \Big| \bar{S}_t = \bar{s} \right), \quad (8.1)$$

$$d\bar{S}_t = \bar{\mu}(\bar{S}_t,t,\theta(\bar{S}_t,t))dt + \bar{\sigma}(\bar{S}_t,t,\theta(\bar{S}_t,t))d\bar{W}_t, \quad (8.2)$$
$$0 < t < T, \quad \bar{s} \in \mathbb{R}^P, \quad P > 2$$

Here, $\bar{s}$ denotes the P-dimensional state variable, $\bar{W}_t$ is a P-dimensional Wiener process with a correlation matrix $\bar{\rho}(\bar{S}_t,t,\theta(\bar{S}_t,t))$, $\bar{S}_t$ is a P-dimensional Itô process, $\bar{\mu}(\cdot)$ is a suitable drift vector function, $\bar{\sigma}(\cdot)$ is a suitable $P \times P$ volatility matrix function with volatilities $\bar{\sigma}_l(\cdot), l = 1,2,\ldots,P$, on the diagonal and zeros elsewhere and $\bar{\Theta}$ is the space of all *suitable* control functions mapping from $\mathbb{R}^P \times [0,T]$ to a set Θ.

8.2 Construction of the P-dimensional Tree-Grid scheme

As in Chapter 7, we will try to approximate the increment of the process during a time step Δt with a stencil of possible future states with weights that can be interpreted as

probabilities (see Figure 7.1). These states will be some of the nodes from a rectangular P-dimensional space grid that is in fact one time layer of the $(p+1)$-dimensional time-space grid.

8.2.1 Notation

At first, before discussing how to generalize the stencil from Chapter 7 and how to choose the proper weights, we present here the notation used in the sequel (similar to the notation from the previous Chapter 7):

- $\bar{s}_1^{(l)}, \bar{s}_2^{(l)}, \ldots, \bar{s}_{N_l}^{(l)}$ -space discretization in the space l-th dimension ($l \leq P$).
- $t_1, t_2, \ldots, t_M$ -time discretization.
- Δt -the time-step. For simplicity we use equidistant time-stepping.
- Δs -space step between two neighboring nodes in direction of any of the P axis. For simplicity we use equidistant space stepping with space-steps of the same size in each direction.
- $h = \max(K\Delta s, \Delta t)$, where $K > 0$ is a parameter used for regulating the stencil size.
- $b = h/\Delta t$.
- $(\bar{s}_o, t_k)$ with $\bar{s}_o = (\bar{s}_{o_1}^{(1)}, \bar{s}_{o_2}^{(2)}, \ldots, \bar{s}_{o_p}^{(P)})$ -the node for which the stencil is constructed.
- $E_l = \bar{\mu}_l(\bar{s}_o, t_k, \theta)\Delta t$ for $l = 1, 2, \ldots, P$.
- $Var_l = \bar{\sigma}_l^2(\bar{s}_o, t_k, \theta)\Delta t$ for $l = 1, 2, \ldots, P$.
- $\bar{\rho}_{i,j}(\bar{s}_o, t_k, \theta)$ -the element in i-th row and j-th column of the correlation matrix $\bar{\rho}$, the correlation between the increment in the i-th and the j-th dimension, $i, j \in \{1, 2, \ldots, P\}$.
- $Cov_{i,j} = \rho_{i,j}(\bar{s}_o, t_k, \theta)\bar{\sigma}_i(\bar{s}_o, t_k, \theta)\bar{\sigma}_j(\bar{s}_o, t_k, \theta)\Delta t$ for $i, j \in \{1, 2, \ldots, P\}$.
- $W_l = Var_l + E_l^2$ for $l = 1, 2, \ldots, P$.
- $W_{i,j} = Cov_{i,j} + E_i E_j$ for $i, j \in \{1, 2, \ldots, P\}$.

8.2.2 Choosing the stencil nodes

Next, we describe how to choose the stencil nodes around an arbitrary node $(\bar{s}_o, t_k)$ for a fixed control θ, using the same approach as in Chapter 7. First, we define the following stencil coordinates for any $l \in \{1, 2, \ldots, P\}$:

If $E_l > 0$,

$$s_{l-} = \left\lfloor s_{o_l}^{(l)} - \sqrt{PW_l b} \right\rfloor_l, s_{l+} = \left\lceil \max\left(s_{o_l}^{(l)} + \sqrt{PW_l b}, s_{o_l}^{(l)} + (s_{o_l}^{(l)} - s_{l-}) \right) \right\rceil_l, \tag{8.3}$$

else if $E_l < 0$,

$$s_{l+} = \left\lceil s_{o_l}^{(l)} + \sqrt{PW_l b} \right\rceil_l, s_{l-} = \left\lfloor \min\left(s_{o_l}^{(l)} - \sqrt{PW_l b}, s_{o_l}^{(l)} - (s_{l+} - s_{o_l}^{(l)}) \right) \right\rfloor_l, \tag{8.4}$$

else ($E_l = 0$),

$$s_{l+} = \left\lceil s_{o_l}^{(l)} + \sqrt{PW_l b} \right\rceil_l, s_{l-} = \left\lfloor s_{o_l}^{(l)} - \sqrt{PW_l b} \right\rfloor_l, \tag{8.5}$$

where $\lceil \rceil_l$ resp. $\lfloor \rfloor_l$ denotes rounding to the nearest greater resp. smaller element from $\bar{s}_1^{(l)}, \bar{s}_2^{(l)}, \ldots, \bar{s}_{N_l}^{(l)}$. If such element does not exist, $\lceil z \rceil_l$ resp. $\lfloor z \rfloor_l$ will return just z.

The following nodes with the respective weights (probabilities) will be used in the stencil located at $(\bar{s}_o, t_k)$:

- Node $(\bar{s}_o, t_{k+1})$ with the weight p_o.
- For $l = 1, 2, \ldots, P$: node $(\bar{s}_{o,l+}, t_{k+1})$, where $\bar{s}_{o,l+}$ is equal to $\bar{s}_o$ up to the element on l-th position that is equal to s_{l+}. The corresponding weights of these nodes are denoted as p_{l+}.
- For $l = 1, 2, \ldots, P$: node $(\bar{s}_{o,l-}, t_{k+1})$, where $\bar{s}_{o,l-}$ is equal to $\bar{s}_o$ up to the element on l-th position that is equal to s_{l-}. The corresponding weights of these nodes are denoted as p_{l-}.
- For $i = 1, 2, \ldots, P$, $j = 1, 2, \ldots, P-1$, $j < i$: node $(\bar{s}_{o,i,j+}, t_{k+1})$, where $\bar{s}_{o,i,j+}$ is equal to $\bar{s}_o$ up to the elements on the i-th and j-th positions. The element on the i-th position is equal to s_{i+}, and the element on the j-th position is equal to s_{j+} for non-negative $W_{i,j}$ and to s_{j-} for negative $W_{i,j}$. The corresponding weights of these nodes are denoted as $p_{i,j}$.
- For $i = 1, 2, \ldots, P$, $j = 1, 2, \ldots, P-1$, $j < i$: node $(\bar{s}_{o,i,j-}, t_{k+1})$, where $\bar{s}_{o,i,j-}$ is equal to $\bar{s}_o$ up to the elements on the i-th and j-th positions. The element on the i-th position is equal to s_{i-}, and the element on the j-th position is equal to s_{j-} for non-negative $W_{i,j}$ and to s_{j+} for negative $W_{i,j}$. The weights of these nodes are the same as weights of the nodes $(\bar{s}_{o,i,j+}, t_{k+1})$ denoted as $p_{i,j}$.

Now, let us define the difference operators

$$\Delta_{l+} s = s_{l+} - s_{o_l}^{(l)}, \quad \Delta_{l-} s = s_{o_l}^{(l)} - s_{l-}, \tag{8.6}$$

for $l = 1, 2, \ldots, P$. As we assume equidistant grid in the space domain in this chapter, it holds $\Delta_{l+} s = \Delta_{l-} s$ and therefore we will sometimes denote this distance just as $\Delta_l s$ ($\Delta_l s := \Delta_{l+} s = \Delta_{l-} s$).

8.2.3 Choosing the stencil weights (probabilities)

To match the first two moments of the approximative increment of the stochastic process and of the increment of the discrete process defined by the "stencil nodes" and their weights and to ensure that the weights can be interpreted as probabilities (are non-negative and

sum up to 1), we require:

$$p_o + \sum_{l=1}^{P}(p_{l+} + p_{l-}) + 2\sum_{\substack{i,j=1\\ j<i}}^{P} p_{i,j} = 1, \tag{8.7}$$

$$\left(p_{l+} + \sum_{j<l} p_{l,j} + \sum_{l<i} p_{i,l}\right)\Delta_{l+}s - \left(p_{l-} + \sum_{j<l} p_{l,j} + \sum_{l<i} p_{i,l}\right)\Delta_{l-}s = E_l, \tag{8.8}$$

for $l = 1, 2, \ldots, P$.

$$\left(p_{l+} + \sum_{j<l} p_{l,j} + \sum_{l<i} p_{i,l}\right)(\Delta_{l+}s)^2 + \left(p_{l-} + \sum_{j<l} p_{l,j} + \sum_{l<i} p_{i,l}\right)(\Delta_{l-}s)^2 = W_l, \tag{8.9}$$

for $l = 1, 2, \ldots, P$.

$$p_{i,j}(\Delta_{i+}s\Delta_{j+}s + \Delta_{i-}s\Delta_{j-}s) = W_{i,j} \tag{8.10}$$

for $i = 1, 2, \ldots, P, \quad j = 1, 2, \ldots, P-1, \quad j < i \quad$ and $W_{i,j} \geq 0$

$$p_{i,j}(\Delta_{i+}s\Delta_{j-}s + \Delta_{i-}s\Delta_{j+}s) = |W_{i,j}| \tag{8.11}$$

for $i = 1, 2, \ldots, P, \quad j = 1, 2, \ldots, P-1, \quad j < i \quad$ and $W_{i,j} < 0$

$$p_o, p_{l+}, p_{l-}, p_{i,j}, \geq 0 \tag{8.12}$$

for $l = 1, 2, \ldots, P, \quad i = 1, 2, \ldots, P, \quad j = 1, 2, \ldots, P-1, \quad j < i.$

The formula for the analytical solution of the $P^2 + P + 1$ equations (8.7)-(8.11) is very long, but can be simplified under the assumption of equidistant spatial grid ($\Delta_l s = \Delta_{l+}s = \Delta_{l-}s$). The solution for such equidistant spatial grid reads:

$$\begin{aligned} p_o =& \frac{\prod_{i=1}^{P}(\Delta_i s)^2 + \sum_{\substack{i,j=1\\ j<i}}^{P}\left(\left(\prod_{\substack{l=1\\ l\neq i,j}}^{P}(\Delta_l s)^2\right)\Delta_i s\Delta_j s|W_{i,j}|\right)}{\prod_{l=1}^{P}(\Delta_l s)^2} \\ & - \frac{\sum_{i=1}^{n}\left(\left(\prod_{\substack{j=1\\ j\neq i}}^{P}(\Delta_j s)^2\right)W_i\right)}{\prod_{l=1}^{P}(\Delta_l s)^2}, \end{aligned} \tag{8.13}$$

$$p_{i,j} = \frac{|W_{i,j}|}{2\Delta_i s\Delta_j s}, \tag{8.14}$$

$$p_{l\pm} = \frac{\left(\prod_{\substack{i=1\\ i\neq l}}^{P}\Delta_i s\right)W_l - \sum_{\substack{i=1\\ i\neq l}}^{P}\left(\left(\prod_{\substack{j=1\\ j\neq i}}^{P}\Delta_j s\right)|W_{l,i}|\right) \pm \left(\prod_{i=1}^{P}\Delta_i s\right)E_l}{2(\Delta_l s)^2\prod_{\substack{i=1\\ i\neq l}}^{P}\Delta_i s}. \tag{8.15}$$

8.3 Appearance of possibly negative weights

The weight p_o (8.13) is positive due to the construction of the stencil. The weights $p_{i,j}$ (8.14) are also clearly positive. The problem may however appear in the case of weights p_{l+}, p_{l-} defined by (8.15). Let us for example assume $E_l = 0$ for $l = 1, 2, \ldots, P$, $W_i = W_j = W_{i,j}$ for $i, j = 1, 2, \ldots, P$. Then, also $\Delta_i s = \Delta_j s$ for $i, j = 1, 2, \ldots, P$ and $p_{l\pm}$ are

negative for $l = 1, 2, \ldots, P \geq 3$. We can artificially increase the diffusion W_l or reduce the covariance $W_{i,j}$ while still preserving consistency with the original problem in some similar way as in Chapter 7, Section 7.2.4. Then W_l will be replaced by $W_l + \mathcal{O}(h^z)\Delta t$ and $W_{i,j}$ will be replaced with $W_{i,j} + \mathcal{O}(h^z)\Delta t$, $z > 0$. However, it is clear that even such modification will not restore the non-negativity of $p_{l\pm}$ for vanishing h and $P \geq 3$. Of course, for smaller covariance terms $W_{i,j}$ the weights (probabilities) may be non-negative, but we cannot ensure the non-negativity for an arbitrary problem. As the non-negativity of weights is equivalent to the monotonicity of the scheme, a property that is needed for convergence proof according to the theory of Barles and Souganidis [4], we also cannot ensure the convergence of the P-dimensional Tree-Grid method for an arbitrary problem.

Although the Tree-Grid method cannot be generalized to higher dimensions (at least not in such a straightforward manner) for an arbitrary problem, the generalization described here may be monotone and thus feasible for control problems with smaller terms $W_{i,j}$.

8.4 Ideas from Tree-Grid schemes applicable to other methods

Some of the ideas presented for the Tree-Grid method can be reused also for other wide stencil methods (e.g. methods from [12]) that are feasible also in higher dimensions:

- By making the stencil dependent also on the time-step Δt in similar manner as in this or in the previous chapter, unconditional stability and monotonicity can be achieved. For larger time steps that would violate the CFL condition, the stencil will simply get wider.
- By following the Remarks 8, 10 one can avoid conditions on the time step regarding the discount factor. This is applicable also in the implicit finite difference methods (e.g. [16, 9]), if one solves in each time layer the corresponding PDE without discounting (that means with zero-coefficient in front of V-term) and then discounts the whole time layer as proposed here.

The remaining drawback of the higher-dimensional (dimension higher than 2) wide-stencil methods after applying the tricks described above in contrast to the Tree-Grid method is the necessity of interpolation.

9 Chapter 9
Conclusion and outlook

In this thesis, we presented new approaches for solving the SCPs and the related HJB equations. We introduced a modification of the PCPT scheme, the so-called PPPT scheme. Although we cannot prove the convergence of this method to the correct solution in general, as the method relays on prediction made on a coarse grid, we have shown experimentally on two numerical examples from finance the superiority of the method compared to the PCPT method.

However, the main contributions of this thesis are undoubtedly the Tree-Grid methods. These new methods are constructed for the cases of one or two space dimensions. We proved the convergence of these methods in both cases. The methods are explicit, but still unconditionally stable and convergent for an arbitrary grid. The unconditional stability is not common among the explicit methods, as these need some CFL condition to be fulfilled in most cases. Therefore, the Tree-Grid methods are methods of choice if explicitness is desirable - either because of the lower time complexity or if parallelization of the algorithm is needed. Moreover, the 2D Tree-Grid method posses another advantage - while falling into the class of the explicit wide stencil schemes, it doesn't use any interpolation in contrast to other schemes in this class. A disadvantage that is also addressed in this thesis is, that we cannot generalize the Tree-Grid method to dimensions higher than two for an arbitrary problem. For the 1D Tree-Grid method we developed also an modification leading to stencil of control-independent size, allowing us to use more effective algorithms for the search of optimal control. We have shown the convergence of all Tree-Grid methods (one dimensional, two dimensional and modified one-dimensional) on numerical examples from finance. To verify our results, we used in the problem formulations parameters from the literature.

9.1 Outlook of the future research

New approaches in this thesis give rise to many interesting questions and ideas. Let us introduce some of the possible directions of the future research related to this thesis:

1. **Convergence conditions for the PPPT method.** The PPPT method from Chapter 4 is rather heuristic in the sense that it is stable and also convergent, but we cannot be sure that the method converges to the actual (viscosity) solution of the SCP. Therefore, it would be interesting to find some conditions of convergence to the correct solution, or at least some bounds on the error of the approximation.

2. **Analytical solution of the optimization part.** In Chapter 6, we state that it should be possible to find also analytical (closed-form) solution of the optimization problem under the control independent stencil modification of the Tree-Grid method. However, this is not trivial due to the artificial diffusion making the target

function more complicated. Deriving such closed-form solution would speed up the computations significantly.

3. **Control independent stencil for the 2D Tree-Grid method.** In Chapter 6, we introduced the control-independent stencil only for the one-dimensional Tree-Grid method. It would be interesting to examine, if one can use some similar technique also in the case of 2D Tree-Grid method. This could again lead to an significant speed-up.

4. **Optimal time-step space-step ratio.** CFL conditions are not needed for the Tree-Grid methods, as these are unconditionally stable. Therefore, as also in the case of the implicit FDMs, the question arises, what the optimal ratio between the time-step and the space steps is. This can be easily examined experimentally, but also an analytic result on this question would be interesting.

5. **Boundary conditions.** By handling the boundary conditions in the Tree-Grid method, we supposed that we know, (or can artificially define) the solution behind the boundaries. This approach was numerically successful, but for example in [35] another approach is proposed: the stencil shrinks near to the boundary, so that no data from outside of the numerical domain is needed. Would it be possible to introduce a similar modification to the Tree-Grid methods?

6. **Parallelization.** As the Tree-Grid methods are explicit they can be easily parallelized. Beside the standard parallelization technique with communication after each time layer, also another approach may be possible: there may exist sets of nodes ("subtrees") in the computational domain that are completely independent in the sense, that none of the values in a node in one set can influence the value in any other node from a different set. Such situation is illustrated in Figure 9.1. If one can find decomposition of the discretization domain into such independence sets, one can run an separate Tree-Grid algorithm on each of this sets with no communication. This idea is applicable also to other wide-stencil schemes. We should note, that there may be just one independence set (all node values impact one another), in which case such parallelization is not feasible.

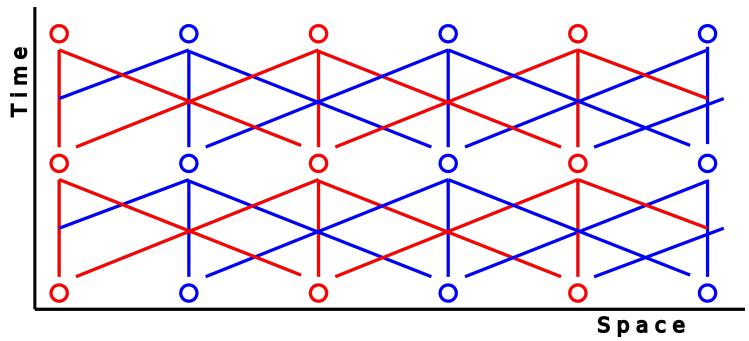

Figure 9.1: Illustration of independence sets of nodes. Blue nodes are connected only with other blue nodes, therefore values in blue nodes do not depend on the values in red nodes. The same holds for the red-nodes.

7. **Monotonicity conditions for higher dimensions.** In Chapter 8, we have shown that a generalization of the Tree-Grid method to dimensions higher than two is not

possible for an arbitrary problem, however might still be possible for some problems with mild covariances. A formulation of the exact conditions on the problem parameters leading to a monotone Tree-Grid scheme in arbitrary dimension would be an interesting contribution.

8. **Modification of the wide-stencil schemes.** In Section 8.4, we discussed approaches used in the construction of the Tree-Grid method that could be employed also in other wide-stencil methods, leading to unconditional stability. However, precise examination and implementation of these ideas is still needed.

9. **HJBI equations.** Another possible future research direction is the application of the Tree-Grid method to the HJBI equations resulting from the stochastic differential games. The adaptation of the method to this setting should be straightforward.

We hope that beside the presented results, the reader could find in this thesis also new ideas for future research in the field of numerical methods for the stochastic control problems and the HJB equations.

References

[1] J. Ahn and M. Song. Convergence of the trinomial tree method for pricing European/American options. *Applied Mathematics and Computation*, 189(1):575–582, 2007.

[2] D. Applebaum. *Lévy processes and stochastic calculus*. Cambridge University Press, 2009.

[3] M. Avellaneda, A. Levy, and A. Parás. Pricing and hedging derivative securities in markets with uncertain volatilities. *Applied Mathematical Finance*, 2(2):73–88, 1995.

[4] G. Barles and P. E. Souganidis. Convergence of approximation schemes for fully nonlinear second order equations. In *Asymptotic Analysis*, number 4, pages 2347–2349, 1991.

[5] J. Barraquand and T. Pudet. Pricing of American path-dependent contingent claims. *Mathematical Finance*, 6(1):17–51, 1996.

[6] R. E. Bellman. *Dynamic Programming*. Courier Dover Publications, 1957.

[7] F. Black and M. Scholes. The pricing of options and corporate liabilities. *Journal of political economy*, 81(3):637–654, 1973.

[8] J. F. Bonnans and H. Zidani. Consistency of generalized finite difference schemes for the stochastic HJB equation. *SIAM Journal on Numerical Analysis*, 41(3):1008–1021, 2003.

[9] Y. Chen and J. W. L. Wan. Monotone mixed narrow/wide stencil finite difference scheme for Monge-Ampère equation. *arXiv preprint arXiv:1608.00644*, 2016.

[10] Z. Chen and P. A. Forsyth. A Semi-Lagrangian approach for natural gas storage valuation and optimal operation. *SIAM Journal on Scientific Computing*, 30(1):339–368, 2007.

[11] M. G. Crandall, H. Ishii, and P-L. Lions. User's guide to viscosity solutions of second order Partial Differential Equations. *Bulletin of the American Mathematical Society*, 27(1):1–67, 1992.

[12] K. Debrabant and E. R. Jakobsen. Semi-Lagrangian schemes for linear and fully nonlinear diffusion equations. *Mathematics of Computation*, 82(283):1433–1462, 2013.

[13] K. Debrabant and E. R. Jakobsen. Semi-Lagrangian schemes for linear and fully nonlinear Hamilton-Jacobi-Bellman equations. *arXiv preprint arXiv:1403.1217*, 2014.

[14] M. Ehrhardt, editor. *Nonlinear models in mathematical finance: new research trends in option pricing*. Nova Science, Hauppauge, 2008.

[15] D. E. Ferguson. Fibonaccian searching. *Communications of the ACM*, 3(12):648, 1960.

[16] P. A. Forsyth and G. Labahn. Numerical methods for controlled Hamilton-Jacobi-Bellman PDEs in finance. *Journal of Computational Finance*, 11(2):1, 2007.

[17] P. A. Forsyth and K. R. Vetzal. Numerical methods for nonlinear PDEs in finance. In *Handbook of Computational Finance*, pages 503–528. Springer, 2012.

[18] P. A. Forsyth, K. R. Vetzal, and R. Zvan. Convergence of numerical methods for valuing path-dependent options using interpolation. *Review of Derivatives Research*, 5(3):273–314, 2002.

[19] Ch. Grossmann, H-G. Roos, and M. Stynes. *Numerical treatment of partial differential equations*. Springer, 2007.

[20] M. Günther and A. Jüngel. *Finanzderivate mit MATLAB: mathematische Modellierung und numerische Simulation*. Springer-Verlag, second edition, 2010.

[21] M. Halická, P. Brunovský, and P. Jurča. *Optimálne riadenie*. EPOS, 2009.

[22] J. C. Hull and A. D. White. Efficient procedures for valuing European and American path-dependent options. *The Journal of Derivatives*, 1(1):21–31, 1993.

[23] M. Ishikawa. Optimal vaccination strategy under saturated treatment using the stochastic SIR model. *Transactions of the Institute of Systems, Control and Information Engineers*, 26:382–388, 2013.

[24] S. Kilianová and D. Ševčovič. A transformation method for solving the Hamilton-Jacobi-Bellman equation for a constrained dynamic stochastic optimal allocation problem. *The ANZIAM Journal*, 55(01):14–38, 2013.

[25] I. Kossaczký, M. Ehrhardt, and M. Günther. Modifications of the PCPT method for HJB equations. In *Application of Mathematics in Technical and Natural sciences: 8th International Conference for Promoting the Application of Mathematics in Technical and Natural Sciences-AMiTaNS'16*, volume 1773, page 030002. AIP Publishing, 2016.

[26] I. Kossaczký, M. Ehrhardt, and M. Günther. On the non-existence of higher order monotone approximation schemes for HJB equations. *Applied Mathematics Letters*, 52:53–57, 2016.

[27] I. Kossaczký, M. Ehrhardt, and M. Günther. The Tree-Grid Method with control-independent stencil. In *Proceedings of Equadiff 2017 Conference*, pages 79–88, 2017.

[28] I. Kossaczký, M. Ehrhardt, and M. Günther. A new convergent explicit Tree-Grid method for HJB equations in one space dimension. *Numerical Mathematics: Theory, Methods and Applications*, 11:1–29, 2018.

[29] I. Kossaczký, M. Ehrhardt, and M. Günther. The Two-dimensional Tree-Grid Method. Preprint 18/02 on www.imacm.uni-wuppertal.de, 2018.

[30] N. V. Krylov. Approximating value functions for controlled degenerate diffusion processes by using piece-wise constant policies. *Electronic Journal of Probability*, 4(2):1–

19, 1999.

[31] H. Kushner and P. G. Dupuis. *Numerical methods for stochastic control problems in continuous time*, volume 24 of *Applications of Mathematics*. Springer Science & Business Media, 2013.

[32] P-L. Lions. Hamilton-Jacobi-Bellman equations and the optimal control of stochastic systems. In *Proceedings of the International Congress of Mathematicians*, volume 1, 1983.

[33] P-L. Lions. Optimal control of diffusion processes and Hamilton–Jacobi–Bellman equations part 2: viscosity solutions and uniqueness. *Communications in Partial Differential Equations*, 8(11):1229–1276, 1983.

[34] Q. Liu and X. Zhou. An introduction to viscosity solution theory. Introductory notes, 2013.

[35] K. Ma and P. A. Forsyth. An unconditionally monotone numerical scheme for the two-factor uncertain volatility model. *IMA Journal of Numerical Analysis*, 37(2):905–944, 2016.

[36] S. Mataramvura and B. Øksendal. Risk minimizing portfolios and HJBI equations for stochastic differential games. *Stochastics An International Journal of Probability and Stochastic Processes*, 80(4):317–337, 2008.

[37] R. C. Merton et al. An analytic derivation of the efficient portfolio frontier. *Journal of Financial and Quantitative Analysis*, 7(4):1851–1872, 1972.

[38] M. Nisio. *Stochastic control theory: dynamic programming principle*, volume 72 of *Probability Theory and Stochastic Modelling*. Springer, 2014.

[39] B. Øksendal and A. Sulem. Stochastic control of Itô-Lévy processes with applications to finance. *Communications on Stochastic Analysis*, 8(1), 2014.

[40] D. M. Pooley, P. A. Forsyth, and K. R. Vetzal. Numerical convergence properties of option pricing PDEs with uncertain volatility. *IMA Journal of Numerical Analysis*, 23(2):241–267, 2003.

[41] Ch. Reisinger and P. A. Forsyth. Piecewise constant policy approximations to Hamilton-Jacobi-Bellman equations. *Applied Numerical Mathematics*, 103:27–47, 2016.

[42] J. B. Rosser. Nine-point difference solutions for Poisson's equation. *Computers & Mathematics with Applications*, 1(3):351–360, 1975.

[43] D. Ševcovic, B. Stehlíková, and K. Mikula. *Analytical and numerical methods for pricing financial derivatives*. Nova Science, Hauppauge, 2011.

[44] S. E. Shreve. *Stochastic calculus for finance II: Continuous-time models*, volume 11. Springer Science & Business Media, 2004.

[45] Q. S. Song. Convergence of Markov chain approximation on generalized HJB equation and its applications. *Automatica*, 44(3):761–766, 2008.

[46] D. Tavella and C. Randall. *Pricing Financial Instruments: The Finite Difference Method*. Wiley Series in Financial Engineering. Wiley, 2000.

[47] J. Topper. A finite element implementation of passport options. Master's thesis, 2003.

[48] J. Wang and P. A. Forsyth. Maximal use of central differencing for Hamilton-Jacobi-Bellman PDEs in finance. *SIAM Journal on Numerical Analysis*, 46(3):1580–1601, 2008.

[49] J. Wang and P. A. Forsyth. Numerical solution of the Hamilton–Jacobi–Bellman formulation for continuous time mean variance asset allocation. *Journal of Economic Dynamics and Control*, 34(2):207–230, 2010.

[50] J. Yong and X. Y. Zhou. *Stochastic controls: Hamiltonian systems and HJB equations*, volume 43 of *Stochastic Modelling and Applied Probability*. Springer Science & Business Media, 1999.