COMENIUS UNIVERSITY IN BRATISLAVA
FACULTY OF MATHEMATICS, PHYSICS AND INFORMATICS

ANALYSIS OF SOLUTIONS OF THE HAMILTON-JACOBI-BELLMAN EQUATION

DIPLOMA THESIS

2014 Bc. Igor KOSSACZKÝ

COMENIUS UNIVERSITY IN BRATISLAVA
FACULTY OF MATHEMATICS, PHYSICS AND INFORMATICS

ANALYSIS OF SOLUTIONS OF THE HAMILTON-JACOBI-BELLMAN EQUATION

DIPLOMA THESIS

Study programme:	Economic and Financial Mathematics
Field of study:	1114 Applied Mathematics
Department:	FMFI.KAMŠ -Department of Applied Mathematics and Statistics
Supervisor:	prof. RNDr. Daniel Ševčovič, CSc.

Bratislava 2014

Bc. Igor KOSSACZKÝ

UNIVERZITA KOMENSKÉHO V BRATISLAVE
FAKULTA MATEMATIKY, FYZIKY A INFORMATIKY

ANALÝZA RIEŠENÍ HAMILTON-JACOBI-BELLMANOVEJ ROVNICE

DIPLOMOVÁ PRÁCA

Študijný program:	Ekonomická a finančná matematika
Študijný odbor:	1114 Aplikovaná matematika
Školiace pracovisko:	Katedra aplikovanej matematiky a štatistiky
Vedúci práce:	prof. RNDr. Daniel Ševčovič, CSc.

Bratislava 2014

Bc. Igor KOSSACZKÝ

93723727

Comenius University in Bratislava
Faculty of Mathematics, Physics and Informatics

THESIS ASSIGNMENT

Name and Surname: Bc. Igor Kossaczký
Study programme: Economic and Financial Mathematics (Single degree study, master II. deg., full time form)
Field of Study: 9.1.9. Applied Mathematics
Type of Thesis: Diploma Thesis
Language of Thesis: English
Secondary language: Slovak

Title: Analysis of solutions of the Hamilton-Jacobi-Bellman equation

Aim: The aim of this study is qualitative and numerical analysis of solutions of the nonlinear partial differential Hamilton-Jacobi-Bellman equation, which arises by modeling of optimal portfolio composition using stochastic dynamic programming. In this study we use knowledge how to model evolution of the underlying process by Brownian motion. Numerical results of solving Hamilton-Jacobi-Bellman equation will be interpreted on real data.

Supervisor: prof. RNDr. Daniel Ševčovič, CSc.
Department: FMFI.KAMŠ - Department of Applied Mathematics and Statistics
Head of department: prof. RNDr. Daniel Ševčovič, CSc.

Assigned: 25.01.2013

Approved: 04.02.2013 prof. RNDr. Daniel Ševčovič, CSc.
Guarantor of Study Programme

....
Student Supervisor

93723727

Univerzita Komenského v Bratislave
Fakulta matematiky, fyziky a informatiky

ZADANIE ZÁVEREČNEJ PRÁCE

Meno a priezvisko študenta: Bc. Igor Kossaczký
Študijný program: ekonomická a finančná matematika (Jednoodborové štúdium, magisterský II. st., denná forma)
Študijný odbor: 9.1.9. aplikovaná matematika
Typ záverečnej práce: diplomová
Jazyk záverečnej práce: anglický
Sekundárny jazyk: slovenský

Názov: Analýza riešení Hamilton-Jacobi-Bellmanovej rovnice / *Analysis of solutions of the Hamilton-Jacobi-Bellman equation*

Cieľ: Cieľom práce bude kvalitatívna a numerická analýza riešení nelineárnej parciálnej diferenciálnej Hamilton-Jacobi-Bellmanovej rovnice, ktorá vzniká v modelovaní skladby optimálneho portfólia pomocou stochastického dynamického programovania. V práci využijeme poznatky o modelovaní vývoja podkladového procesu pomocou Brownovho pohybu. Numerické výsledky riešenia Hamilton-Jacobi-Bellmanovej rovnice budú interpretované na reálnych dátach.

Vedúci: prof. RNDr. Daniel Ševčovič, CSc.
Katedra: FMFI.KAMŠ - Katedra aplikovanej matematiky a štatistiky
Vedúci katedry: prof. RNDr. Daniel Ševčovič, CSc.

Dátum zadania: 25.01.2013

Dátum schválenia: 04.02.2013

prof. RNDr. Daniel Ševčovič, CSc.
garant študijného programu

študent

vedúci práce

Here, I want to say special Thank You to my supervisor, Professor Daniel Ševčovič for helpful consultations and support, and to Professor Matthias Ehrhardt for inspiring discussions on boundary conditions.

Abstract

KOSSACZKÝ, Igor: Analysis of solutions of the Hamilton-Jacobi-Bellman equation [Diploma Thesis], Comenius University in Bratislava, Faculty of Mathematics, Physics and Informatics, Department of Applied Mathematics and Statistics; Supervisor: prof. RNDr. Daniel Ševčovič, CSc., Bratislava, 2014, 64 p.

Hamilton-Jacobi-Bellman (HJB) equation is one of the cornerstones of Stochastic time-continuous optimal control theory. It is an fully nonlinear partial differential equation, so the analytical solution is in many cases not feasible. Therefore, aim of this work is to develop and test numerical methods for solving one special type of this equation. In our case, the equation arises from portfolio optimization problem, presented in [9]. Our approach is based on studying numerical methods for standard differential equation, and modifying them, to be suitable for our equation. We implemented presented methods in Octave, and tested accuracy of this methods on examples with known analytical solution. We compared computational time and suitability of the methods in case of different parameters. For comparison, we implemented also the method proposed in [9]. We discussed, how changes in model parameters affect the shape of the solution of HJB equation and interpreted the changes in shape economically. We generalized result from [9] about the solution of quadratic programming problem arising form our HJB equation and proved our generalization.

Keywords: Hamilton-Jacobi-Bellman equation, Portfolio optimization, Stochastic optimal control, Godunov method, Riccati transformation, Numerical analysis of partial differential equations

Abstrakt

KOSSACZKÝ, Igor: Analýza riešení Hamilton-Jacobi-Bellmanovej rovnice [Diplomová práca], Univerzita Komenského v Bratislave, Fakulta matematiky, fyziky a informatiky, Katedra aplikovanej matematiky a štatistiky; školiteľ: prof. RNDr. Daniel Ševčovič, CSc., Bratislava, 2014, 64 s.

Hamilton-Jacobi-Bellmanova rovnica je jedným zo základov teórie stochastického optimálneho riadenia so spojitým časom. Je to plne nelineárna parciálna diferenciálna rovnice, čiže analytické riešenie nie je v mnohých prípadoch dostupné. Preto je cieľom tejto práce vyvynúť numerické metódy na riešenie jedného typu tejto rovnice. V našom prípade rovnica vychádza z problému optimalizácie portfólia, prezentovaného v [9]. Náš prístup je založený na štúdiu štandardných diferenciálnych rovníc a ich prispôsobení, tak aby boli vhodné pre našu rovnicu. Uvedené metódy sme naimplementovali v Octave, a testovali sme ich presnosť na príkladoch so známym analytickým riešením. Porovnali sme výpočtový čas a vhodnosť týchto metód v prípade rôznych parametrov. Pre porovnanie sme naprogramovali aj metódu navrhnutú v [9]. Pozreli sme sa, ako zmeny v parametroch ovplyvňujú tvar riešenia, a zmeny v tvare sme ekonomicky interpretovali. Zovšeobecnili sme výsledok z [9] o riešení problému kvadratického programovania vychádzajúceho z našej HJB rovnice a naše zovšeobecnenie sme dokázali.

Keywords: Hamilton-Jacobi-Bellmanova rovnica, Optimalizácia portfólia, Stochastické optimálne riadenie, Godunovova metóda, Riccatiho transformácia, Numerická analýza parciálnych diferenciálnych rovníc

Contents

Introduction

Stochastic optimal control is mathematical discipline concerning choice of control variables or control functions in order to influence some process, so that the outcome will be in some sense optimal. However, as already the name of this discipline reveals, some stochastic (random) factors come also into play. Such theory is ideal for many different fields of applied mathematics. In our work, we will be concerned with solving portfolio optimization problem in time. This issue is part of financial mathematics and can be loosely characterized by questions: How should the investor invest his money to achieve highest return? How should he react on market changes?

One of the possible approaches how to solve some stochastic optimal control problem is to solve the related Hamilton-Jacobi-Bellman (HJB) Equation. This work is largely based on paper [9] that is concerned with numerical methods for solving this equation. The reason is that analytical solution for such fully-nonlinear PDE's are often not feasible. In [9], this equation is transformed to quasi-linear parabolic PDE form, which can be solved more effectively. However, in this work we will examine some methods for the original HJB Equation without transformation. As we can see, in this work we combine several different parts of applied mathematics: numerical analysis, optimal control, stochastic processes, partial differential equations and financial mathematics.

The work is divided into 6 chapters. In Chapter 1 we introduce the reader to portfolio optimization, quickly sketch the concept of stochastic differential equations and apply it in modeling the asset price evolvement. After that, we state the derivation of Hamilton-Jacobi-Bellman Equation for our problem. Chapter 2 represents preparation for the next chapters where numerical methods will be introduced. We define here some useful equivalent forms of HJB equation describe the implementation of some Octave functions that will be routinely used later. The most important part of this chapter is generalization of some results of a specific quadratic programming problem related to our HJB equation from [9].

Chapter 3 and Chapter 4 represent the main core of this work. In Chapter 3, we present explicit methods for solving HJB Equation. We also discuss here two different approaches to boundary conditions. Chapter 4 is devoted to implicit numerical methods. Method based on Riccati transformation from [9] is also presented.

Performance of the numerical methods is discussed in Chapter 5. We test these methods with parameters based on real data from Swiss market index SMI. We are concerned with experimental order of convergence as well as with time complexity. Some cases of error behavior are also discussed. In Chapter 6 we take a closer look on the shape of solution of our HJB Equation. We will examine, how the solution changes if some parameters are changed, and we will try to interpret the results economically.

1 Hamilton-Jacobi-Bellman equation in portfolio optimization

The goal of this chapter is to introduce a dynamic portfolio optimization problem, which is the motivation of our study of Hamilton-Jacobi-Bellman-equation. Although larger part of this theory is introduced in [9], for understanding of the problematics, it is necessary to introduce it also here. Moreover, we will describe here the basic idea of derivation of Hamilton-Jacobi-Bellman equation.

1.1 Introduction to portfolio optimization

Portfolio optimization is one of the most important parts of financial mathematics. Let us first clarify a few expressions from finance, that will be used often in this text. By an asset we understand anything with some value that is traded on the market. Commonly traded assets are for example stocks of companies (in this work we will use stocks of largest swiss companies), commodities (gold, oil, coffee), currencies (euro, dollar) or financial derivatives (options, forwards). By an investor we refer to a person owning and trading assets. In our case, we will think about an investor, whose only motivation to trade assets is to make money, not to consume. For example, our investor will not buy coffee to drink it, but to sell it later for better price. Now, a set of assets owned by an investor is named simply portfolio. The main concern of portfolio optimization is the problem, how to set or change the composition of a portfolio, according to investor preferences and price development on the market. Already from this simple and loose formulation, many questions arise:

- How to capture properties of assets in the portfolio and the relationships between them?
- How to model the evolvement of these properties and relationships?
- How to express investor's preferences?
- How exactly is the portfolio problem formulated?

Now, we will try to shortly sketch answers to these questions. For deeper understanding of the problematics we refer to some Financial mathematics textbooks, as for example [19].

Model of the asset price and its dynamics

Since the future price of an asset is not known exactly today, we have to model it as an random variable. Expected value of this variable express our expectation about the future value. Standard deviation models the volatility of the price, so it expresses the level of uncertainty about its value. If we want to capture dependence between two asset prices, we model it by correlation between the related random variables. However, we have somehow to model also the development of the asset price. Clearly the price in the future depends on the price today, but cannot be forecasted precisely. Ideal for modeling such behavior are stochastic differential equations, which will be quickly introduced now. Let us begin with the definition of Wiener process:

Definition 1.1. *Wiener process is a stochastic process W_t which fulfills the following conditions:*

1. *All trajectories of Wiener process are continuous almost surely, and $W_0 = 0$ almost surely.*
2. *Random variable W_t (value of Wiener process in time t) has normal distribution $N(0,t)$.*
3. *$W_{t+s} - W_s$ has normal distribution $N(0,t)$, and the increments of W_t are independent, e.g. $W_{t_1}, W_{t_2} - W_{t_1}, \ldots, W_{t_k} - W_{t_{k-1}}$ are independent for all $0 \le t_1 < \ldots < t_k$.*

Now, let us think about an ordinary differential equation: $\frac{\partial y(t)}{\partial t} = f(y(t),t)$. If we approximate $\frac{\partial y(t)}{\partial t}$ by $\frac{dy(t)}{dt}$ where $dy(t)$ is differential of function $y(t)$ and formally multiply the equation by dt, we end up with ODE in another form: $dy(t) = f(y(t),t)dt$. This form describes change in $y(t)$ during a small time interval, dt. Now, we can denote the change in value of Wiener process in time t during a small time interval dt as dW_t. Using this change in Wiener process we can add stochastic part to our new form of ODE, and we get stochastic differential equation (SDE):

$$dy(t) = f(y(t),t)dt + g(y(t),t)dW_t \tag{1}$$

The part $g(y(t),t)dW_t$ describes how the function $y(t)$ changes during a small time interval, if the change in Wiener process during this small time interval is dW_t. The change dW_t is stochastic, therefore the solution of an SDE is a stochastic process, in contrast to ODE, where the solution is function. Stochastic process, which is solution of the equation in the form (1) is called **Ito process**. If we need to model two Ito processes that will be correlated, we will use two independent Wiener processes W_t^1 and W_t^2, and the SDE's will look like

$$\begin{aligned} dy(t) &= f(y(t),t)dt + g_1(y(t),t)dW_t^1 + g_2(y(t),t)dW_t^2 \\ dx(t) &= v(x(t),t)dt + w_1(x(t),t)dW_t^1 + w_2(x(t),t)dW_t^2 \end{aligned}$$

Let us note that this introduction to stochastic differential equations was just very formal, for more exact derivation of theory we refer to [19], [24], [1].

Investor's preferences

Different investors may have pretty different preferences. However, some characteristics of an investor are the same almost everytime. We can assume that for every investorn higher return of his investment activities is better than lower. Most investors will also prefer less risky investments over the risky one. However, higher return is often bound with risky investments in contrast to low but safe return. That's where the individual characteristics of an investor come into play. There are several approaches how to describe preferences of an investor. In Markowitz model (see [18], or [19]), an investor declares minimal return that he expects and then he try to minimize the risk (volatility) of the investition. Another option may be to declare the maximal risk we are willing to undergo, and then to maximize the expected return. This two approaches are often called mean-variance optimization. However, utility theory proposed by Von

Neumann and Morgenstern in [20], provides us with another approach: each investor is characterized by utility function $U(x)$, which assigns to any amount of money x subjective utility that this money amount has for the investor. Then, the investor can decide between two investments with returns characterized by random variables X, Y simply by comparing expected values of their utility functions, $\mathbb{E}(U(X))$, $\mathbb{E}(U(Y))$. This approach is called utility optimization. As shown in [11], the utility theory approach is consistent with mean-variance optimization. Moreover, as shown in [23], Markowitz model can be also expressed in terms of utility maximization by choosing suitable quadratic utility function (see also [12]).

Formulation of portfolio optimization problem

We can have many different formulations of portfolio optimization problem, depending on the situation. If the investor wants simply to put money in assets, and withdraw it after some time, we speak about one-period portfolio problem which leads to Markowitz model or nonlinear programing (maximizing utility function with respect to budget constraints). If the investor wants to modify the portfolio composition in some time points, in order to reflect market changes, we speak about multi-period portfolio optimization. As in [23] or [12], often discrete optimal control is used in this case. However, investor may change the portfolio composition pretty often, so that we can approximate his behavior with continuous changing of portfolio weights. This approximation leads to continuous optimal control and to the Hamilton-Jacobi-Bellman Equation which is the main concern of this work. For better understanding of the background behind nonlinear programming and optimal control, we refer to [7], [5], [6].

1.2 Modeling the evolution of portfolio value

Now, we will describe a standard model for portfolio value development which is used for example in [9], [19] or [24]. Let us consider n assets combined in portfolio with weights represented by an n-dimensional vector of weights θ. Because we will simulate evolution of n assets, and we want to take into account all possible relations between them as well as some degree of independence for each one, we'll need n independent Wiener processes W_t^j. Let us denote the vector of this Wiener processes as $\bar{W}_t$. We will model the development of value of the capital invested in each (i-th) asset by exponential Brownian motion:

$$dY_t^i = (r + \mu_i)Y_t^i dt + Y_t^i \bar{\Sigma}_i d\bar{W}_t \tag{2}$$

Row vector $\bar{\Sigma}_i$ determines how much each Brownian motion affects value of the asset, r is risk-free interest rate and μ_i determines how much higher is the drift of the assets from risk-free rate. Because weighted sum of independent Wiener processes is a Wiener process again, this equation can be written even simpler:

$$dY_t^i = (r + \mu_i)Y_t^i dt + Y_t^i \sigma_i dW_t \tag{3}$$

where W_t is one-dimensional Wiener process and $\sigma_i = (\bar{\Sigma}_i^T \bar{\Sigma}_i)^{1/2}$ is the volatility. Let us note, that we use exponential Brownian motion instead of classic one, to achieve that the magnitude of perturbations in the value of the capital is proportional to the actual amount of the capital. In other words, if we invest in an asset 10 times more money, we should expect also 10 times bigger gains or losses by the movement of the asset value. Let's state also the equation for the evolution of capital Y^r invested in risk-free asset (bank account) with risk-free interest rate r:

$$dY_t^r = rY_t^r dt \tag{4}$$

Let the total value of our capital be equal to Y. We put to the i-th asset an amount of $Y_i = \theta_i Y$ money, and the remaining capital ($Y^r = (1 - \sum_{i=1}^n \theta_i)Y$) will go to bank account with risk-free interest rate, or, in the case $\sum_{i=1}^n \theta_i > 1$ we will borrow missing amount of money with this risk-free interest rate. Then, evolution of the value of our portfolio will follow the equation, which we get by summing equations of the form (2) for each asset with the equation (4) for investment with risk-free interest rate.

$$\begin{aligned} dY_t &= dY_t^r + \sum_{i=1}^n dY_t^i = rY_t^r dt + \sum_{i=1}^n ((r + \mu_i)Y_t^i dt + Y_t^i \bar{\Sigma}_i d\bar{W}_t) \\ &= rY dt + \sum_{i=1}^n (\mu_i \theta_i Y dt + \theta_i \bar{\Sigma}_i Y d\bar{W}_t) = (r + \mu^T \theta) Y dt + \theta^T \bar{\Sigma} Y d\bar{W}_t \end{aligned} \tag{5}$$

where μ is the vector of differences between drift of each asset and risk-free rate r, and $\bar{\Sigma}$ is the matrix composed of rows $\bar{\Sigma}_i$. Let us note, that the investor can change weights of assets θ_i in time, so θ_i is also function of time. However, to avoid confusement we will not use any new indices to emphasize this fact. Sometimes θ_t will be used to denote vector of functions of time θ_i. As in the case of one asset, also in the case of whole portfolio we can model the evolution of its value by one one-dimensional Wiener process:

$$dY_t = (r + \mu^T \theta) Y dt + \sigma(\theta) Y dW_t = (r + \mu^T \theta) Y dt + (\theta^T \Sigma \theta)^{1/2} Y dW_t \tag{6}$$

where $\Sigma = \bar{\Sigma}\bar{\Sigma}^T$. Let us consider a constant inflow of ε money per time unit to our portfolio. In such case, the equation for the evolution of portfolio value will change slightly:

$$dY_t = (\varepsilon + (r + \mu^T \theta) Y) dt + (\theta^T \Sigma \theta)^{1/2} Y dW_t \tag{7}$$

Now, we will present a theorem that will help us transform this equation into a more feasible form.

Theorem 1.2 (Ito Lemma)**.** *Let Y_t be an Ito process:*

$$dY_t = u(t, \omega) dt + v(t, \omega) dW_t$$

Let $g(t, y) \in C^2([0, \infty) \times \mathbb{R})$. Then $X_t = g(t, Y_t)$ is also an Ito process, and it holds

$$dX_t = \frac{\partial g}{\partial t}(t, Y_t) dt + \frac{\partial g}{\partial y}(t, Y_t) dY_t + \frac{1}{2}\frac{\partial^2 g}{\partial y^2}(t, Y_t) v^2 dt \tag{8}$$

Proof of the theorem can be found for example in [19].

Let us substitute function $g(y, t)$ from Ito Lemma by natural logarithm $\ln(y)$. For Y_t, we substitute process defined by (7). Then, using Ito Lemma to express $dX_t = d\ln(Y_t)$, we get

$$dX_t = (\varepsilon e^{-X_t} + r + \mu^T \theta - \frac{1}{2}\theta^T \Sigma \theta) dt + (\theta^T \Sigma \theta)^{1/2} dW_t \tag{9}$$

This process describes movement of the logarithm of portfolio value and it has the form of Brownian motion. From now on, we will work with this transformed process X_t.

1.3 Deriving Hamilton-Jacobi-Bellman equation

In the previous part, we described evolution of portfolio value with respect to parameter θ. This parameter θ, representing vector of proportions of each asset in portfolio is chosen in every moment by investor. We assume that the only informations influencing investor's choice are the actual portfolio value, and the time in which choice is done. Therefore, we can write θ as a vector function $\theta \equiv \theta(x,t) : \mathbb{R} \times [0,T) \to \Theta \subseteq \mathbb{R}^n$ where Θ is some set of feasible choices (not all choices must be feasible -we may impose restrictions, for example prohibition of short positions). Now, how can we find out this function $\theta(x,t)$? It clearly depends on investor's preferences. Those are mostly represented by an utility function. In our case we will consider utility function $U(X_T)$ of the logarithm of portfolio value at final time T. At this point, let us note some important differences of our utility function from common utility functions:

- Mostly, utility functions are defined on $[0,\infty)$, because portfolio cannot reach negative value. However, argument of our utility function is logarithm of portfolio value which has values in $\mathbb{R}$, therefore our utility function should be defined on $\mathbb{R}$.
- Mostly utility functions are concave, to reflect investors risk-averse attitude to risky profit. Since now argument of utility function is logarithm of profit instead of profit, we will demand that the function $U(\ln(x))$ is concave and not necessarily $U(x)$.
- Our utility function should be increasing as any other utility functions, because greater logarithm of portfolio value is better, than smaller logarithm of portfolio value.

Now, the investor simply chooses optimal $\hat{\theta}(x,t)$ as

$$\hat{\theta}(x,t) = \arg\max_{\theta(x,t)} \mathbb{E}(U(X_T)) \tag{10}$$

It should be clear, that we don't maximize over all feasible vectors θ, but over all feasible vector functions $\theta(x,t)$, so the resulting argument will be also an vector function. We should also note that the whole process X_t and therefore also its final value X_T are influenced by the choice of $\theta(x,t)$, as can be seen in (9). Now, the optimal choice of portfolio weights in time t in case of $X_t = x$ is function $\hat{\theta}$ evaluated in (x,t). Another interesting function is

$$V(x,t) = \max_{\theta|_{[t,T)}} \mathbb{E}(U(X_T)|X_t = x) \tag{11}$$

where $\theta|_{[t,T)}$ is $\theta(x,t)$ constrained to interval $[t,T)$. This function is called **Value function** and it equips us with the possibility to compare portfolio values in different times. Therefore, it is some kind of very natural generalization of utility function from x-domain to whole x-time-domain. At the final time T, it is exactly utility function $U(x)$. Now, we will present a derivation of an partial differential equation which will allow us to compute this value function $V(x,t)$ from the terminal condition $V(x,T) = U(x)$. The derivation can be found in [17], slightly different approach is used in [10].

$$\begin{aligned}
V(x,t) &= \max_{\theta|_{[t,T)}} \mathbb{E}(U(X_T)|X_t = x) \\
&= \max_{\theta|_{[t,t+dt)}} \mathbb{E}(\max_{\theta|_{[t+dt,T)}} \mathbb{E}(U(X_T)|X_{t+dt} = X_{t+dt})|X_t = x) \\
&= \max_{\theta|_{[t,t+dt)}} \mathbb{E}(V(X_{t+dt}, t+dt)|X_t = x) \qquad (12)
\end{aligned}$$

where the first equality is consequence of Bellman principle of optimality which in our situation states, that if function $\theta(x,t)$ is optimal solution of the portfolio optimization problem on time interval $[t,T]$, then $\theta(x,t)|_{[t+dt,T]}$ will be also optimal solution for portfolio optimization problem on time interval $[t+dt,T]$. Bellman principle introduced by American mathematician Richard Bellman (see [2]) is one of the cornerstones of optimal control theory. For deeper understanding of this topic we reffer to [5], [6]. Now we employ Ito Lemma (Theorem 1.2) to compute $dV_t = dV(X_t,t)$:

$$\begin{aligned}
dV_t &= \frac{\partial V}{\partial t}dt + \frac{\partial V}{\partial x}dX_t + \frac{1}{2}\frac{\partial^2 V}{\partial x^2}\theta^T\Sigma\theta dt \\
&= [\frac{\partial V}{\partial t} + \frac{\partial V}{\partial x}(\varepsilon e^{-X_t} + r + \mu^T\theta - \frac{1}{2}\theta^T\Sigma\theta) + \frac{1}{2}\theta^T\Sigma\theta\frac{\partial^2 V}{\partial x^2}]dt \\
&\quad - \frac{\partial V}{\partial x}(\theta^T\Sigma\theta)^{1/2}dW_t \qquad (13)
\end{aligned}$$

where we used X_t as derived in (9). The expression $V(X_t,t) + dV(X_t,t)$ is nothing else than Euler-Maruyama approximation (see [22]) of $V(X_{t+dt}, t+dt)$ which is weakly convergent with order $O(dt)$. Therefore, we can substitute $\mathbb{E}(V(X_{t+dt}, t+dt)|X_t = x)$ by $\mathbb{E}(V(X_t,t) + dV(X_t,t)|X_t = x) + O(dt)$ in (12) where $dV(X_t,t)$ is computed in (13). To simplify the notation, let us denote $A = (\varepsilon e^{-X_t} + r + \mu^T\theta - \frac{1}{2}\theta^T\Sigma\theta)$ and $B = \theta^T\Sigma\theta$:

$$\begin{aligned}
V(x,t) &= \max_{\theta|_{[t,t+dt)}} \{\mathbb{E}(V(X_t,t) + dV(X_t,t)|X_t = x) + O(dt)\} \\
&= \max_{\theta|_{[t,t+dt)}} \{\mathbb{E}(V(X_t,t) + [\frac{\partial V}{\partial t} + A\frac{\partial V}{\partial x} + \frac{1}{2}B\frac{\partial^2 V}{\partial x^2}]dt - B^{1/2}\frac{\partial V}{\partial x}dW_t|X_t = x) + O(dt)\}
\end{aligned}$$

where all derivatives of V are evaluated in (X_t,t) Since $X_t = x$, $V(X_t,t) + [\frac{\partial V}{\partial t} + A\frac{\partial V}{\partial x} + \frac{1}{2}B\frac{\partial^2 V}{\partial x^2}]dt$ is deterministic and we can put it out of the expectation. Moreover, since $V(x,t) + \frac{\partial V}{\partial t}$ is independent of θ, we can put it also out of maximum:

$$V(x,t) = V(x,t) + \frac{\partial V}{\partial t}dt + \max_{\theta|_{[t,t+dt)}} \{(A\frac{\partial V}{\partial x} + \frac{1}{2}B\frac{\partial^2 V}{\partial x^2})dt - \mathbb{E}(B^{1/2}\frac{\partial V}{\partial x}dW_t|X_t = x) + O(dt)\}$$

The only stochastic part of the expression inside the expectation is increment of Wiener process, dW_t, which has zero-expectation. Therefore, the whole expectation will be $\mathbb{E}(B^{1/2}\frac{\partial V}{\partial x}dW_t|X_t = x)) = 0$. Finally, we subtract $V(x,t)$ on each side and get

$$0 = \frac{\partial V}{\partial t}dt + \max_{\theta|_{[t,t+dt)}} \{(A\frac{\partial V}{\partial x} + \frac{1}{2}B\frac{\partial^2 V}{\partial x^2})dt + O(dt)\}$$

Now if we divide both sides by dt and then run with $dt \to 0$ the term $O(dt)/dt$ will approach zero, and we finally end up with an equation

$$0 = \frac{\partial V}{\partial t} + \max_{\theta_t}\{(A\frac{\partial V}{\partial x} + \frac{1}{2}B\frac{\partial^2 V}{\partial x^2})\}$$

what is after substituting back for A, B,

$$\frac{\partial V}{\partial t} + \max_{\theta_t}\{(\varepsilon e^{-x} + r + \mu^T\theta - \frac{1}{2}\theta^T\Sigma\theta)\frac{\partial V}{\partial x} + \frac{1}{2}\theta^T\Sigma\theta\frac{\partial^2 V}{\partial x^2}\} = 0 \tag{14}$$

This partial differential equation is named **Hamilton-Jacobi-Bellman Equation** and we can use it, together with terminal condition in the form

$$V(x,T) = U(x) \tag{15}$$

where $U(x)$ is the utility function at final time, to find the value function $V(x,t)$. However, closed-form analytical solution is rarely feasible. Therefore, solving this equation numerically is the main goal of this work. Moreover, let us note that

$$\hat{\theta}(x,t) = \arg\max_{\theta_t}\{(\varepsilon e^{-x} + r + \mu^T\theta - \frac{1}{2}\theta^T\Sigma\theta)\frac{\partial V(x,t)}{\partial x} + \frac{1}{2}\theta^T\Sigma\theta\frac{\partial^2 V(x,t)}{\partial x^2}\} \tag{16}$$

is optimal choice of portfolio weights in time t if the portfolio value in time t is e^x. Therefore, knowing solution to Hamilton-Jacobi-Bellman provides us not only with the possibility to compare different portfolios in different times, but also to compute optimal portfolio composition in any time easily. Let us note, that such portfolio composition is optimal only under the assumption, that the investor rearranges the portfolio in every moment (continuously). To show all this results rigorously (see[17], [10]), deeper stochastic optimal control theory would be needed. However, the derivation above provides us with the very idea what the value function $V(x,t)$ is, and why the Hamilton-Jacobi-Bellman equation holds.

2 Transformations of HJB equation and related functions

2.1 Other forms of HJB equation

In Chapter 1, we derived HJB equation in the form

$$\frac{\partial V}{\partial t} + \max_{\theta_t}\{(\varepsilon e^{-x} + r + \mu^T\theta - \frac{1}{2}\theta^T\Sigma\theta)\frac{\partial V}{\partial x} + \frac{1}{2}\theta^T\Sigma\theta\frac{\partial^2 V}{\partial x^2}\} = 0 \tag{17}$$

However, to solve this equation numerically, it'll be helpful to write it in some other form depending on the method used. Numerical methods proposed in [9] go even further, and use Riccati transformation of the original equation. In all forms that will be proposed, we will use substitution

$$\varphi = 1 - \frac{\partial_x^2 V}{\partial_x V} \tag{18}$$

and abbreviations $\partial_t V = \frac{\partial V}{\partial t}$, $\partial_x V = \frac{\partial V}{\partial x}$, $\partial_x^2 V = \frac{\partial^2 V}{\partial x^2}$.

First order PDE form

Using substitution

$$\alpha(\varphi) = \min_{\theta\in\Theta}(-\mu^T\theta + \frac{\varphi}{2}\theta^T\Sigma\theta) \tag{19}$$

we get HJB equation in the form

$$\partial_t V + [\varepsilon e^{-x} + r - \alpha(\varphi)]\partial_x V = 0 \tag{20}$$

If we ignore the fact, that $\alpha(\varphi)$ is also function of derivatives of V, this seems to be regular 1. order PDE (Transport equation). Later, we will also try to solve HJB equation in this form with methods for 1. order PDE's. Moreover, since minimization problem in (19) is maximization problem from (16) multiplied and subtracted by functions which don't depend on argument θ, its solution

$$\hat{\theta} = \arg\min_{\theta\in\Theta}(-\mu^T\theta + \frac{\varphi}{2}\theta^T\Sigma\theta) \tag{21}$$

will be optimal choice of portfolio weights in (x,t). We can see, that optimal portfolio weights depends only on φ defined in (18). We can say even more about the dependence between φ and the optimal weights $\hat{\theta}$. If φ is large, the volatility term $\frac{\varphi}{2}\theta^T\Sigma\theta$ plays an important role in minimized expression and so we will concentrate more on its minimization (by putting bigger weights on assets with small volatility, or by avoiding large positive correlations). On the other hand, if φ is small, then $-\mu^T\theta$ is more important and for minimizing the whole expression it'll be essential to put big weights to assets with large return μ.

Parabolic PDE form

Using substitutions

$$\beta(\varphi) = \mu^T\hat{\theta} - \hat{\theta}^T\Sigma\hat{\theta} \tag{22}$$
$$\gamma(\varphi) = \hat{\theta}^T\Sigma\hat{\theta} \tag{23}$$

where $\hat{\theta} = \arg\min_{\theta\in\Theta}(-\mu^T\theta + \frac{\varphi}{2}\theta^T\Sigma\theta)$, we can write our HJB equation in the form

$$\partial_t V + [\varepsilon e^{-x} + r + \beta(\varphi)]\partial_x V + \gamma(\varphi)\partial_x^2 V = 0 \tag{24}$$

Again, ignoring the fact that φ is a function of derivatives of V, we can see HJB equation in this form as 2. order parabolic PDE, and try to handle it numerically as such.

Ricatti Transformation

Now, let us think about the HJB equation in the form (20). The next Theorem, which is formulated and proved in [9] provides us with another equation which can be solved instead of the original HJB equation:

Theorem 2.1 (Ricatti Transformation)**.** *Let the function V satisfy (20) for φ defined as in (18). Then, φ is a solution to the Cauchy problem for the quasi-linear parabolic equation:*

$$\partial_t\varphi + \partial_x^2\alpha(\varphi) + \partial_x[(\varepsilon e^{-x} + r)\varphi + (1-\varphi)\alpha(\varphi)] = 0 \tag{25}$$
$$\varphi(x,T) = U''(x)/U'(x), \qquad x \in \mathbb{R}$$

As already mentioned, optimal portfolio weights $\hat{\theta}$ depend just on φ (as solution of the problem (21)), so we will be able to compute them without knowing $V(x,t)$, if we solve this transformed equation. However, since it holds

$$\varphi = 1 - \frac{\partial}{\partial x}(\ln(\frac{\partial V}{\partial x})) \tag{26}$$

V can be formaly computed as

$$V = \int \exp(\int (1-\varphi)) \tag{27}$$

Constants which will appear after integration are just multiplicative and additive and don't affect the portfolio choice implied by the value function V.

2.2 Analysis of quadratic programming problem

In this part we focus our attention on the analysis of the function

$$\alpha(\varphi) = \min_{\theta\in\Theta}(-\mu^T\theta + \frac{\varphi}{2}\theta^T\Sigma\theta) \tag{28}$$

Let us note, that this quadratic programming problem has to be solved also in the case, when Parabolic PDE form of HJB equation and functions $\beta(\varphi)$ and $\gamma(\varphi)$ are used. Several interesting properties of the function $\alpha(\varphi)$ are proved in the article [9] for the case of standard simplex Θ. Economically, this requirement means that the sum of weights of assets is normalized to 1, and we prohibit short positions (negative weights). Now we will try to generalize this properties for any

$$\Theta = \{\theta | A\theta \le a\} \tag{29}$$

where A is $m \times n$ matrix and a is an m-dimensional vector. Let us note that any set of the form $\{\theta | A\theta \le a, B\theta = b, \theta_i \ge 0, \forall i \in I\}$ where I is some index set can be rewritten into the form (29). Therefore, the system of such sets Θ is large enough (standard simplex is also contained), what gives us many possibilities in formulating restrictions on θ. Examples of such restrictions are forced diversification or Merton model in which sum of weights can be less than one.

Theorem 2.2 (Properties of alpha). *Let Σ be positive definite matrix and $\mu \in \mathbb{R}^n$. Then $\alpha(\varphi)$ defined by (28), (29) is non-decreasing C^1 continuous function. Moreover, $\alpha'(\varphi) = \frac{1}{2}\hat{\theta}^T\Sigma\hat{\theta}$ where $\hat{\theta}$ is unique minimizer of (28), (29) for $\varphi > 0$, and the function $\varphi \to \hat{\theta}(\varphi)$ is locally Lipschitz continuous for $\varphi > 0$.*

The theorem comes from [9] where it is formulated for Θ standard simplex. The only change needed by generalizing to Θ defined by (29) is, that $\alpha(\varphi)$ is not necessarily increasing, just non-decreasing. Small change should be done also in the proof. In our case, continuity of the function $\varphi \to \hat{\theta}(\varphi)$ is consequence of properties of strictly convex functions with minimum in $\mathbb{R}^n$, minimized on convex closed (but not necessarily bounded) set. For complete proof, we refer to [9]. The next lemma will be employed by proving further interesting properties of $\alpha(\varphi)$:

Lemma 2.3 (Generalized Cauchy-Schwartz inequality). *Let $w \in \mathbb{R}^n$, and X is $m \times n$, $m \le n$ matrix of rank of m. Then, it holds*

$$w^T w \ge w^T X (X^T X)^{-1} X^T w$$

Proof.

$$\begin{aligned}
&w^T w - w^T X (X^T X)^{-1} X^T w = w^T (I - X(X^T X)^{-1} X^T) w \\
&= w^T (I - X(X^T X)^{-1} X^T)^T (I - X(X^T X)^{-1} X^T) w \\
&= w^T P_{NX}^T P_{NX} w = (P_{NX} w)^T (P_{NX} w) \ge 0
\end{aligned}$$

where $P_{NX} = I - X(X^T X)^{-1} X^T$. □

Let us note, that the matrix $P_X = X(X^T X)^{-1} X^T$ is nothing else then projection matrix on the row space of the matrix X. Therefore, our inequality can be rewritten as $w^T w \ge w^T P_X w = (P_X w)^T (P_X w)$ and it reflects an geometrically intuitive fact: the length of an vector is always greater or equal to the length of its projection on some subspace. Similarly, the matrix P_{NX} is projection matrix on the orthogonal complement to row space of X, and the proof is done just by using Pythagorean theorem. If we choose X of the dimension $1 \times n$ we get classic Cauchy-Schwartz inequality.
Next theorem which is again generalization of an theorem from [9], will tell us more about the form of the function $\alpha(\varphi)$. Suppose, that the matrix A from (29) has m

rows. Then, the set Θ is defined by m inequalities. Without loss of generality, assume that there are not two up to multiplication by constant same inequalities among them. Denote by A_i i-th row of matrix A. Let M be any subset of $\{1, 2, ...m\}$. We define

$$I_M = \{\varphi > 0 | A_i\hat{\theta}(\varphi) = a \Leftrightarrow i \in M\} \tag{30}$$

Theorem 2.4 (Form of function alpha)**.** *For each I_M and $\varphi \in I_M$ it holds $\alpha(\varphi) = a_M\varphi - b_M/\varphi + c_M$ and $\hat{\theta}(\varphi) = u_M - v_M/\varphi$, where $a_M \geq 0$, $b_M \geq 0$, c_M, are constants and u_M, v_M are constant vectors depending on the interval I_M.*

Proof. Let $\varphi \in I_M$. Let us write the Lagrange function to the optimization problem (28),(29):

$$L(\theta, \lambda) = -\mu^T\theta + \frac{\varphi}{2}\theta^T\Sigma\theta + \lambda^T(A\theta - a) \tag{31}$$

Then, corresponding Kuhn-Tucker conditions (see [13]) are:

$$(\frac{\partial L}{\partial \hat{\theta}})^T = -\mu + \varphi\Sigma\hat{\theta} + A^T\lambda = 0 \tag{32}$$

$$(\frac{\partial L}{\partial \lambda})^T = A\hat{\theta} - a \leq 0 \tag{33}$$

$$\lambda^T(\frac{\partial L}{\partial \lambda})^T = \lambda^T(A\hat{\theta} - a) = 0 \tag{34}$$

$$\lambda \geq 0 \tag{35}$$

Since $\varphi \in I_M$, inequalities (33) corresponding to indices $i \in M$ are fulfilled in the form of equality. On the other hand, for indices $i \notin M$ are inequalities strict, and therefore necessarily $\lambda_i = 0$, $\forall i \notin M$, to satisfy condition (34). Now let us remove from A those rows for which holds $A_i\hat{\theta} < a_i$, and denote this new matrix as $\tilde{A}$. Note that the rows of $\tilde{A}$ are linearly independent (just one of two inequalities between the same linear combination and different constant can be fulfilled as equality), therefore the rank of $\tilde{A}$ is equal to the number of rows. Let us denote the vector of corresponding Lagrange multipliers by $\tilde{\lambda}$. Components of λ which we removed to get $\tilde{\lambda}$ were exactly those zero-valued. Therefore, we can rewrite the equality (32) as $-\mu + \varphi\Sigma\hat{\theta} + \tilde{A}^T\tilde{\lambda} = 0$. Then, it holds $\hat{\theta} = \frac{1}{\varphi}\Sigma^{-1}(\mu - \tilde{A}^T\tilde{\lambda})$. After substituting to $\tilde{A}\hat{\theta} = a$ we get $\tilde{\lambda} = (\tilde{A}\Sigma^{-1}\tilde{A}^T)^{-1}(\varphi a - \tilde{A}\Sigma^{-1}\mu)$. Substituting to equation for $\hat{\theta}$ and using definition $\alpha(\varphi) = -\mu^T\hat{\theta} + \frac{\varphi}{2}\hat{\theta}^T\Sigma\hat{\theta}$ we get statement of the theorem for constants:

$$u_M = \Sigma^{-1}A^T(A\Sigma^{-1}A^T)^{-1}a \tag{36}$$

$$v_M = \Sigma^{-1}A^T(A\Sigma^{-1}A^T)^{-1}A\Sigma^{-1}\mu - \Sigma^{-1}\mu \tag{37}$$

$$a_M = \frac{1}{2}a^T(A\Sigma^{-1}A^T)^{-1}a \tag{38}$$

$$b_M = \frac{1}{2}(\mu^T\Sigma^{-1}\mu - \mu^T\Sigma^{-1}A^T(A\Sigma^{-1}A^T)^{-1}A\Sigma^{-1}\mu) \tag{39}$$

$$c_M = -a^T(A\Sigma^{-1}A^T)^{-1}A\Sigma^{-1}\mu \tag{40}$$

Inequality $a_M > 0$ results from the fact that Σ^{-1}, and therefore also $(A\Sigma^{-1}A^T)$ and $(A\Sigma^{-1}A^T)^{-1}$ are positive definite. Inequality $b_M > 0$ follows after using Lemma 2.3 for $w = \Sigma^{-1/2}\mu$, $X = \Sigma^{-1/2}A^T$. Let us note, that the condition (35) which results in

$\varphi a - \tilde{A}\Sigma^{-1}\mu \geq 0$ should be also satisfied, and also then $\hat{\theta}$ will be only a candidate on the solution. However, since for a convex optimization problem are (according to [7]) Kuhn-Tucker conditions sufficient, and for linear constraints necessary (all feasible points are Rg5-regular, see [7]), if the solution exists and $\varphi \in I_M$ this condition will be satisfied and the candidate will be the solution. □

Corollary 2.5. *Function $\alpha(\varphi)$ is C^∞ smooth on $\bigcup_{\forall M} int(I_M)$.*

2.3 Numerical examination of properties

Basically, we have two possible ways how to get functions $\alpha(\varphi), \beta(\varphi), \gamma(\varphi)$. The first option is to compute these functions in some points and then get all other points by interpolation. Disadvantage of this approach is, that we get $\alpha(\varphi)$ just on some interval. Second approach is based on analysis of quadratic programming problem from part 2.2. We should at first find sets I_M for those just certain constraints of Θ are active and then, following Theorem 2.4, we can express $\alpha(\varphi)$, $\hat{\theta}$, and $\beta(\varphi), \gamma(\varphi)$ inside this sets exactly. However, here we will focus on the first approach, because it can be programmed easily and doesn't need searching for I_M sets from case to case.
To compute $\alpha(\varphi), \beta(\varphi), \gamma(\varphi)$ on some interval J in N points with constant distance between them we implemented function `compute_alfa_points`. Core of this function is Octave routine `qp` designed for solving quadratic programming problem. Outputs are vectors of α, β, γ-values, as well as optimal weights $\hat{\theta}$ in given φ's. Now, if we want to compute $\alpha(\varphi), \beta(\varphi), \gamma(\varphi)$ for arbitrary $\varphi \in J$ we should use some interpolation technique. For interpolation by linear spline we implemented function `alfa_method1`. Function `alfa_method2` is based on interpolation by cubic Hermite spline. This spline connects values in each two neighboring nodes with an cubic polynomial, in such manner, that the resulting function is continuous, with continuous first derivatives. This is exactly the property demanded for function $\alpha(\varphi)$. However, since we do not know derivatives in nodes, we approximate them with central finite differences. The resulting function will for each $\varphi = \varphi_i + qh$ be:

$$\begin{aligned} \alpha(\varphi = \varphi_i + qh) &= (2q^3 - 3q^2 + 1)\alpha(\varphi_i) + (-2q^3 + 3q^2)\alpha(\varphi_{i+1}) \\ &+ h(q^3 - 2q^2 + q)d\alpha_i + h(q^3 - q^2)d\alpha_{i+1} \end{aligned} \tag{41}$$

where φ_i is the first value left from φ, for which we have computed $\alpha(\varphi_i)$, h is the distance between φ_i and φ_{i+1} (first value right from φ for which we know $\alpha(\varphi)$), $q \in [0, 1]$, $d\alpha_i = (\alpha(\varphi_{i+1}) - \alpha(\varphi_{i-1}))/(2h)$ and $d\alpha_{i+1} = (\alpha(\varphi_{i+2}) - \alpha(\varphi_i))/(2h)$.

Inverse of function alpha

In the later chapters we will compute so-called traveling wave solution, for a very specific terminal condition. We will do that almost analytically, however the inverse function to function α will be needed. We will use interpolation to approximate it again.

- First, we implemented function `invalfa_points`. This function gets equidistant φ-points from interval J, and their α-values on input. The range of these α-values is some other interval I. We divide this interval by N equidistant points, and get

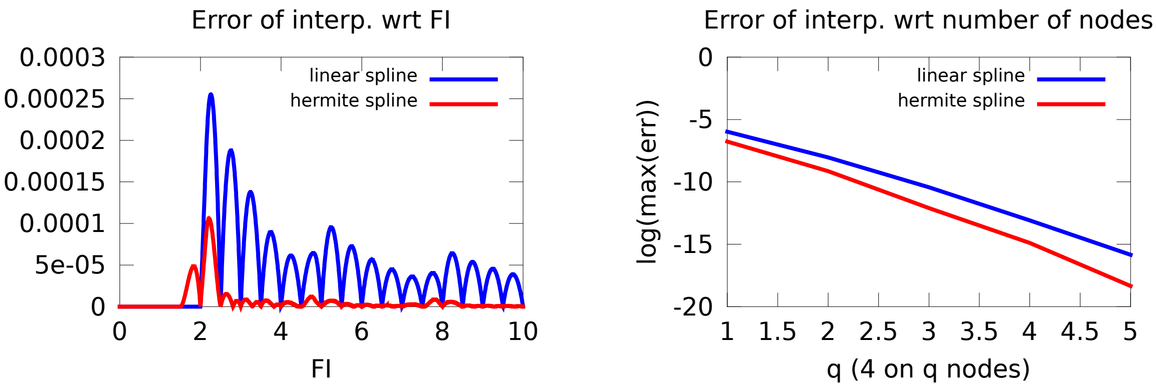

Figure 1: Comparison of interpolation methods

their pre-images by using linear spline interpolation on related φ-values. This pre-images are values of inverse α-function in N equidistant points from interval I

- We can get value of inverse α-function in any point from interval I by interpolation. To achieve this, we again use our functions `alfa_method1`, `alfa_method2`.

Application on our data

Now we will illustrate the results of analysis from part 2.2 on our data, which consist of expected returns and covariance matrix of 20 Swiss assets from the Swiss stock index SMI. How are these computed, will be discussed in Chapter 5.1 in detail. We set natural constraints: weights of all stocks should sum up to 1 and we prohibit short positions (weights must be non-negative).

Comparison of interpolation techniques

Let us compare our two interpolation methods for interpolation of function $\alpha(\varphi)$. First picture in Figure 1 illustrates error of interpolation with respect to φ from interval $[0, 10]$ for 30 interpolation nodes for `alfa_method1` (linear spline) and `alfa_method2` (hermite spline). Second picture in Figure 1 illustrates logarithm of maximum error of interpolation on interval $[0, 10]$ with respect to parameter q, where 4^q is the number of interpolation nodes. We can see, that for higher number of interpolation nodes (which would be also our case), Hermite spline interpolation tends to be several-times better than linear spline interpolation, therefore we will use only `alfa_method2` in the future.

Function alpha and its derivatives

Figure 2 illustrates function $\alpha(\varphi)$, and Figure 3 illustrates its first and second derivative. We see, that we get similar result as in article [9], which is also consistent with Theorem 2.2 . Function alpha is C^1-continuous, however its first derivative is not differentiable everywhere, and therefore its second derivative is not even continuous.

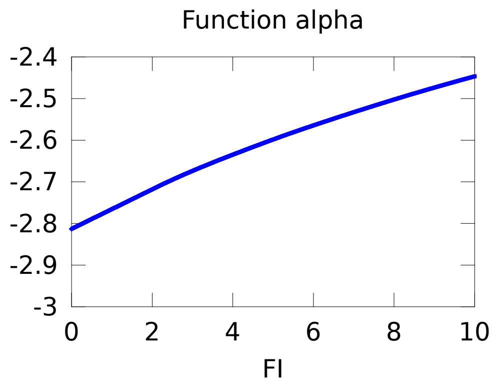

Figure 2: Graph of function $\alpha(\varphi)$

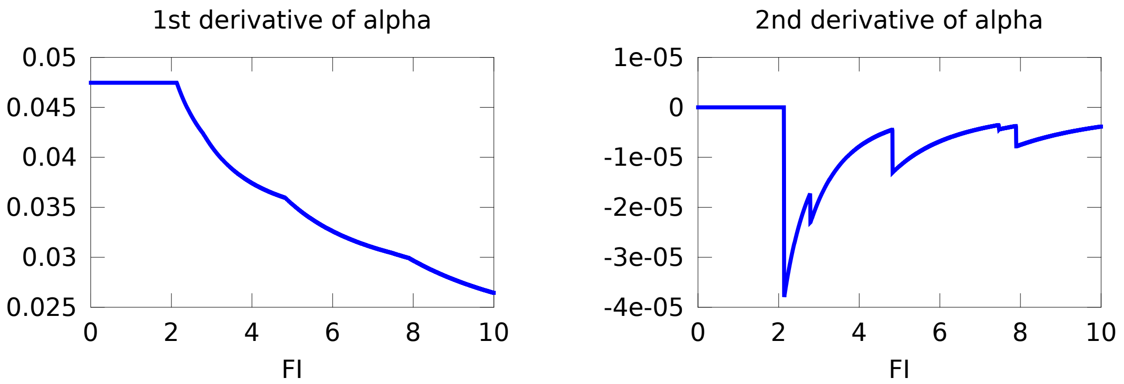

Figure 3: Derivatives of function $\alpha(\varphi)$

Functions beta and gamma

Figure 4 illustrates the functions $\beta(\varphi)$ and $\gamma(\varphi)$. They seem to be continuous but not differentiable in contrast to function $\alpha(\varphi)$. The continuity is trivial consequence of continuity of $\hat{\theta}$ proven in Theorem 2.2. The difference between $\alpha(\varphi)$ and functions $\beta(\varphi), \gamma(\varphi)$ lies in Envelope Theorem which is used to prove differentiability of $\alpha(\varphi)$ in [9], but cannot be used in the case of $\beta(\varphi), \gamma(\varphi)$.

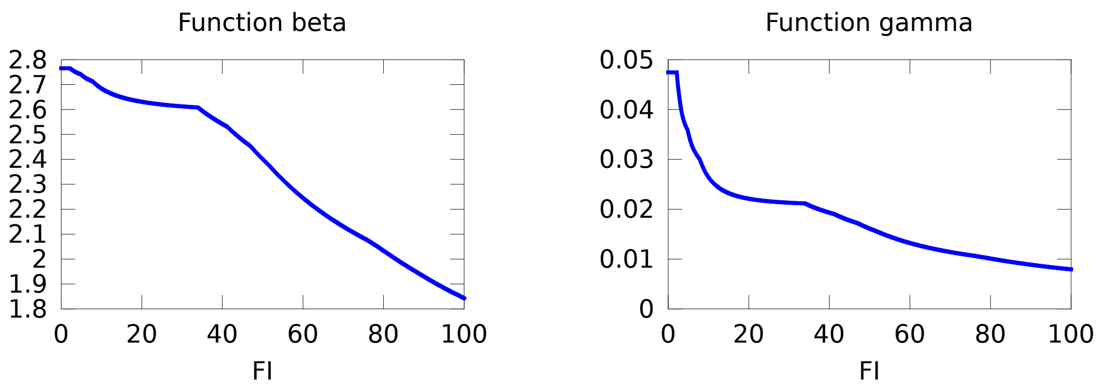

Figure 4: Functions $\beta(\varphi)$ and $\gamma(\varphi)$

Active Assets

Using the theory from Theorem 2.4, we can deduce from the graph of second derivative of function $\alpha(\varphi)$, that with respect to $\varphi \in [0, 10]$, six different sets of active assets (assets with non-zero weight) may occur. This confirms also first part of Figure 5, in which we plot optimal portfolio composition with respect to $\varphi \in [0, 10]$. The width of each colored stripe for some φ indicates the proportion of the asset in the portfolio for this φ. We see six different assets, each entering the portfolio on different level of φ. With $\varphi = 0$,only one asset is present in the portfolio, representing a risky choice, but with highest possible return. On the other side, with $\varphi = 10$, the portfolio is composed of six different assets, which represents less risky choice. However if we take into account φ from larger interval, many new assets come into play. Second part of Figure 5 shows portfolio composition for φ from interval $[0, 200]$.

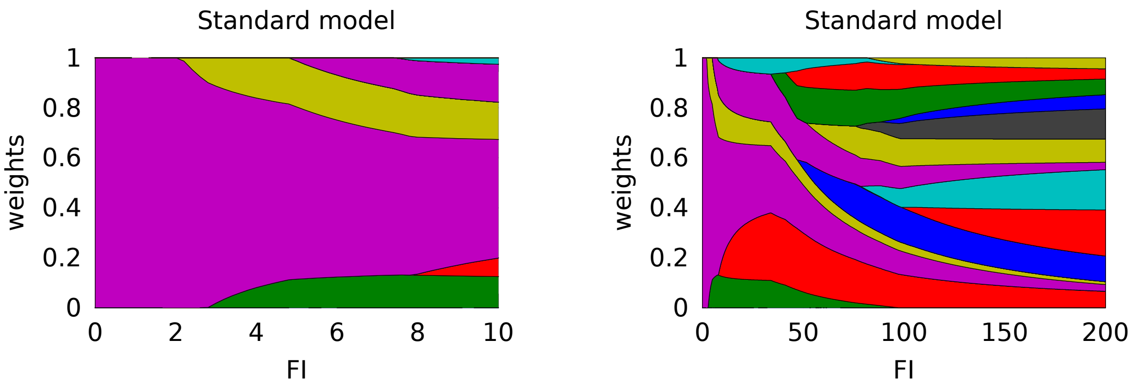

Figure 5: Proportions of assets depending on φ in standard model

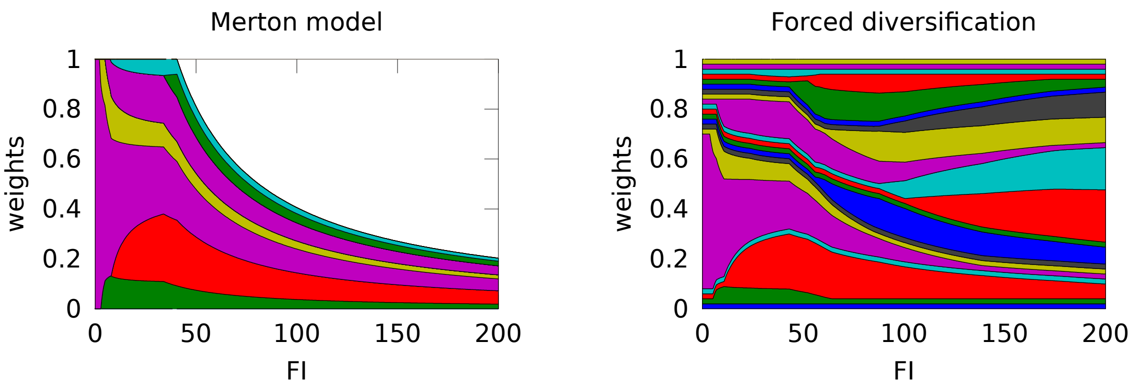

Figure 6: Proportions of assets in portfolio depending on φ in Merton model and by forced diversification

Also constraints are important for portfolio composition. In the first part of Figure 6, we see portfolio composition for Merton model; that means, the sum of weights may not be equal to 1, but can be also less than one (but non-negative). The rest is supposed to be invested into risk-free asset. For $\varphi = 0$, we chose in that case the same risky asset with weight one, however for $\varphi = 200$, sum of weights is already less than

0.3. We should note, that we used $\mu - 1$ instead of μ for better ilustration in this case. Second part of Figure 6 shows portfolio composition for forced diversification model, in which we demand that the investor invests at least 2% of his capital in each stock (and all weights should sum up to one).

On Figures 5, 6 we can observe the behavior that was theoretically examined in Chapter 2.1: for small φ, we concentrate on large return, investing as much as possible to the asset with highest expected return. For large φ, the volatility is more important, and we try to minimize it, either by diversification, putting more money in assets with small volatility, or into the risk-free asset in Merton model. Because of utility functions used in our numerical simulations, we will not need $\varphi > 10$. Therefore, we will later work only with those 6 assets which we identified as relevant for $\varphi \in [0, 10]$, what will speed up our computations.

Computing Fi

Most of our numerical method are designed to solve HJB equation, which has value function $V(x,t)$ as solution. However, since the portfolio composition depends on $\varphi = 1 - \partial_x^2 V / \partial_x V$, it will be useful to transform the solution to the φ-variable. To achieve this, we implemented Octave function `compute_fi` which gets solution in $V(x,t)$ on input and returns solution in $\varphi(x,t)$ as output.

3 Explicit methods and boundary conditions

As we already mentioned analytical solution of HJB Equation is rarely feasible. Therefore, solving HJB equation numerically is the core of this work. Methods examined here can be divided into 3 categories according to form of equation they are based on (see Chapter 2.1): methods for first order PDE form, methods for parabolic PDE form and methods based on Riccati transformation, which are proposed in [9]. Moreover, methods for first order PDE form can be further categorized as explicit or implicit. Methods based on parabolic PDE form or Riccati transformation are implicit (semi-implicit). Another important problem shaping the final formulation of the numerical scheme is, how to pose boundary conditions (BC's). Boundary conditions for methods based on Riccati transformation are already proposed in [9], and BC's for other methods are based on similar reasoning. However, for explicit methods we can avoid using any boundary conditions. Short overview of implemented methods follows:

- Explicit first order PDE form methods (based on [16])
  - Simple explicit method (without boundary conditions) (Chapter 3.1)
  - Modified Godunov method without BC's (Chapter 3.2)
  - Modified Godunov method with BC's (1. approach to BC's) (Chapter 3.4)
  - Modified Godunov method with BC's (valid approach to BC's) (Chapter 3.5)
- Implicit first order PDE form methods
  - with parameter q for implicit, Crank-Nicholson or up to BC's explicit form of the method (Chapter 4.1)
- Implicit parabolic PDE form methods (based on [21])
  - with parameter q for implicit, Crank-Nicholson or up to BC's explicit form of the method (Chapter 4.2)
  - Method with exponential correction for large φ (Chapter 4.3)
- Riccati transformation methods (from [9])
  - Semi-implicit method (Chapter 4.4)
  - Fully implicit method (Chapter 4.4)

These methods can be also categorized as either finite diference methods, or finite volume methods. Modified Godunov methods and methods based on Riccati transformation are finite volume methods, other methods are classified as finite difference methods.

3.1 Simple explicit method

At first, let us suppose $\varepsilon = 0$. We will try to solve HJB equation in the form (20):

$$\partial_t V + [r - \alpha(\varphi)]\partial_x V = 0$$

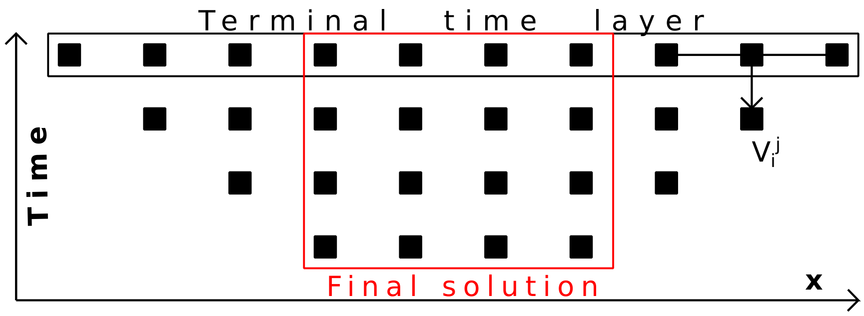

Figure 7: Simple explicit scheme without boundary conditions

with finite difference method. We will use notation V_i^j for numerical value of $V(x_i, t_j)$ where (x_i, t_j) is a point on equidistant time-space-grid. Let us substitute derivatives with finite differences. Time derivative will be substituted by backward difference, and derivatives in x will be substituted by central differences. If we rearrange the equation so that only V_i^{j-1} will be on the left-hand side, and all other terms on the right hand side, the numerical scheme follows:

$$\begin{aligned} V_i^{j-1} &= V_i^j + dt(r - \alpha(\varphi_i^j))d_x V_i^j \\ \varphi_i^j &= 1 - \frac{d_x^2 V_i^j}{d_x V_i^j} \\ d_x V_i^j &= \frac{V_{i+1}^j - V_{i-1}^j}{2dx} \\ d_x^2 V_i^j &= \frac{V_{i+1}^j - 2V_i^j + V_{i-1}^j}{(dx)^2} \end{aligned} \tag{42}$$

where dx is the size of x-step, and dt is the size of time-step. As terminal condition we use $V_i^M = U(x_i)$ where $U(x)$ is an utility function. Setting boundary conditions can be pretty difficult. However, we can overcome this problem by computing the terminal condition on larger domain. Then in every next (in earlier time, because we are running backwards in time) time layer, we will compute just values for which we have complete data, that means, we will not compute the most-left and most-right value. Therefore, if we want to use M time layers (including terminal time layer), and we want to end up with N values computed in last time layer (in time, the earliest one), we should compute the terminal condition for $N + 2M - 2$ values. Figure 7 will make this approach clear.

3.2 Modified Godunov method

Originally, Godunov scheme was developed by russian mathematician Sergei Konstantinovich Godunov in 1959 for modelling conservative systems (see [4]). This topic is deeply examined in [16]. However, we will now try to modify this method to be suitable for our equation, which is not conservative. Again, we suppose $\varepsilon = 0$. Our equation $\partial_t V + [r - \alpha(\varphi)]\partial_x V = 0$ would be conservative, if the expression $[r - \alpha(\varphi)]$ would be constant. Then, we would have simple transport equation $\partial_t V + a\partial_x V = 0$ Solution of this equation is wave traveling right for $a > 0$ and left for $a < 0$ with time.

That means, value of each point "moves" right with speed a (resp. left with speed -a). Let us think about numerical scheme with M time layers. Since we have terminal condition instead of initial, we will observe behavior of the solution backwards in time. Now, we can say that each value in some point x_i and in time layer t_j, moves with some speed to left or right as time approaches previous time layer t_{j-1}. If we plot position of that value with varying time, we get nothing else than contour of the solution. Let us do an simplification: suppose that this value moves between the two time layers with constant speed witch can be computed in the t_j time-layer as $[r-\alpha(\varphi_i^j)]$. The closer the time layers will be the better will be such approximation. Now when we have better insight into our approximation, we can proceed to proposing the numerical scheme. This will be an finite-volume scheme, which means that we will work with some interval averages rather then values in concrete points. This averages will be computed on intervals $[x_{i-1/2}^j, x_{i+1/2}^j]$ for each time layer t_j, and we will call this intervals x_i^j-*cells*. The approximation of average value of $V(x,t)$ in the x_i^j-cell will be denoted as v_i^j. The core of this modified Godunov method is the REA algorithm. Idea of the algorithm is already hidden in its name which is abbreviation of reconstruct-evolve-average. We modify the original REA algorithm for conservative systems from [16], to be suitable for our equation.

REA algorithm

1. Reconstruct a piecewise constant function $v(x,t_j)$ from the cell averages v_i^j, so that the constant value on each interval x_i^j will be v_i^j.

2. Evolve this equation for data $v(x,t_j)$ in t_j-time-layer to obtain $v(x,t_{j-1})$ in the previous time-layer. This in our case means, that we should for each discontinuity in $x_{i+1/2}^j$ compute its speed as $[r-\alpha(\varphi_{i+1/2}^j)]$. Then we are able to determine where this discontinuity is projected in t_{j-1}-time-layer. Between discontinuities function $v(x,t_{j-1})$ stays constant with the same value as in t_j-time-layer.

3. Average the new function $v(x,t_{j-1})$ over each x_i^{j-1}-cell to obtain new cell averages.

This algorithm is repeated everytime we approach from one time layer to the previous one. As starting values we should use averages of utility function $U(x)$ over x_i^M intervals. For simplicity, we approximate the average by value from the middle of the interval $U(x_i)$.
In step 3 of the algorithm, we average the new function over each x_i^{j-1}-interval. That can be easy, if at most two discontinuities propagate into this interval from the t_j-layer, one from the left boundary $x_{i-1/2}$ and one from the right boundary $x_{i+1/2}$. We also demand that the discontinuity from the left boundary in t_j-layer will be also in the t_{j-1}-layer more left than the discontinuity from the right boundary. We can satisfy this requirements by choosing sufficiently small time step:

$$\max_i(|r-\alpha(\varphi_{i+1/2}^j)|)\cdot dt \leq \frac{dx}{2} \tag{43}$$

Under this condition, we can compute the new average v_i^{j-1} by simple formula:

$$v_i^{j-1} = v_i^j - \chi_L \frac{[r-\alpha(\varphi_{i-1/2}^j)]dt}{dx}(v_{i-1}^j - v_i^j) + \chi_R \frac{[r-\alpha(\varphi_{i+1/2}^j)]dt}{dx}(v_{i+1}^j - v_i^j) \tag{44}$$

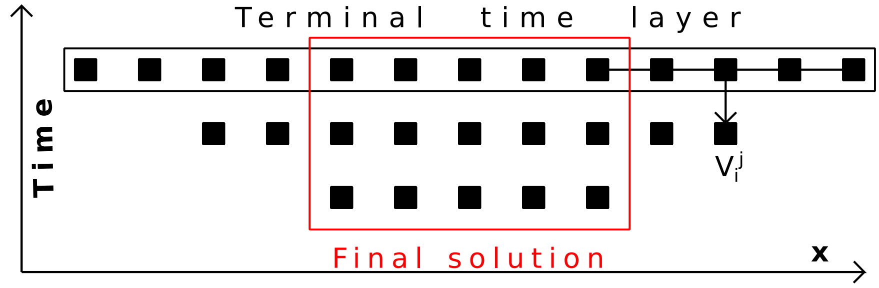

Figure 8: Modified Godunov scheme without boundary conditions

where $\chi_L = 1$ for $[r - \alpha(\varphi^j_{i-1/2})] < 0$ and $\chi_L = 0$ for $[r - \alpha(\varphi^j_{i-1/2})] \geq 0$. Similarly, $\chi_R = 1$ for $[r - \alpha(\varphi^j_{i+1/2})] > 0$ and $\chi_R = 0$ for $[r - \alpha(\varphi^j_{i+1/2})] \leq 0$. χ_L, χ_R are simply identifying if the wave moves inside the cell (with time running backwards) and the minus sign in front of the second term on right-hand-side serves to correct the negative value of $[r - \alpha(\varphi^j_{i-1/2})]$ if the wave flows in. Now the question is, how to compute $\varphi^j_{i-1/2}$, $\varphi^j_{i+1/2}$. We will do so by using central differences:

$$
\begin{aligned}
d_x v^j_{i+1/2} &= \frac{v^j_{i+1} - v^j_i}{dx} \\
d^2_x v^j_{i+1/2} &= \frac{v^j_{i+2} - v^j_{i+1} - v^j_i + v^j_{i-1}}{2dx^2} \\
\varphi^j_{i+1/2} &= 1 - \frac{d^2_x v^j_i}{d_x v^j_i}
\end{aligned}
$$

Next problem we should tackle are boundary conditions. We can overcome it in the same manner as in the case of simple explicit method. That means, in every time layer we will compute only those values, for which we have all data needed. However, because of the formula for approximation of second derivative used here, for M time layers each with at least N values v^j_i, we need to compute $N + 4M - 4$ values in the terminal time-layer. Figure 8 makes this approach clear.

3.3 Modification for money inflow

Let us consider $\varepsilon > 0$. The generalization to this case seems to be straightforward. We should take $[\varepsilon e^{-x} + r - \alpha(\varphi)]$ instead of $[r - \alpha(\varphi)]$ in our numerical schemes. However for large negative values of x this new coefficient may be pretty large which causes numerical instability. Particularly, for Modified Godunov method, we are forced by (43) to use pretty small time step. Therefore, for this methods we have to avoid computing solution for large negative values of x. The problem is, that proposed methods require to work in late time layers on far larger domains which can very probably include also large negative values of x, causing failure of the method. Even using small time step in Modified Godunov method may not help, since then we need more time steps which makes the terminal time layer even larger, which means even more negative values of x.

For simple explicit method we solve this problem instantly by taking forward diferences

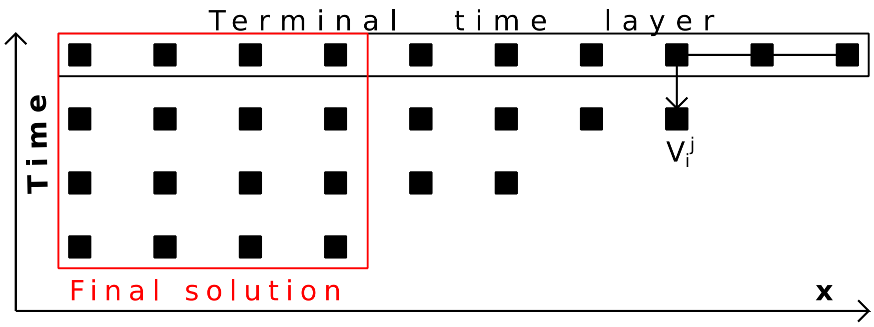

Figure 9: Simple explicit scheme without boundary conditions modified for money inflow

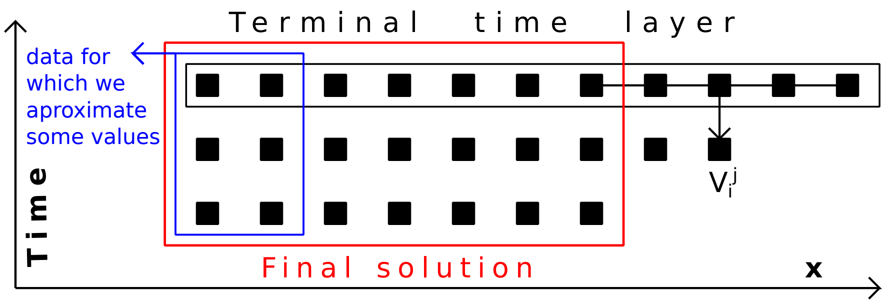

Figure 10: Modified Godunov scheme without boundary conditions modified for money inflow

in x instead of central:

$$d_x V_i^j = \frac{V_{i+1}^j - V_i^j}{dx}$$
$$d_x^2 V_i^j = \frac{V_{i+2}^j - 2V_{i+1}^j + V_i^j}{dx^2}$$

This means, we do not need to compute one extra value on the left boundary for every later time-layer, but we need to compute two extra values on the right boundary for every later time layer instead of one. Figure 9 illustrates this approach.

The case of Modified Godunov method is more delicate, since we need to know cell averages propagating into our computational domain from the left. However, for x negative enough, whole term $[\varepsilon e^{-x} + r - \alpha(\varphi)]$ will be positive on the left boundary, what means that only values from the right propagate into the left-boundary cells. That means, we do not need to know the values behind the left boundary. The trick is just to choose the left boundary left-enough so that the wave will propagate from right, but not too much left, so that the e^{-x} will not be too large. The last problem is, that we don't have enough data to compute speeds of propagation of two most-left discontinuities. However, we can approximate it with the propagation speed of third-most-left discontinuity. For small x-step such approximation should be good. The method is illustrated on the Figure 10.
We implemented simple explicit method as Octave function `HJB_explicit`. Modified Godunov method is implemented as Octave function `HJB_godunov0`. Both of this functions have an parameter "typ" which should be set to 0 for the case of $\varepsilon = 0$ and to 1, if this modification ($\varepsilon > 0$) is needed.

3.4 First approach to boundary conditions

Let us try to develop boundary conditions for Modified Godunov method, so that we wouldn't need to compute extra values in later time layers anymore. Again, let us consider $\varepsilon = 0$. First, let us do one important assumption; Assume, that for our utility function $U(x)$ holds

$$\lim_{x\to\infty} 1 - \frac{U''(x)}{U'(x)} = const \qquad \lim_{x\to-\infty} 1 - \frac{U''(x)}{U'(x)} = const \tag{45}$$

These requirements are not very restrictive, they hold for a large class of suitable utility functions for example linear utility function $U(x) = x$ or exponential utility function $U(x) = -e^{-x}$. These requirements insure that for large positive or large negative x, the solution behaves approximately as traveling wave solution. On this approximation will be based also our boundary conditions. Now, let us assume, that the right boundary is far enough, and the solution behaves everywhere around it almost like traveling wave solution with constnat wave speed. If the wave comes to the boundary cells from left, we have no problem, since new values on the right boundary depend just on values inside our computational domain. If the wave comes from right, than we approximate the solution right from the boundary by traveling wave solution, and we can compute exactly which interval from the terminal layer flows into the boundary cell. The final formula for computing value in the right boundary cell in any time layer t_j will be

$$v_R^{j-1} = v_R^j - \chi_L \frac{Adt}{dx}(v_{R-1}^j - v_R^j) + \chi_R \frac{Adt}{dx}(U(x_R + dx + \tilde{A}(M-j)) - v_R^j) \tag{46}$$

where M is index of the terminal time layer, A is the wave speed between time layers $j-1$ and j, and $\tilde{A}$ is wave speed between time layers j and M. If the wave comes from left we do not need to compute $\tilde{A}$, and A will be approximated by $[\varepsilon e^{-x_R} + r - \alpha(\varphi_{R-3/2}^j)]$, since we have enough data for computing $\varphi_{R-3/2}^j$, but not for computing $\varphi_{R-1/2}^j$ or $\varphi_{R+1/2}^j$. If the wave comes from right, we have several possibilities how to approximate A, $\tilde{A}$. Some of the possibilities are

$$r - \alpha(\varphi_{R-3/2}^{j-1}), \qquad r - \alpha(\varphi_{R-3/2}^{M}), \qquad \frac{1}{M-j}\sum_{i=j+1}^{M}[r - \alpha(\varphi_{R-3/2}^{i})] \tag{47}$$

Since we treat our solution near the boundary as traveling wave solution, A and $\tilde{A}$ should be equal. Also, in case of traveling wave solution the possibilities proposed in (47) how to compute speed should be equivalent. However, since the real solution is not exactly traveling wave, and because of numerical errors, using different approximations from (47) provides us with slightly different results. For our purposes, we choosed first approximation for computing A and third for computing $\tilde{A}$. Figure 11 illustrates this approach.

We should also note that we also do not have enough data for computing v_{R-1}^{j-1}, if the wave comes from right. In that case, we again approximate $\alpha(\varphi_{R-1/2}^j)$ by $\alpha(\varphi_{R-3/2}^j)$.

Now let us examine the behavior on the left boundary. Again, we assume that the left boundary cells are far-left enough, so that the solution can be approximated by traveling wave. All following deductions are done in the same way as on the right

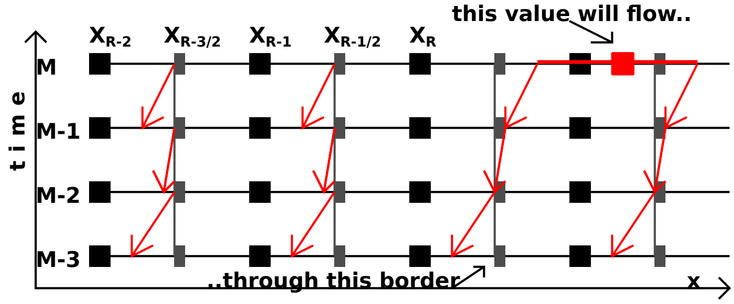

Figure 11: illustration of our first approach to boundary conditions. Red vectors represent wave speeds, and red square represents the value which propagates from the right boundary in time M-3

boundary, therefore we state here just the formula for computing the value v_L^{j-1} of the left boundary cell:

$$v_L^{j-1} = v_L^j - \chi_L \frac{Adt}{dx}(U(x_L - dx + \tilde{A}(M-j)) - v_L^j) + \chi_R \frac{Adt}{dx}(v_{L+1}^j - v_L^j) \tag{48}$$

Again, we don't have enough data to compute $\varphi_{L-1/2}^j$ and $\varphi_{L+1/2}^j$, and we approximate them with $\varphi_{L+3/2}^j$. We implemented Modified Godunov method with boundary conditions proposed here, as Octave function `HJB_godunovx`.

Error spreading through the boundary

The approximations leading to this boundary conditions seem to be correct. However, after implementing the method, we can see the error spreading through the boundary. This is also illustrated in Chapter 5.4. Now we will examine on a simple example, why this happens. Let us think about the case that $\varepsilon = 0$ and utility function has the form $U(x) = -e^{-x}$. Then solution of the HJB equation is

$$V(x,t) = -e^{-(x+(T-t)(r-\alpha(2)))} \tag{49}$$

which is simply traveling wave. It can be verified simply by substituting into the HJB equation. We will examine such cases with analytical solution later in Chapter 5.2. We can see, that this solution implies constant $\varphi(x,t) = 2$, which implies constant portfolio composition and constant wave speed in Modified Godunov method. We will demand, that our numerical method will preserve this property. We can check, that φ in the last time layer is already constant also if computed numerically:

$$\begin{aligned}\varphi_{i+1/2}^M &= 1 - \frac{d_x^2 v_i^M}{d_x v_i^M} = 1 - \frac{v_{i+2}^M - v_{i+1}^M - v_i^M + v_{i-1}^M}{2dx(v_{i+1}^M - v_i^M)} \\ &= 1 - \frac{e^{2dx} - e^{dx} - 1 + e^{-dx}}{2dx(e^{dx}-1)} = 1 - e^{-dx}\frac{e^{2dx}-1}{2dx}\end{aligned} \tag{50}$$

which does not depend on i (just on step size dx) and approaches 2 with $dx \to 0$. Now, let us suppose, that wave is traveling from right, and denote the wave speed by A. Then, value of some non-boundary cell v_i^{M-2} in the $M-2$-time layer will be computed as

$$v_i^{M-2} = (1-A)v_i^{M-1} + Av_{i+1}^{M-1} = (1-A)^2 v_i^M + 2(1-A)Av_{i+1}^M + A^2 v_{i+2}^M \tag{51}$$

Now, if we will try to compute numerically $\varphi_{i-3/2}^{M-2}$ by employing this formula (for $v_{i-3}^{M-2}, v_{i-2}^{M-2}, v_{i-1}^{M-2}v_i^{M-2}$), we will get the same result as in (50). This is caused by the fact that weights $(1-A)^2, 2(1-A)A, A^2$ in expression $(1-A)^2v_i^M+2(1-A)Av_{i+1}^M+A^2v_{i+2}^M$ sum up to 1. So in this case, φ remained constant and demanded property is preserved. However, if v_i^{M-2} is value of a boundary cell, it will be computed by the formula

$$v_i^{M-2} = (1-A)v_i^{M-1} + Av_{i+1+A}^M = (1-A)^2v_i^M + (1-A)Av_{i+1}^M + Av_{i+1+A}^M \qquad (52)$$

where v_{i+1+A}^M is the value from the terminal layer which propagates through the right boundary in time $M-2$, and v_{i+1}^M, v_{i+1+A}^M can be computed exactly as $U(x_R+dx), U(x_R+(1+A)dx)$. However, if we use this value (and values $v_{i-3}^{M-2}, v_{i-2}^{M-2}, v_{i-1}^{M-2}$) to compute $\varphi_{i-3/2}^{M-2}$ numerically, we will not get the same result as in (50). This means, numerically computed φ is not constant near the boundary, therefore also wave speed is different, and the solution is not traveling wave anymore. In chapter 5.4, we can see how this jump in φ affects the whole result.

3.5 Valid approach to boundary conditions

Although our first approach was based on the fact that $\lim_{x\to\infty} 1 - \frac{U'(x)}{U''(x)} = const$, $\lim_{x\to-\infty} 1 - \frac{U'(x)}{U''(x)} = const$ and therefore φ is almost constant near the boundary, we have seen that after 2 iterations it does not hold anymore even in the case when φ should be exactly constant everywhere. Our next approach to boundary conditions, inspired by discussions with Professor Ehrhardt, will be based on constant φ near the boundary even more. Again, we first assume $\varepsilon = 0$. Then, the boundary condition is formulated as:

$$\varphi^j = 1 - d_x^2v^j/\partial_x v^j \text{ and therefore also } d_x^2v^j/d_xv^j \text{ are constant for the boundary cells.}$$

As that constant for the second expression, we will take the value $1 - \varphi^{j+1} = \frac{d_x^2v^{j+1}}{d_xv^{j+1}}$ from the boundary in the previous time layer. As φ near the boundary, which should remain constant, we will take $\varphi_{N-1/2}$, because we have enough values to compute it numerically. Let us remind our approximations of first and second derivative near the boundary

$$d_xv_{N-1/2}^j = \frac{v_N^j - v_{N-1}^j}{dx}, \qquad d_x^2v_{N-1/2}^j = \frac{v_{N+1}^j - v_N^j - v_{N-1}^j + v_{N-2}^j}{2dx^2} \qquad (53)$$

Now, we can have a closer look at the implementation of boundary condition:

$$\frac{d_x^2v_{N-1/2}^j}{d_xv_{N-1/2}^j} = 1 - \varphi_{N-1/2}^{j+1}$$

$$\frac{v_{N+1}^j - v_N^j - v_{N-1}^j + v_{N-2}^j}{2dx(v_N^j - v_{N-1}^j)} = 1 - \varphi_{N-1/2}^{j+1}$$

$$v_{N+1}^j = [2dx(1-\varphi_{N-1/2}^{j+1}) + 1]v_N^j + [-2dx(1-\varphi_{N-1/2}^{j+1}) + 1]v_{N-1}^j - v_{N-2}^j \qquad (54)$$

This condition is needed only if the wave flows into the right boundary cell from right, since in that case, we don't know the value flowing in. If the wave flows into right boundary cell from left, we can compute the value of the cell by using (44). We derive the left boundary condition in the same manner and end up with

$$v_1^j = [2dx(1 - \varphi_{5/2}^{j+1}) + 1]v_2^j + [-2dx(1 - \varphi_{5/2}^{j+1}) + 1]v_3^j - v_4^j \tag{55}$$

Now, let us summarize the whole algorithm. In **each time layer:**

1. We compute wave speeds $[r + \alpha(\varphi_{i+1/2}^j)]$. We approximate wave speeds near the boundary for which we don't have enough data by $[r + \alpha(\varphi_{5/2}^j)]$ (for the left boundary), and by $[r + \alpha(\varphi_{N-1/2}^j)]$ (for the right boundary)

2. We compute values of all non-boundary cells (2,3..N) using formula (44).

3. We look, if the wave flows into the right boundary cell ((N+1)th cell) from right. If yes, we compute its value using (54), otherwise we compute its value using (44).

4. We look if the wave flows into the left boundary cell (first cell) from left. If yes, we compute its value using (55), otherwise we compute its value using (44).

Case of money inflow

Let us examine the case of $\varepsilon > 0$. Now we should use new wave speeds of the form $[\varepsilon e^{-x} + r - \alpha(\varphi)]$ throughout the whole algorithm. The question arises, if our boundary conditions remain legitimate. If the right boundary is deep in positive numbers enough, then εe^{-x} is small near the boundary, so we can neglect it, approximate the behavior behind the boundary by traveling wave, set φ near the boundary as approximately constant, and use the same approach as in the case of $\varepsilon = 0$. The problem on the left boundary will also vanish; If the left boundary is deep in negative numbers enough, εe^{-x} will be large, wave speed $[\varepsilon e^{-x} + r - \alpha(\varphi)]$ will be positive, and the wave will move from the right so that we don't need any left boundary condition. However, as also in Chapter 3.3, we should take care that εe^{-x} is not too large and therefore the whole scheme unstable.

Modified Godunov method with this type of boundary conditions is implemented as Octave function `HJB_godunov1`.

4 Implicit methods, Ricatti transformation

4.1 Implicit method based on 1. order PDE form

The method, which will be introduced now, can be seen as improvement of simple explicit method from Chapter 3.1. At first, let us think about $\varepsilon = 0$. We again begin with 1. order PDE form of HJB equation:

$$\partial_t V + [r - \alpha(\varphi)]\partial_x V = 0$$

Now, we substitute the derivatives with finite differences. Time derivative will be substituted by backward difference, and derivatives in x will be substituted by central differences. However, they will be evaluated in some time between $j-1$ and j in contrast to Simple explicit scheme, in which they were evaluated strictly in time j. The numerical scheme can be now expressed by a set of equations:

$$V_i^{j-1} = V_i^j + dt(r - \alpha(\varphi_i^j))d_x V_i^* \tag{56}$$

$$d_x V_i^* = q\frac{V_{i+1}^{j-1} - V_{i-1}^{j-1}}{2dx} + (1-q)\frac{V_{i+1}^j - V_{i-1}^j}{2dx}, \qquad \varphi_i^j = 1 - \frac{d_x^2 V_i^j}{d_x V_i^j}$$

$$d_x V_i^j = \frac{V_{i+1}^j - V_{i-1}^j}{2dx}, \qquad d_x^2 V_i^j = \frac{V_{i+1}^j - 2V_i^j + V_{i-1}^j}{(dx)^2} \tag{57}$$

where dx is the size of x-step, dt is the size of time-step, and $q \in [0,1]]$ is a parameter specifying the type of equation. As terminal condition we use $V_i^M = U(x_i)$ where $U(x)$ is an utility function. By substituting for $d_x V_i^*$ in (56), and by rearranging it, we get

$$V_i^j - \frac{dt}{2dx}[r - \alpha(\varphi_i^j)](q-1)(V_{i+1}^j - V_{i-1}^j) = V_i^{j-1} - \frac{dt}{2dx}[r - \alpha(\varphi_i^j)]q(V_{i+1}^{j-1} - V_{i-1}^{j-1}) \tag{58}$$

For $q = 0$ we get explicit scheme, for $q = 1$, we get implicit scheme, and for $q = 1/2$ we get Crank-Nicholson type of scheme. N-1 equations of the form (58) for $j = 2, 3, 4...N$ form system of equations which can be represented as

$$AV^j = BV^{j-1} \tag{59}$$

where V^j is a column vector with values V_i^j on i-th position. A is an tridiagonal matrix with $\frac{dt}{2dx}[r - \alpha(\varphi_i^j)](q-1)$ on (i,i-1)th position, 1 on (i,i)th position, and $-\frac{dt}{2dx}[r - \alpha(\varphi_i^j)](q-1)$ on the (i,i+1)th position, for i=1,2,3...N. B is also (almost) tridiagonal matrix with $\frac{dt}{2dx}[r - \alpha(\varphi_i^j)]q$ on (i,i-1)th position, 1 on (i,i)th position, and $-\frac{dt}{2dx}[r - \alpha(\varphi_i^j)]q$ on the (i,i+1)th position, for i=1,2,3...N. V^{j-1} is column vector of unknown values V_i^{j-1}, from the next (previous in time) time-layer. By solving this system of equations, we approach from one time layer to the next (previous in time). We haven't yet set the first and the last row of matrices A, B. At least for B it is necessary, since if any row would be zero-valued, B wouldn't be invertible. To do it, we will employ boundary conditions derived in part 3.5. This means, near the right boundary we demand φ to be constant with some value φ_R, and therefore also $\partial_x^2 V/\partial_x V$ should be constant with value $1 - \varphi_R$. after substituting $\partial_x^2 V$, $\partial_x V$ with approximations 57, we get

$$\frac{2(V_{N+1}^{j-1} - 2V_N^{j-1} + V_{N-1}^{j-1})}{dx(V_{N+1}^{j-1} - V_{N-1}^{j-1})} = 1 - \varphi_R$$

$$[2 + (1-\varphi_R)dx]V_{N-1}^{j-1} - 4V_N^{j-1} + [2 - (1-\varphi_R)dx]V_{N+1}^{j-1} = 0 \tag{60}$$

We use the same approach for left boundary condition:

$$[2+(1-\varphi_L)dx]V_3^{j-1} - 4V_2^{j-1} + [2-(1-\varphi_L)dx]V_1^{j-1} = 0 \tag{61}$$

where φ_L is the value of φ near the left boundary (supposed to be constant). Coefficients on the left side of (60) will form the first row of matrix B, and coefficients on the left side of (61) will form its last row. The first and last row of matrix A will be zero-valued (in accordance with zero-valued right sides of (60), (61)). Our last problem is, how to determine values φ_R (or φ_L). Natural choice would be to take φ_N (or φ_2) from the previous layer. However, for the case of $\varepsilon > 0$, we will see that choice of another values is more reasonable.

Case of money inflow

Now let us examine the case of $\varepsilon > 0$. Again we should in the whole scheme substitute $[r - \alpha(\varphi)]$ by $[r + \varepsilon e^{-x} - \alpha(\varphi)]$. The question is the same as in Chapter 3.5: Are our boundary conditions valid also now? As already explained in Chapter 3.5, near the right boundary εe^{-x} is small (can be approximated by zero), so there is no problem with boundary condition. However, near the left boundary εe^{-x} grows dramatically with $x \to -\infty$, so the wave speed $[r+\varepsilon e^{-x}-\alpha(\varphi)]$ cannot be approximated by constant, and we cannot speak about traveling wave anymore. In our case, we can assume the wave is flowing from right (since εe^{-x} can be assumed to be large enough). If we think again in terms of finite volumes, we can state, that the "outflow" from the left boundary cell is bigger then the "inflow", since $\varepsilon e^{-x_{j-1}} > \varepsilon e^{x_j}$. As consequence the solution does not preserve its wave-profile, and we cannot regard φ near the left boundary as constant. The solution to this issue which we propose here is not well reasoned, and should be considered as rather heuristic. We will test it's functionality by numerical experiments.

- If we will take $\varphi_L = \varphi_2^j$ in (61), then, according to the boundary condition, also φ_2^{j-1} will be equal to φ_2^j, and φ_2^j will remain constant on the whole left boundary. This is problem, since φ may not remain constant for the real solution with $\varepsilon > 0$.

- Therefore, we will use $\varphi_L = \varphi_3^j$ by computing (j-1)-time layer. New φ_2^{j-1}, will be equal to φ_3^j, however new φ_3^{j-1} will have some other value depending on the whole system of equations. This new value will be taken as φ_L for the computation of next time layer.

- In favor of this approach speaks also the fact, that φ_3^j is value right from the boundary, and the wave travels also from the right.

- For symetry, we will use the same approach also on the right boundary, taking $\varphi_R = \varphi_{N-1}$instead of $\varphi_R = \varphi_N$. This will not cause any harm, because φ is supposed to be constant in some neighborhood of the right boundary. For the same reason, this approach is also approved in case of $\varepsilon = 0$.

This first order PDE form implicit method is implemented as Octave function `HJB_1order`, with q (for implicit, explicit or other version of the scheme) as parameter.

4.2 Implicit method based on parabolic PDE form

This scheme is inspired by numerical schemes for solving Heat equation and parabolic PDE's in general, presented in [21]. In contrast to all methods presented up to now, this method will be based on parabolic PDE form of HJB Equation:

$$\partial_t V + [\varepsilon e^{-x} + r + \beta(\varphi)]\partial_x V + \gamma(\varphi)\partial_x^2 V = 0 \tag{62}$$

However, the whole process of derivation of the scheme runs the same way as in the case of implicit scheme based on 1. order PDE form (Chapter 4.1), therefore, we will describe only the main points. After substituting derivatives with finite differences we end up with the scheme

$$\begin{aligned} V_i^{j-1} &= V_i^j + dt[\varepsilon e^{-x_i} + r + \beta(\varphi_i^j)]d_x V_i^* + dt\gamma(\varphi_i^j)d_x^2 V_i^* \\ d_x V_i^* &= q\frac{V_{i+1}^{j-1} - V_{i-1}^{j-1}}{2dx} + (1-q)\frac{V_{i+1}^j - V_{i-1}^j}{2dx} \\ d_x^2 V_i^* &= q\frac{V_{i+1}^{j-1} - 2V_i^{j-1} + V_{i-1}^{j-1}}{(dx)^2} + (1-q)\frac{V_{i+1}^j - 2V_i^j + V_{i-1}^j}{(dx)^2} \end{aligned} \tag{63}$$

$$\varphi_i^j = 1 - \frac{d_x^2 V_i^j}{d_x V_i^j}, \qquad d_x V_i^j = \frac{V_{i+1}^j - V_{i-1}^j}{2dx}, \qquad d_x^2 V_i^j = \frac{V_{i+1}^j - 2V_i^j + V_{i-1}^j}{(dx)^2} \tag{64}$$

Let us define abbreviations

$$\begin{aligned} b_{-1}^i &= \frac{dt}{2dx}(\varepsilon e^{-x_i} + r + \beta(\varphi_i^j)) - \frac{dt}{dx^2}\gamma(\varphi_i^j) \\ b_0^i &= \frac{2dt}{dx^2}\gamma(\varphi_i^j) \\ b_1^i &= \frac{dt}{2dx}(\varepsilon e^{-x_i} + r + \beta(\varphi_i^j)) + \frac{dt}{dx^2}\gamma(\varphi_i^j) \end{aligned} \tag{65}$$

By subtituting for $d_x V_i^*$, $d_x^2 V_i^*$ in (63), rearanging it and employing substitutions (65), we get

$$qb_{-1}^i V_{i-1}^{j-1} + [1+qb_0^i]V_i^{j-1} - qb_1^i V_{i+1}^{j-1} = (q-1)b_{-1}^i V_{i-1}^j + [1+(q-1)b_0^i]V_i^j - (q-1)b_1^i V_{i+1}^j \tag{66}$$

For $q = 0$ we get explicit scheme, for $q = 1$, we get implicit scheme, and for $q = 1/2$ we get Crank-Nicholson type of scheme. Equations of the form (66) for i=2,3,4...N again form system of equations

$$AV^j = BV^{j-1} \tag{67}$$

where V^j is a column vector with values V_i^j on i-th position. A is an tridiagonal matrix with $(q-1)b_{-1}^i$ on (i,i-1)th position, $[1+(q-1)b_0^i]$ on (i,i)th position, and $-(q-1)b_1^i$ on the (i,i+1)th position, for i=2,3...N. B is also (almost) tridiagonal matrix with qb_{-1}^i on (i,i-1)th position, $[1+qb_0^i]$ on (i,i)th position, and $-qb_1^i$ on the (i,i+1)th position, for i=2,3...N. V^{j-1} is column vector of unknown values V_i^{j-1}, from the next (previous in time) time-layer. By solving this system of equations, we again get the values of V_i^{j-1} from the next time layer.

We again need to set the first and last row of matrices A, B. We will do that exactly the same way as in Chapter 4.1: B will have $[2-(1-\varphi_3^j)dx]$ on the (1,1)-position, -4 on

(1,2)-position, $[2+(1-\varphi_3^j)dx]$ on (1,3)-position, $[2-(1-\varphi_{N-1}^j)dx]$ on the (N+1,N+1)-position, -4 on (N+1,N)-position, $[2+(1-\varphi_{N-1}^j)dx]$ on (N+1,N-1)-position. First and last row of matrix A will be zero-valued. This Parabolic PDE form implicit method is implemented as Octave function `HJB_parabolic0`, with q (for implicit, explicit or other version of the scheme) as parameter.

4.3 Implicit method with exponential correction

After implementation of proposed methods and testing them on a few examples (see Chapter 5.4), we will see that our methods perform quite bad for utility functions with rapid slope growth (that means large φ), as for example $U(x) = e^{-10x}$. Our next approach will try to fix it. It's main idea is filtering out the large-φ-part by using a simple substitution and deriving PDE for a new variable. Let us define

$$W(t,x) = e^{Kx}V(t,x), \qquad W^T(x) = e^{Kx}U(x) \tag{68}$$

where K is some real parameter. Then, $V(t,x) = e^{-Kx}W(t,x)$, and our HJB equation

$$\partial_t V + [\varepsilon e^{-x} + r + \beta(\varphi)]\partial_x V + \gamma(\varphi)\partial_x^2 V = 0 \tag{69}$$

together with terminal condition $V(T,x) = U(x)$ and boundary conditions derived in Chapter 3.5 can be written as:

$$\begin{aligned}&\partial_t W e^{-Kx} + [\varepsilon e^{-x} + r + \beta(\varphi)](\partial_x W - KW)e^{-Kx}\\ &+\gamma(\varphi)(\partial_x^2 W - 2K\partial_x W + K^2 W)e^{-Kx} = 0\end{aligned} \tag{70}$$

with terminal condition $W(T,x) = W^T(x)$. The main advantage of this PDE problem in contrast to the original problem (69), is that φ for terminal condition $W^T(x)$ will be far smaller then φ for terminal condition $U(x)$ of the original problem, if proper K is set. For example, we can "flatten" utility function e^{-10x} to function e^{-x}, by choosing $K = 9$.
We divide (70) by e^{-Kx}, and substitute derivatives with finite differences. We again end up with a scheme

$$\begin{aligned}W_i^{j-1} &= W_i^j + dt[\varepsilon e^{-x_i} + r + \beta(\varphi_i^j)](d_x W_i^* - KW_i^*)\\ &+ dt\gamma(\varphi_i^j)(d_x^2 W_i^* - 2K d_x W_i^* + K^2 W_i^*)\end{aligned} \tag{71}$$

$$\begin{aligned}W_i^* &= qW_i^{j-1} + (1-q)W_i^j\\ d_x W_i^* &= q\frac{W_{i+1}^{j-1} - W_{i-1}^{j-1}}{2dx} + (1-q)\frac{W_{i+1}^j - W_{i-1}^j}{2dx}\\ d_x^2 W_i^* &= q\frac{W_{i+1}^{j-1} - 2W_i^{j-1} + W_{i-1}^{j-1}}{dx^2} + (1-q)\frac{W_{i+1}^j - 2W_i^j + W_{i-1}^j}{dx^2}\\ \varphi_i^j &= 1 - \frac{d_x^2 V_i^j}{d_x V_i^j} = 1 - \frac{d_x^2 W_i^j - 2K d_x W_i^j + K^2 W_i^j}{d_x W_i^j - K W_i^j}\end{aligned} \tag{72}$$

$$d_x W_i^j = \frac{W_{i+1}^j - W_{i-1}^j}{2dx}, \qquad d_x^2 W_i^j = \frac{W_{i+1}^j - 2W_i^j + W_{i-1}^j}{dx^2} \tag{73}$$

Now, we define abbreviations

$$\begin{aligned} b^i_{-1} &= \frac{dt}{2dx}[\varepsilon e^{-x_i} + r + \beta(\varphi^j_i)] - dt\gamma(\varphi^j_i)(\frac{1}{dx^2} + \frac{K}{dx}) \\ b^i_0 &= dt[\varepsilon e^{-x_i} + r + \beta(\varphi^j_i)]K - dt\gamma(\varphi^j_i)(-\frac{2}{dx^2} + K^2) \\ b^i_1 &= -\frac{dt}{2dx}[\varepsilon e^{-x_i} + r + \beta(\varphi^j_i)] - dt\gamma(\varphi^j_i)(\frac{1}{dx^2} - \frac{K}{dx}) \end{aligned} \tag{74}$$

By substituting for $d_x W^*_i$, $d^2_x W^*_i$ in (71), rearanging it, and employing substitutions (74), we get

$$\begin{aligned} &qb^i_{-1}W^{j-1}_{i-1} + [1 + qb^i_0]W^{j-1}_i + qb^i_1 W^{j-1}_{i+1} \\ &= (q-1)b^i_{-1}W^j_{i-1} + [1 + (q-1)b^i_0]W^j_i - (q-1)b^i_1 W^j_{i+1} \end{aligned} \tag{75}$$

For $q = 0$ we get explicit scheme, for $q = 1$, we get implicit scheme, and for $q = 1/2$ we get Crank-Nicholson type of scheme. Equations of the form (75) for i=2,3,4...N again form system of equations

$$AW^j = BW^{j-1} \tag{76}$$

where W^j is a column vector with values W^j_i on i-th position. A is an tridiagonal matrix, with $(q-1)b^i_{-1}$ on (i,i-1)th position, $[1 + (q-1)b^i_0]$ on (i,i)th position, and $(q-1)b^i_1$ on the (i,i+1)th position, for i=2,3...N. B is also (almost) tridiagonal matrix, with qb^i_{-1} on (i,i-1)th position, $[1 + qb^i_0]$ on (i,i)th position, and qb^i_1 on the (i,i+1)th position, for i=2,3...N. W^{j-1} is column vector of unknown values W^{j-1}_i, from the next (previous in time) time-layer. By solving this system of equations, we again get the values of W^{j-1}_i from the next time layer.

Boundary conditions

As well as in previous methods, we need to set the first and last row of matrix B to be invertible. We again employ boundary conditions. Standard boundary conditions were of the form $d^2_x V^{j-1}_N / d_x V^{j-1}_N = 1 - \varphi_R$ near the right boundary and $d^2_x V^{j-1}_2 / d_x V^{j-1}_2 = 1 - \varphi_L$ near the left boundary. After transformation to variable W, the right boundary condition transforms to

$$\frac{d^2_x W^{j-1}_N - 2K d_x W^{j-1}_N + K^2 W^{j-1}_N}{d_x W^{j-1}_N - K W^{j-1}_N} = 1 - \varphi_R \tag{77}$$

After substituting for $d_x W^{j-1}_N$, $d^2_x W^{j-1}_N$ and rearanging the equation we get

$$\begin{aligned} &[\frac{1}{dx^2} + \frac{K}{dx} + \frac{1-\varphi_R}{2dx}]W^{j-1}_{N-1} + [-\frac{2}{dx^2} + K^2 + (1-\varphi_R)K]W^{j-1}_N \\ &+[\frac{1}{dx^2} - \frac{K}{dx} - \frac{1-\varphi_R}{2dx}]W^{j-1}_{N+1} = 0 \end{aligned} \tag{78}$$

Now the values in the last row B will be formed by coefficients on the left-hand side of the equation (78). By the same approach we derive boundary condition in this form for the left boundary, and values for the first row of matrix B follow.
However, we have also another possibility how to pose boundary conditions for our

problem. To state our boundary conditions we already demanded $\lim_{x\to\infty} U''(x)/U'(x) = const$, $\lim_{x\to-\infty} U''(x)/U'(x) = const$. Now, we will moreover demand

$$\begin{aligned} \lim_{x\to\infty} \frac{U'(x)}{U(x)} &= C_R & \qquad \lim_{x\to-\infty} \frac{U'(x)}{U(x)} &= C_L \\ \lim_{x\to\infty} U'(x) - C_R U(x) &= 0 & \qquad \lim_{x\to-\infty} U'(x) - C_L U(x) &= 0 \end{aligned} \tag{79}$$

for some constants C_L, C_R. Also this property holds for large class of utility functions. Moreover, by differentiating (79) also $\lim_{x\to\infty} U''(x)/U'(x) = C_R$, $\lim_{x\to-\infty} U''(x)/U'(x) = C_L$ follows. Therefore, as it is explained in Chapters 3.4, 3.5, the solution near and behind the boundaries can be approximated by traveling wave (at least in case of $\varepsilon = 0$) and $\partial_x^2 V/\partial_x V$ and now also $\partial_x V/V$ remain constant near the boundaries. Moreover, it holds $\partial_x^2 V = C_R \partial_x V = C_R^2 V$ near the right boundary and $\partial_x^2 V = C_L \partial_x V = C_L^2 V$ near the left boundary. Now, let us examine $\partial_x^2 W/\partial_x W$ near the right boundary:

$$\begin{aligned} \frac{\partial_x^2 W}{\partial_x W} &= \frac{\partial_x^2 (V e^{Kx})}{\partial_x (W e^{Kx})} = \frac{\partial_x^2 V + 2K\partial_x V + K^2 V}{\partial_x V - KV} \\ &= \frac{C_R^2 V + 2KC_R^2 V + K^2 V}{C_R V - KV} = \frac{C_R^2 + 2KC_R^2 + K^2}{C_R - K} = G_R \end{aligned} \tag{80}$$

where G_R is constant. So we can pose the boundary condition on the right boundary in the same manner as in Chapters 4.1, 4.2 : $\partial_x^2 W - G_R \partial_x W = 0$. After substituting derivatives with finite differences and multiplying the expression with dx, we get

$$[2 + G_R dx] W_{N-1}^{j-1} - 4W_N^{j-1} + [2 - G_R dx] V_{N+1}^{j-1} = 0 \tag{81}$$

what is formally the same as in (60). Coefficients on the left hand side of (81) will form last row of matrix B. We analogously derive the left boundary condition:

$$[2 + G_L dx] W_1^{j-1} - 4W_2^{j-1} + [2 - G_L dx] V_3^{j-1} = 0 \tag{82}$$

Coefficients on the left hand side of (81) will form first row of matrix B. First and last row of matrix A will be zero valued. As G_L, G_R, we take values of $d_x^2 W_3^{j-1}$, $d_x^2 W_{N-1}^{j-1}$ form the previous time layer, what is again based on the argumentation used by setting boundary conditions in Chapter 4.1.

Now we introduced two approaches to boundary conditions in case of implicit method with exponential correction. We will use the second approach (based on equations (81), (82)), because we got better numerical results by testing it. Our Implicit method with exponential correction is implemented as Octave function `HJB_parabolic1`, with q (for implicit, explicit or other version of the scheme) and K as parameters.

4.4 Methods based on Ricatti transformation

One of the goals of this work is to compare numerical method based directly on Hamilton-Jacobi-Bellman equation, with methods for solving related equation from Theorem 2.1, which are proposed in [9]. We describe here shortly derivation of this method, paper [9] should be read for more details. We begin with the equation for $\varphi(x,t) = 1 - \partial_x^2 V(x,t)/\partial_x V(x,t)$ from Theorem 2.1:

$$\partial_t\varphi + \partial_x^2\alpha(\varphi) + \partial_x[(\varepsilon e^{-x} + r)\varphi + (1-\varphi)\alpha(\varphi)] = 0 \tag{83}$$

with terminal condition $\varphi(x,T) = U''(x)/U'(x)$. We integrate our equation over finite volumes $[x_{i-1/2}, x_{i+1/2}]$, therefore, this method can be seen as finite volume method.

$$\int_{x_{i-1/2}}^{x_{i+1/2}} \partial_t\varphi dx + \int_{x_{i-1/2}}^{x_{i+1/2}} \partial_x^2\alpha(\varphi) + \partial_x[(\varepsilon e^{-x} + r)\varphi + (1-\varphi)\alpha(\varphi)]dx = 0 \tag{84}$$

Our scheme will consist of $M+1$ time layers, each with $N-1$ values φ_i^j approximating $\varphi(x_i, t_j)$, where j is index of time layer. We approximate $\partial_t\varphi$ on $[x_{i-1/2}, x_{i+1/2}]$ with value constant $\frac{\varphi_i^j - \varphi_i^{j-1}}{dt}$, where j, $j-1$ is index of the time-layer. Thus first integral in (84) will be approximated by $\frac{\varphi_i^j - \varphi_i^{j-1}}{dt} dx$. Now we evaluate the second integral expression, and get

$$\left[\partial_\varphi\alpha(\varphi)\partial_x\varphi + (\varepsilon e^{-x} + r)\varphi - (1-\varphi)\alpha(\varphi)\right]_{x_{i-1/2}}^{x_{i+1/2}} \tag{85}$$

where we employed $\partial_x\alpha(\varphi) = \partial_\varphi\alpha(\varphi)\partial_x\varphi$. We approximate $\varphi(x_{i+1/2}, t_*)$ by $\varphi_{i+1/2}^* = (\varphi_{i+1}^* + \varphi_i^*)/2$ and $\varphi(x_{i-1/2}, t_*)$ by $\varphi_{i-1/2}^* = (\varphi_i^* + \varphi_{i-1}^*)/2$, where $*$ is time index which will be specified later. We approximate $\partial_x\alpha(\varphi(x_{i+1/2}, t_*))$ by $(\alpha(\varphi_{i+1}^*) - \alpha(\varphi_i^*))/(\varphi_{i+1}^* - \varphi_i^*)$. For approximation of $\partial_x\alpha(\varphi(x_{i-1/2}, t_*))$ we use the same approach. The expression $\partial_x\varphi(x_{i+1/2}, t_*)$ will be approximated by $(\varphi_{i+1}^* - \varphi_i^*)/dx$ and $\partial_x\varphi(x_{i-1/2}, t_*)$ will be approximated by $(\varphi_i^* - \varphi_{i-1}^*)/dx$. Now we define an abbreviation

$$\begin{aligned} D_{i\pm}^* &= (\varepsilon e^{x_{i\pm1/2}} + r)\varphi_{i\pm1/2}^* + (1 - \varphi_{i\pm1/2}^*)\alpha(\varphi_{i\pm1/2}^*) \\ E_{i\pm}^* &= (\pm\alpha(\varphi_{i\pm1}^*) \mp \alpha(\varphi_i^*))/(\pm\varphi_{i\pm1}^* \mp \varphi_i^*) \end{aligned} \tag{86}$$

Substituting our approximations into (85) and using (86), we get can express the second integral as

$$E_{i+}^* \frac{\varphi_{i+1}^* - \varphi_i^*}{dx} + D_{i+}^* - E_{i-}^* \frac{\varphi_i^* - \varphi_{i-1}^*}{dx} - D_{i-}^* \tag{87}$$

Now we substitute both approximations of integrals into (84) rearrange the equation and end up with

$$\varphi_i^{j-1} = \varphi_i^j + \frac{dt}{dx}\left[E_{i+}^* \frac{\varphi_{i+1}^* - \varphi_i^*}{dx} + D_{i+}^* - E_{i-}^* \frac{\varphi_i^* - \varphi_{i-1}^*}{dx} - D_{i-}^*\right] \tag{88}$$

Symbol $*$ will be substituted by time index, depending on the scheme.

Semi-implicit scheme

We choose time index $* = j$ for expressions E_{i+}^*, E_{i-}^*, D_{i+}^*, D_{i-}^* and $* = j-1$ in expressions with φ^*. We end up with scheme

$$\begin{aligned} &-\frac{dt}{dx^2} E_{i-}^j \varphi_{i-1}^{j-1} + \left[1 + \frac{dt}{dx^2}(E_{i+}^j + E_{i-}^j)\right]\varphi_i^{j-1} - \frac{dt}{dx^2} E_{i+}^j \varphi_{i+1}^{j-1} \\ &= \varphi_i^j + \frac{dt}{dx}\left[D_{i+}^j - D_{i-}^j\right] \end{aligned} \tag{89}$$

$N-2$ such equations (for $i = 2, 3, 4 \ldots N-1$) form tridiagonal system of the form $A\varphi^{j-1} = \varphi^j + b$. Coeficients of matrix A are computed from the values from previous time layer.

Fully implicit scheme

Now we choose time index $* = j-1$ for all expressions in (88). Therefore, we have nonlinear system of equations which should be computed iteratively. Let as denote f_i^k the k-th iterative approximation of φ_i^{j-1}. In k-th iteration, we approximate values of $E_{i\pm}^{j-1}$, $D_{i\pm}^{j-1}$ by substituting f_i^k for φ_i^{j-1} and denote this approximations as $\tilde{E}_{i\pm}^k$, $\tilde{D}_{i\pm}^k$. We compute the next approximation f_i^{k+1} of φ_i^{j-1} by solving tridiagonal system

$$\begin{aligned}
&-\frac{dt}{dx^2}\tilde{E}_{i-}^k f_{i-1}^{k+1} + \left[1 + \frac{dt}{dx^2}(\tilde{E}_{i+}^k + \tilde{E}_{i-}^k)\right] f_i^{k+1} - \frac{dt}{dx^2}\tilde{E}_{i+}^k f_{i+1}^{k+1} \\
&= \varphi_i^j + \frac{dt}{dx}\left[\tilde{D}_{i+}^k - \tilde{D}_{i-}^k\right]
\end{aligned} \tag{90}$$

We end after K such steps and take actual f^{k+1} as final value of new time layer φ^{j-1}.

Boundary conditions

As well as in all previous numerical schemes, we need to set the first and last row of matrix A in our tridiagonal system. To achieve this, we again employ boundary conditions, proposed in [9]. Because now we model equation in variable φ instead of HJB equation in variable V, we have more options how to pose boundary conditions.

- If $\varepsilon = 0$, we use the same argumentation as in Chapter 3.5, that φ should be constant near the boundaries. Therefore, as left boundary condition we get $\varphi_2^{j-1} = \varphi_1^{j-1}$ and as right boundary condition we get $\varphi_{N-1}^{j-1} = \varphi_{N-2}^{j-1}$. We have also other possibilities how to state boundary condition based on the assumption of constant φ near the boundary, however, paper [9] uses this form.

- If $\varepsilon > 0$, the part εe^{-x} vanishes with $x \to \infty$, so we can approximate it with zero near the right boundary (assuming it is right enough), and use the same right boundary condition as in case of zero-valued ε: $\varphi_{N-1}^{j-1} = \varphi_{N-2}^{j-1}$.

- The left boundary condition for the case of $\varepsilon > 0$ was not resolved elegantly in previous methods. For $x \to -\infty$ leading term in equation (83) is $\partial_x(\varepsilon e^{-x}\varphi) = \varepsilon e^{-x}(\partial_x\varphi - \varphi)$. The whole left hand side of (83) must be equal to zero. Therefore, to balance the εe^{-x} part of leading term which is approaching to ∞ with $x \to -\infty$ we require that $(\partial_x\varphi - \varphi) \to 0$ with $x \to -\infty$. This means, we can approximate $(\partial_x\varphi - \varphi)$ with zero near the left boundary. Substituting derivative with finite difference, and rearranging the equation for approximation we get left boundary condition for the case of $\varepsilon > 0$: $\varphi_2^{j-1} = \varphi_1^{j-1}(1 + dx)$.

Let us note, that we can't pose left boundary condition of this form in numerical methods for solving HJB Equation in its standard form (with value function $V(t, x)$ as unknown). Semi-implicit method based on Riccati transformation is implemented as Octave function `HJB_ricatti0`. Fully implicit method is implemented as Octave function `HJB_ricatti1`, with K for number of iterates in each computation of new time layer as parameter. Both methods have also parameter "typ" which should be set to 0 for $\varepsilon = 0$, and to 1 for $\varepsilon > 0$, to choose the form of the left boundary condition.

5 Numerical analysis of the solutions of HJB Equation

5.1 Paramater estimation

The core of this Chapter will be the testing of proposed and implemented numerical methods on real data. We will solve numerically Hamilton-Jacobi-Bellman Equation arising from portfolio composition problem, for portfolio composed from assets contained in Swiss Market Index (SMI). SMI is most important stock index of Switzerland, containing 20 largest and most liquid stocks. For comparison, in paper [9], German Stock index DAX is used. To be able to write related HJB Equation, we need to estimate expected return for each of twenty assets, and the variance-covariance matrix of its returns. According to (3), price of i-th asset on day t, Y_t^i is supposed to behave according to stochastic differential equation

$$dY_t^i = (r + \mu_i)Y_t^i dt + Y_t^i \sigma_i dW_t^i \tag{91}$$

If we define $X_t^i = \ln(Y_t^i)$, then using Ito Lemma (Theorem 1.2) to express $dX_t = d\ln(Y_t)$, we get

$$dX_t^i = (r + \mu_i - \frac{1}{2}\sigma_i^2)dt + \sigma_i dW_t^i \tag{92}$$

Solution of this SDE is simply

$$X_t^i = (r + \mu_i - \frac{1}{2}\sigma_i^2)t + \sigma_i W_t^i + c \tag{93}$$

for some constant c. Now, return at the day t, $r_t^i = \ln(Y_t^i/Y_{t-1}^i)$ can be computed as $X_t^i - X_{t-1}^i = (r + \mu_i - \frac{1}{2}\sigma_i^2) + \sigma_i W_1^i$ and is normally distributed with expected value $(r + \mu_i - \frac{1}{2}\sigma_i^2)$ and variance σ_i. Let us note, that our model assumes that returns of the same stock in different days are independent and equally distributed. We estimated the expected value and the covariance matrix of daily returns from the data:

1. We downloaded daily close prices between 1.1.2012 and 1.1.2013 for all 20 assets from [3].
2. We deleted data from days when the assets were not traded. We end up with time serie of T prices Y_t^i, $(t = 1, 2, ...T)$ for each asset i $(i = 1, 2, ...20)$.
3. We estimated daily returns for each day and each asset by formula $\xi_t^i = \ln(Y_{t+1}^i/Y_t^i)$. We denote column vector of returns of assets 1,2,...20 at day t as ξ_t.
4. We computed vector of estimations of expected returns of our assets as $\bar{\xi} = \frac{1}{T-1}\sum_{t=1}^{T-1}\xi_t$
5. We computed estimation of covariance matrix of returns as $\tilde{\Sigma} = \frac{1}{T-1}\sum_{t=1}^{T-1}(\xi_t - \bar{\xi})(\xi_t - \bar{\xi})^T$. Estimations of variances σ_i^2 are on the diagonal. We denote vector of this estimations as $\bar{\sigma}^2$.
6. We estimate vector of values $\tilde{\mu}_i$ as $\tilde{\mu} = \bar{\xi} - r + \frac{1}{2}\bar{\sigma}^2$, where r is daily interest rate.

7. We got parameters $\tilde{\mu}$, $\tilde{\Sigma}$ for HJB Equation with respect to time unit one day. However, we want to work with time unite one year, therefore we will use μ, Σ, computed as $\mu = 252 \cdot \tilde{\mu}$, $\Sigma = 252 \cdot \tilde{\Sigma}$ (a year has 252 trading days).

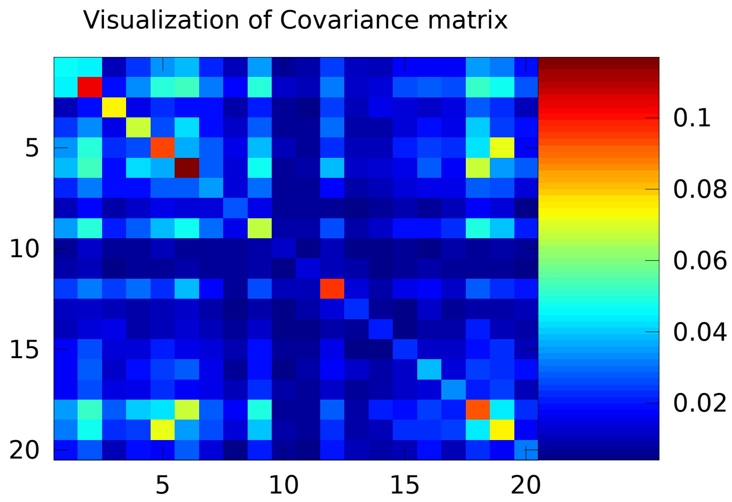

Figure 12: Visualization of estimated covariance matrix. The color indicates how large is the covariance between two assets

Table 1: Estimates of mu, sigma for assets contained in Swiss Market Index (SMI)

Number	Abbreviation	Name	μ	σ^2
1	ABBN	ABB	1.72405	0.046529
2	ADEN	Adecco	2.70618	0.103478
3	ATLN	Actellion	2.4415	0.075208
4	BAER	Julius Baer	1.92889	0.06818
5	CFR	Richemont	2.81299	0.09493
6	CSGN	Credit Suisse	2.68864	0.117013
7	GEBN	Geberit	1.57091	0.035217
8	GIVN	Givaudan	1.36168	0.027055
9	HOLN	Holcim	2.30295	0.066889
10	NESN	Nestle	0.95219	0.011614
11	NOVN	Novartis	1.01524	0.013757
12	RIGN	Transocean	2.56095	0.096639
13	ROG	Roche	1.32976	0.022718
14	SCMN	Swisscom	1.23516	0.020718
15	SCGN	SGS	1.44593	0.022691
16	SREN	Swiss Re	1.89619	0.038837
17	SYNN	Syngenta	1.70829	0.033078
18	UBSN	UBS	2.64162	0.093419
19	UHR	Swatch Group	2.42305	0.075652
20	ZURN	Zurich	1.52781	0.031421

Let us note, that μ, Σ are maximum likelihood estimations of the true parameters. For more details on the maximum likelihood theory, we reffer to [15]. Table 1 shows estimated μ and σ^2 for all twenty assets. Figure 12 illustrates estimated covariance matrix Σ. We see that all assets are slightly positively correlated. As we already mentioned in Chapter 2.3, we will work only with $\varphi \leq 10$. Therefore, as we discovered, in the standard model (non-negative weights summing up to 1) only 6 out of all 20 assets are relevant and will be present in the portfolio with non-zero weight. Those six assets are ADEN,ATLN,CFR,CSGN,RIGN,USBN (underlined in Table 1). All of them are assets which seem to be well-performing (high μ). In following chapters, we will mostly work just with those six assets.

5.2 Test solutions

Our numerical methods were developed in order to be able to solve HJB Equation in cases, when analytical solution is not feasible. However, we need to somehow test the accuracy of those methods. To do this, we will employ two test examples, for which exact analytical solution is known. The first example leads to a solution, that has form of traveling wave in variable $V(x,t)$, and is constant in variable $\varphi(x,t)$. The second example, presented in [9], leads to a solution that has form of traveling wave in variable $\varphi(x,t)$.

Constant Fi Case

In Chapter 3.4, we already used HJB Equation with $\varepsilon = 0$ and utility function $U(x) = -e^{-x}$ and its analytical solution to illustrate the error spreading through the boundary. It is easy to chcek, that we get similar analytical solution for any $U(x) = -e^{-ax+b}$, $\varepsilon = 0$. For such utility function, we have constant $\varphi = 1 - U''(x)/U'(x) = 1 + a$ in the terminal time layer, what indicates wave traveling with constant speed $r + \alpha(1 + a)$. Therefore, the solution will be

$$V(x,t) = -e^{-(x+(T-t)(r-\alpha(a+1)))} \tag{94}$$

what can be veryfied easily by substituting it into HJB Equation ($\partial_t V + [r - \alpha(\varphi)]\partial_x V = 0$). The solution in φ is just a constant value $1 + a$, therefore, in this case, investor does not change the portfolio at all.

Traveling wave Fi Case

Here we will shortly describe, how to construct utility function for which, in case of $\varepsilon = 0$, $r = 0$, a solution with traveling wave φ occurs. The derivation of this traveling wave solution in φ is described in [9], and follows the analysis in [8]. Here we state only the basic equations needed for construction. First we pose an ODE

$$z'(\xi) = K_0 + c\alpha^{-1}(z(\xi)) - z(\xi)(1 - \alpha^{-1}(z(\xi))) \tag{95}$$

and define constants

$$c = \frac{\alpha(v^+)(1-v^+) - \alpha(v^-)(1-v^-)}{v^+ - v^-}, \qquad K_0 = -cv^+ + \alpha(v^+)(1-v^+) \tag{96}$$

depending on some values v^+, v^-. Now we state a Theorem form [9], which shows, how to construct the traveling wave solution in φ:

Theorem 5.1 (Traveling wave solution). *For any two values $v^\pm$, such that $0 < v^- < v^+$ and function $\alpha()$ is C^∞-smooth in $v^\pm$, there exist an unique traveling wave solution $\varphi(x,t) = v(x+c(T-t))$ of transformed HJB Equation (25) such that $\lim_{x\to-\infty} \varphi(x,t) = v^+$ and $\lim_{x\to\infty} \varphi(x,t) = v^-$. The traveling wave profile $v(\xi)$ is a decreasing function given by $v(\xi) = \alpha^{-1}(z(\xi))$ where $z(\xi)$ is a solution to the ODE (95). Traveling wave speed c is given by (96).*

Following this theorem we construct traveling wave solution in φ, and find utility function related to this solution:

1. We set $v^+ = 1.5$, $v^- = 1.3$ (these values are also used in [9]). Following (96), we compute c, K_0.
2. We solve ODE (95) using Octave routine `ode45`. We get traveling wave profile as $v(\xi) = \alpha^{-1}(z(\xi))$, where $z(\xi)$ is the solution of (95).
3. We compute the traveling wave solution in φ, as $\varphi(x,t) = v(x + c(T-t))$. The solution will be computed in grid-points (x_i, t_j) and since we know the solution of ODE (95) also only in some grid points, we employ here Hermite spline interpolation (`alfa_method2`).
4. Following equation (27), $V = \int \exp(\int(1-\varphi))$, we use numerical integration and exponential transformation to get solution in V from solution in φ.
5. We take terminal time layer of this solution in V as utility function $U(x)$

Finally, let us list utility functions which will be used by testing of our numerical methods:

- $U(x) = -e^{-x}$ implemented as Octave function `exp_utility`
- $U(x) = -e^{-5x}$ implemented as Octave function `exp5_utility`
- $U(x) = -e^{-10x}$ implemented as Octave function `exp10_utility`
- $U(x)$ leading to traveling wave solution in φ constructed according tho the outline above, implemented as Octave function `tvs_utility`

5.3 Comparison of the methods

In this part, we will compare the accuracy and time complexity of proposed methods. We should note that this two properties are not the the only criterions for quality of this methods. Here, we will test the methods for $\varepsilon = 0$, however in praxis the case of $\varepsilon > 0$ is important.

Accuracy

To test accuracy, we will first use utility function derived from traveling wave solution and parameters $\varepsilon = 0$, $r = 0$. We will compare the solution computed numerically, with semi-analytical traveling wave solution from Chapter 5.2. We use the same parameters as in paper [9]: We consider time horizon of the length $T = 10$, the computational domain in x is set to $[x_L, x_R] = [-4, 4]$. Traveling wave solution is computed for limiting values $v^- = 0.3$, $v^+ = 1.5$. In our numerical scheme we employ time-space step binding $dt = 0.1dx$. To estimate the error, we employ $L_\infty((0,T) : L_2)$-norm which is defined as

$$L_\infty((0,T) : L_2) : \|f(x,t)\| = \max_{t\in(0,T)} \left[\left(\frac{1}{x_R - x_L} \int_{x_L}^{x_R} f(x,t)^2 dx \right)^{1/2} \right] \tag{97}$$

We will compute experimental order of convergence using formula:

$$EOC_i = \frac{\ln(err_i/err_{i-1})}{\ln(dx_i/dx_{i-1})} \tag{98}$$

where err_i is estimated error of approximation for the x step of length dx_i, and err_{i-1} is estimated error of approximation for the x step of length dx_{i-1}. For `HJB_ricatti0`, we will analyze the error in φ, for all other methods we will analyze the error in V as well as in φ. We employ here the function `compute_fi`, which transforms original solution in V, to solution in φ. By estimating the error, we will not take into account $30.dx$ columns of boundary values, where the error is largest. Table 2 summarizes errors in φ of the methods, and the estimated experimental order of convergence of those methods. First part of Figure 13 illustrates the evolvement of the error φ. We plotted only the error evolvement of four methods, other versions of `HJB_parabolic0`- and `HJB_1order`-methods perform in a very similar way as the plotted versions. Table 3 summarizes the error in V. From Table 3, we can see, that the approximation of solution in V is absolutely bad, we can say that the method explodes. However, interesting is that after conversion to approximation in φ, this one is not bad at all. This means, that it makes more sense to compute or at least express the solution in terms of variable $\varphi(x,t)$. Moreover, as examined in Chapter 2, the portfolio composition, which is important for the investor, depends only on $\varphi(x,t)$. If we compare error evolvement of the approximation of solution in φ, `HJB_ricatti0`-method seems to be not so good for larger time steps as other methods. On the other hand, `HJB_ricatti0`-method implies the most consistent order of convergence. The method is empirically first order accurate with respect to norm (97) and used step-binding. We confirmed the result from [9], where a very similar (fully-implicit) method based on Riccati transformation was prooved to be empirically first order accurate. This cannot be said about `HJB_1order`- and `HJB_parabolic0`-methods where the EOC decreases rapidly. In case of `HJB_godunov1`-method is the first order accuracy also possible. To understand deeper the error behavior and determine more confidently the experimental order of convergence, it is necessary to search for causes of the error behavior. It may arise from the numerical scheme, but also from the boundary conditions or from approximations of functions $\alpha(\varphi)$, $\alpha^{-1}(\varphi)$, $\beta(\varphi)$, $\gamma(\varphi)$. In this cases, using of smaller stepsize may not help, and we need to compute better approximations of those functions, or enlarge the computation domain, so that the boundary conditions based on limiting behavior will be better justified.

Table 2: Error of numerical methods in φ-variable for varying step size and estimated order of convergence. q=1 indicates implicit version of the method, q=1/2 Crank-Nicholson version and q=0 explicit version of the method.

HJB_Method	dx:	0.1	0.05	0.025	0.0125
riccati0	ERR	1.41490E-03	5.44520E-04	2.60940E-04	1.29570E-04
	EOC		1.378	1.061	1.010
parabic0, q=1	ERR	6.38160E-04	2.16910E-04	1.52310E-04	1.33760E-04
	EOC		1.557	0.510	0.187
parabolic0, q=1/2	ERR	5.90730E-04	2.15970E-04	1.51870E-04	1.33550E-04
	EOC		1.452	0.508	0.187
parabolic0, q=0	ERR	6.07420E-04	2.15030E-04	1.51440E-04	1.33340E-04
	EOC		1.498	0.506	0.184
1order, q=1	ERR	6.38460E-04	2.16920E-04	1.52320E-04	1.33760E-04
	EOC		1.557	0.510	0.187
1order, q=1/2	ERR	5.89100E-04	2.15970E-04	1.51880E-04	1.33550E-04
	EOC		1.448	0.508	0.186
1order, q=0	ERR	6.07420E-04	2.15030E-04	1.51440E-04	1.33340E-04
	EOC		1.498	0.506	0.184
godunov1	ERR	6.26090E-04	2.90710E-04	1.49730E-04	1.31510E-04
	EOC		1.107	0.957	0.187

Table 3: Error of numerical methods in V-variable for varying stepsize. q=1 indicates implicit version of the method, q=1/2 Crank-Nicholson version and q=0 explicit version of the method.

HJB_Method / dx:	0.1	0.05	0.025	0.0125
parabolic0, q=1	1.3731E+10	1.3031E+10	1.2737E+10	1.2603E+10
parabolic0, q=1/2	1.1309E+10	1.1833E+10	1.2139E+10	1.2304E+10
parabolic0, q=0	9.3503E+09	1.0756E+10	1.1572E+10	1.2013E+10
1order, q=1	1.3731E+10	1.3030E+10	1.2737E+10	1.2603E+10
1order, q=1/2	1.1309E+10	1.1833E+10	1.2139E+10	1.2304E+10
1order, q=0	9.3503E+09	1.0756E+10	1.1572E+10	1.2013E+10
godunov1	1.8320E+10	1.5103E+10	1.3724E+10	1.3085E+10

Time complexity

Table 4 summarizes the time needed for computation of solution in φ (or computation of solution in V with conversion to solution in φ). Second part of figure 13 illustrates the results. At this point, we should make a few important remarks. The computational time clearly depends on the machine, therefore, computations may run faster (slower) on other computers. Also, very probably, some improvements in the method source code may be done to make the methods faster. For example in [9], Thomas algorithm is used to solve tridiagonal system in the numerical scheme. We just use Octave routine for computing inverse matrices. On the first sight into the table, it may look strange, that explicit versions of the methods (`HJB_parabolic0`, `HJB_1order`, q=0) don't perform for larger stepsizes better than the implicit versions (q=1, q=1/2). The

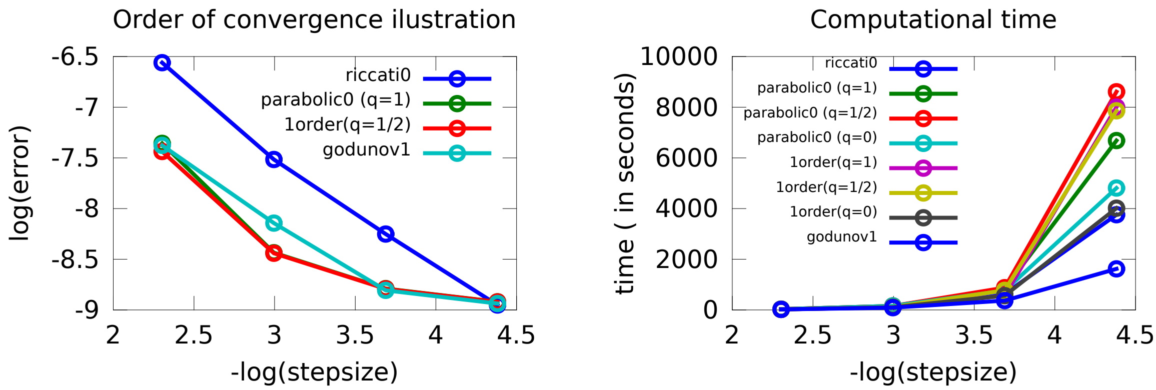

Figure 13: Left: Evolvement of error for some of the methods. Right: Time needed for computation of the solution approximation. q=1 indicates implicit version of the method, and q=1/2 indicates Crank-Nicholson version of the method

answer is, that they are not fully explicit, due to the boundary conditions. The matrix that should be inverted is not exactly identity, it has boundary conditions in first and last row. However, it is sparser than (almost) tridiagonal matrices that need to be inverted in case of implicit versions, what causes faster performance for smaller stepsize.

Table 4: Time (in seconds) needed for computation for varying stepsize. q=1 indicates implicit version of the method, q=1/2 Crank-Nicholson version and q=0 explicit version of the method.

HJB_Method / dx:	0.1	0.05	0.025	0.0125
riccati0	26.639	124.971	631.24	3758.877
parabic0, q=1	32.199	157.694	838.692	6686.022
parabolic, q=1/2	32.361	159.595	879.216	8609.735
parabolic, q=0	33.723	159.657	764.093	4808.581
1order, q=1	19.274	107.683	567.701	8004.611
1order, q=1/2	21.187	108.243	774.425	7859.558
1order, q=0	19.102	107.85	564.775	4017.364
godunov1	20.884	87.799	366.851	1619.84

Conclusion

Taking into account accuracy as well as time complexity, for larger step sizes `HJB_parabolic0-` and `HJB_1order-` methods seem to be more suitable. For smaller step sizes finite volume methods `HJB_godunov1` and `HJB_ricatti0` seems to be better. Method `HJB_godunov1` is fastest (since it is fully-explicit), however for stepsize smaller then 0.0125, `HJB_ricatti0` may be most suitable, because of most consistent experimental order of convergence. In general, explicit versions of `HJB_parabolic0-` and `HJB_1order-` methods seem to be faster. However, the next and very important criterion for the suitability of the methods will be their performance under $\varepsilon > 0$. This will be examined in Chapter 5.5.

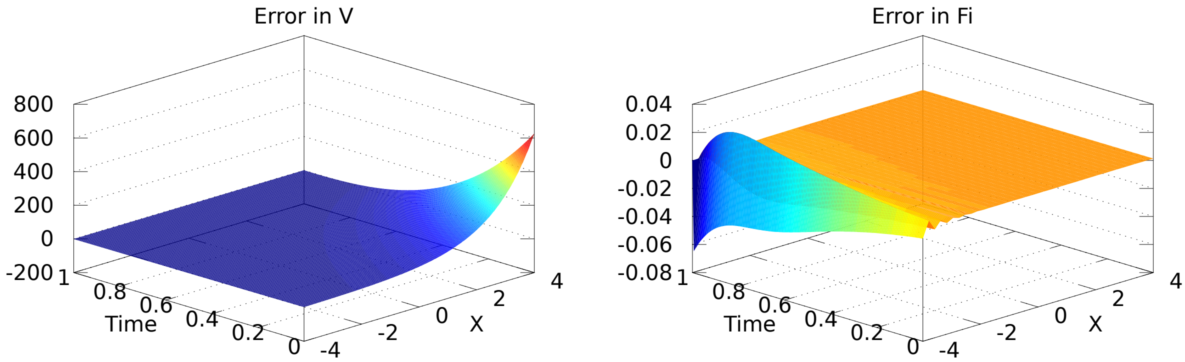

Figure 14: illustration of the error in variable $V(x,t)$ and in variable $\varphi(x,t)$. TVS utility function is used in both cases.

5.4 Analysis of the error behavior

In this part, we will illustrate a few cases of error behavior. Some of the errors lead to strongly biased solution, other can lead to complete failure of the method. We will use stepsizes $dx = 0.02$, $dt = 0.00625$ and time horizon $T = 1$.

Error in V and in Fi

In previous chapter, we have seen, that the error of approximation of the solution in $V(x,t)$ is pretty large, in contrast to approximation of solution in $\varphi(x,t)$, where the error of acceptable size. Interesting is, that for the solution in V, the error is largest in the first time layer (that is computed as last) and on the right boundary, in contrast to solution in φ, where the error is present on the left boundary, and is significantly smaller in the first time layers. This error behavior is illustrated in Figure 14. We used utility function implied by traveling wave solution in φ (`tvs_utility`). In shorter time horizon, the error in V is significantly smaller. For example, if we take one month instead of one year, the error in $L_\infty((0,T) : L_2)$-norm will be 0.193 instead of 195.65 .

Error in case of first approach to boundary conditions

In section 3.4, we proposed an approach to boundary conditions based on approximation for the solution in V behind the boundaries by traveling wave. However, we also explained how those boundary conditions fail in case of simple exponential utility function $U(x) = -e^{-x}$. First part of figure 15 illustrates the error in φ, when this approach (method `HJB_godunvx`) is used. Original solution should be constant ($\varphi \equiv 2$). If we use proper boundary conditions (Chapter 3.5, method `HJB_godunv1` we get constant solution with value 1.9967. (which can be improved using smaller step sizes).

Utility functions with large Fi

For utility functions, which are characteristic by rapid slope growth and large φ, such as $U(x) = -e^{-10x}$, most of our methods are not suitable. For this method, φ will explode, causing failure of the interpolation of $\alpha(\varphi)$ and we end up with no numerical solution. However, for this case, method with exponential correction `HJB_parabolic1`

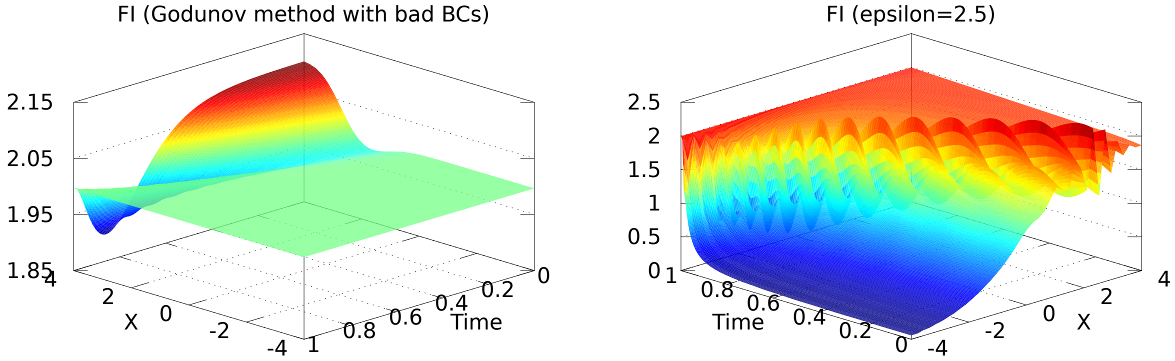

Figure 15: Left: φ in case of the first approach to boundary conditions. We can see how the error is spreading from the right boundary (true φ should be constant). Right: φ in case of HJB_parabolic0 method with $\varepsilon = 2.5$. The error propagates in form of waves.

was developed in Chapter 4.3. After setting parameter $K = 9$, the method computes solution for $U(x) = -e^{-10x}$ successfully. For utility function $U(x) = -e^{-5x}$ also other methods works, however, if we set $\varepsilon > 0$, again only the `HJB_parabolic1`-method remains feasible.

Case of positive epsilon

Some of the proposed methods work well in cases of $\varepsilon = 0$, but show some error behavior, or even completely fail in cases of $\varepsilon > 0$. The failure or success in this cases depends on many factors and we will shortly analyze them in the next part. Here, we just illustrate one of the forms of error behavior in case of positive ε ($\varepsilon = 2.5$). On the left picture in Figure 15, we see approximation of solution in φ for terminal condition $U(x) = -e^{x}$ computed with `HJB_parabolic0`-method with parameter $q = 1$. We can observe the error in the form of waves, propagating from left corner of the terminal time layer. The source of this error behavior is probably the heuristic approach to boundary condition in `HJB_parabolic0`-method, where we take the second most-left value of φ from the previous layer as the most-left (boundary) value in the new time layer. We should note, that using finer stepsizes will fix this particular problem -the solution will be smoother.

5.5 Performance of methods in case of positive epsilon

Up to now, we tested our methods just for the case of no money inflow ($\epsilon = 0$). However, the ability of the methods to compute the solution also in case of $\epsilon > 0$ is important in many cases. In this part, we will test the performance of the methods in case of five different money inflow rates ϵ: $\epsilon = 0.05, 0.1, 0.5, 1, 5$. We will use exponential utility function $U(x) = -e^{-x}$ (`exp_utility`), time horizon T=10, computational domain $[x_L, x_R] = [-4, 4]$ and x-stepsize $dx = 0.1$. We use two different time stepsizes $dt = 0.01$ and 0.001. We test, if the method returns some approximation of solution, or if it fails. To be sure, that some method does not return an result that is completely different from results from other methods, we compute for each method mean value of the solution in φ. Table 5 summarizes the results.

Table 5: Performance of the methods with respect to two different time step sizes, and four different rates of money inflow ε. In cases when the method returns some approximation of solution, we state the mean value of the solution in the table. In cases when the method fails, we write "-" instead of it.

HJB_Method	dt/epsilon:	0.05	0.1	0.5	1	5
riccati0	0.01	1.83838	-	-	-	-
	0.001	1.8384	1.73851	1.38631	-	-
explicit	0.01	1.82078	1.71838	-	-	-
	0.001	-	-	-	-	-
godunov0	0.01	1.82673	1.72635	-	-	-
	0.001	-	-	-	-	-
godunov1	0.01	1.82724	1.72733	-	-	-
	0.001	1.82742	1.7276	1.40772	1.24958	-
1order, q=0	0.01	1.82768	-	-	-	-
	0.001	1.8266	1.7221	1.36109	1.15952	-
1order, q=1/2	0.01	1.82811	1.72495	1.37243	-	-
	0.001	1.82666	1.72219	1.36136	1.1599	0.59319
1order, q=1	0.01	1.82854	1.72567	1.37451	1.18195	0.64467
	0.001	1.82671	1.72228	1.36162	1.16029	0.59373
parabolic0, q=0	0.01	1.82768	-	-	-	-
	0.001	1.8266	1.7221	1.36109	1.15952	-
parabolic0, q=1/2	0.01	1.82811	1.72495	1.37243	-	-
	0.001	1.8267	1.72224	1.36133	1.15991	0.59319
parabolic0, q=1	0.01	1.82854	1.72567	1.3745	1.18194	0.64467
	0.001	1.8267	1.72236	1.36165	1.16027	0.59373

Utility functions with large Fi

In previous chapter, we examined difficulties by solving HJB equation with terminal condition in form of utility function with large φ. In that case, using `HJB_parabolic1` method with exponential correction was needed. Adding money inflow rate $\varepsilon > 0$ makes to the equation, makes the situation even more difficult. However, in many cases `HJB_parabolic1` works well also for $\varepsilon > 0$. The succes or failure of the method depends on more factors: value of ε, time horizon T, stepsizes dx, dt and also on the utility function. The choice of correction parameter K for the method is also important. Here we summarize some cases, when the `parabolic1` method returns successfully some approximation of the solution:

- Time horizon $T = 10$, utility function $U(x) = e^{-5x}$, $\varepsilon = 1$, parameter $K = 5.5$.
- Time horizon $T = 10$, utility function $U(x) = e^{-10x}$, $\varepsilon = 0.1$, parameter $K = 10.5$.
- Time horizon $T = 1$, utility function $U(x) = e^{-5x}$, $\varepsilon = 5$, parameter $K = 4$.
- Time horizon $T = 1$, utility function $U(x) = e^{-10x}$, $\varepsilon = 0.5$, parameter $K = 9$.

In all cases we used computational domain $[x_L, x_R] = [-4, 4]$, x-stepsize $dx = 0.1$ and time stepsize $dt = 0.001T$.

Conclusion

In case of large ε implicit versions (q=1) seem to be most reliable. However, with respect to their time complexity (Chapter 5.3), we should use other methods, if feasible. Explicit methods without boundary conditions (`HJB_explicit`, `HJB_godunov0`) fail in cases of smaller time steps. The reason is too large computationl domain in the last time layer. The explicit method with boundary condition `HJB_godunov1`, provide us for larger ε with approximation of solution, which may be different from results of other methods. We see it on slightly different mean values of the solution for $\varepsilon = 0.5$, $\varepsilon = 1$. This can indicate some relevant bias in the method, possibly due to it's explicit character. On the other hand, we should note, that the boundary conditions for `HJB_godunov1`- method as well and `HJB_riccati0`- method are better argumented than the approach in `HJB_1order`- and `HJB_parabolic0`- methods. These method can also compute the solution much faster, if large number of space and time steps is needed. By comparing precision, experimental order of convergence computational time and suitability for computations with positive money inflow rate, Crank-Nicholson versions of the `HJB_1order`- and `HJB_parabolic0`- methods are in no way the methods of choice.

6 Qualitative analysis of the solutions of HJB equation

In this chapter, we will take a closer look on the solution of Hamilton-Jacobi-Bellman Equation. We will analyze, how the changes in parameters of the equation affect the solution, and we will try to interpret changes in solution economically. We will examine both solutions in variable $V(x,t)$ and in variable $\varphi(x,t)$, since they provide us with different informations. At first, let us recapitulate some facts about this variables to understand the interpretations better:

- Since $V(x,t)$ represents value function, solution in $V(x,t)$ gives us the information, how valuable is the amount of money e^x in time t for the investor. This subjective value depends on the utility, that the investor expects in the final time under the condition, that he will use the optimal investment strategy. Therefore, if the value function $V(x,t)$ increases due to some change in parameter, it means that the investor expects higher utility at the final time.

- According to Chapter 2, the composition of the portfolio depends only on parameter φ. Therefore, $\varphi(x,t)$ provides us with information about the optimal portfolio composition in time t, if the portfolio value (in money units, not subjective) is equal to e^x. As also examined in Chapter 2, for small φ, the portfolio composition is more return-oriented, that means, most of the money are invested in assets with high return (which are often most risky). For larger φ, portfolio composition is more conservative, the portfolio is more diversified and assets with low volatility are more important. Therefore, increase in φ means more more conservative investing.

Solutions of HJB Equation for different situations, which will be presented now, are computed with `HJB_parabolic0`-method with parameter $q = 1$, time horizon $T = 1$, computational domain $[x_L, x_R] = [-4, 4]$ and stepsizes $dx = 0.08$, $dt = 0.01$.

6.1 Changes in model

Benchmark solutions

Figure 16 illustrates solution in $V(x,t)$ for exponential utility function $U(x) = -e^{-x}$ (`exp_utility`) and solution in $\varphi(x,t)$ for utility function implied by traveling wave solution in $\varphi(x,t)$ (TVS utility function, `tvs_utility`). We used $r = 0$, $\varepsilon = 0$. Both of this solutions are simple traveling waves. In case of $V(x,t)$, the wave travels from left to right with time. Therefore, $V(x,t_0)$ in present time t_0 is greater then $V(x,t_1)$ in some future time t_1. This can be interpreted as the expectation of greater utility in future, if optimal strategy is used. Let us note, that, as we can see, the wave travels from left to right also in the case of TVS utility function with solution in φ.

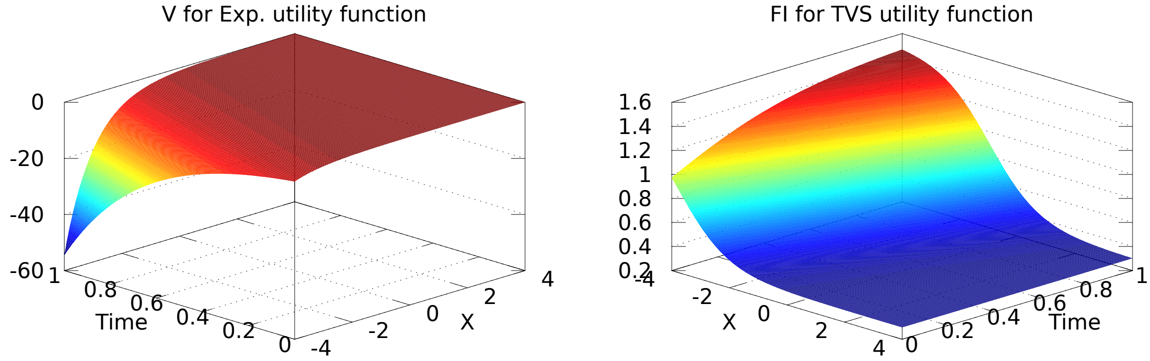

Figure 16: Left: Solution of HJB equation in variable $V(x,t)$ for exponential utility function. Right: Solution of HJB equation in variable $\varphi(x,t)$ for TVS utility function.

Change in covariance matrix

Now, we will examine, how the change in covariance matrix of the assets Σ affects the shape of the solution. In the first part of Figure 17, we can see solution in V for exponential utility function, with Σ 100 times larger. The solution is also traveling wave, however, now is the wave traveling from right to left with time. Therefore, $V(x,t)$ is in this case increasing with time. This is caused by switch in sign of $\alpha(2)$, the speed of traveling wave for exponential utility function. The second part of Figure 17, illustrating solution in φ of HJB Equation for TVS utility function with 300 times larger Σ, presents similar behavior: again, the wave travels from right to left, in contrast to the solution with original Σ. This is again caused by switch in sign of the wave speed (see Chapter 5.2) $c = \frac{\alpha(v^+)(1-v^+)-\alpha(v^-)(1-v^-)}{v^+-v^-}$.

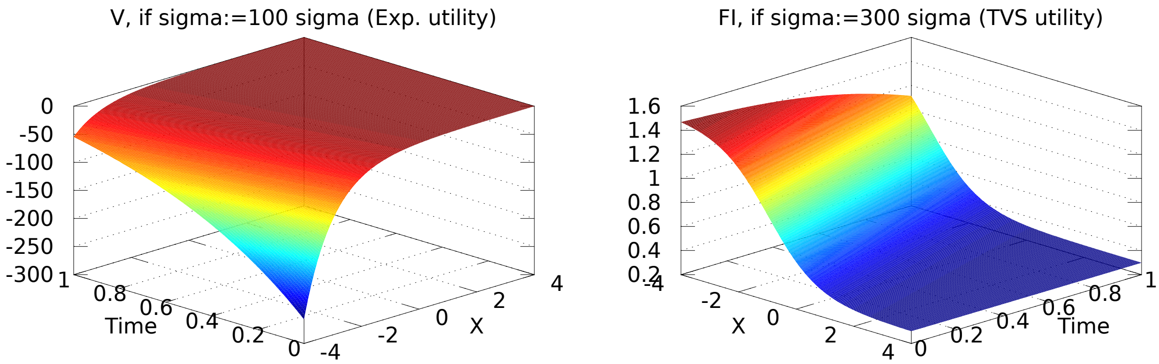

Figure 17: Left: Solution of HJB equation in variable $V(x,t)$ for exponential utility function, in case of covariance matrix Σ multiplied by 100. Right: Solution of HJB equation in variable $\varphi(x,t)$ for TVS utility function, in case of Σ multiplied by 300.

Economic interpretation: For large Σ, $V(x,t_0)$ in present time t_0 is smaller than $V(x,t_1)$ in some future time t_1. This means, we expect decrease in our utility in future. This is caused by big volatility, causing big spread of possible future outcomes, despite of using optimal strategy. This spread can result in higher or lower return than expected, however, our concave utility functions put more weight to decrease in return than to increase, therefore we expect smaller utility.

Change in model constraints

Now, we will take a look on the change of value function $V(x,t)$ in case of changed constraints for weights θ. In our original model, that we used up to now, all weights must be non-negative, and sum up to 1. Now we will consider two other models:

1. Model with allowed short positions, where we can have also negative weights, but not smaller than -1.

2. Model with forced diversification, where we demand, that each weight of asset is at least 0.02.

In both new models, weights should some up to 1. Figure 18 illustrates the differences between $V(x,t)$ for these new models and the $V(x,t)$ for original model. In these new models, function $\alpha(\varphi)$ changes and therefore also the speed of traveling wave in $V(x,t)$-solution. As we can see in Figure 18, the difference is largest on the left boundary, where the utility function (and therefore also value function) grows most rapidly.

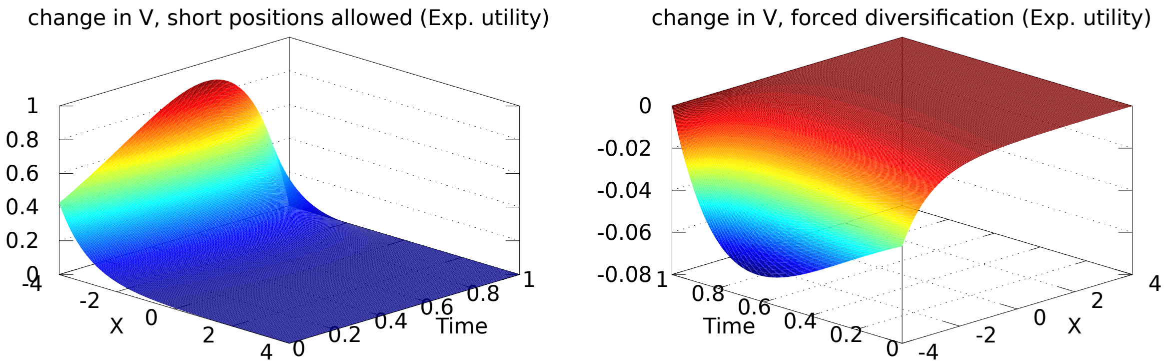

Figure 18: Left: Difference between the solutions in $V(x,t)$ of HJB equation for model with allowed short position and for standard model. Right: Difference between the solutions in $V(x,t)$ of HJB equation for model with forced diversification and for standard model. In both cases exponential utility function is used.

Economic interpretation: In model with allowed short positions, the investor has more possibilities, how to allocate money in the assets, than in original model, therefore he can use better strategy, leading to higher expected utility in any time. Therefore, value function $V(x,t)$ in case of the model with allowed short positions is bigger than $V(x,t)$ in case of original model. On the other hand, if we prescribe how the investor should invest some amount of money (case of forced diversification), the investor will have less possibilities, how to invest, and the optimal strategy from original model, leading to highest expected utility, may not be feasible. Therefore, $V(x,t)$ in this case will be lower then in case of original model.

6.2 Changes in parameters epsilon, r

Adding money inflow

Up to now, we was examining solutions of HJB Equation for $\varepsilon = 0$. Now, we will take a look on the change of solution, if constant money inflow $\varepsilon = 1$ is added. Figure 19 illustrates the behavior of the solutions for exponential utility function. From the first picture, we can deduce higher $V(x,t)$ in case of $\varepsilon = 1$ than, in case of $\varepsilon = 0$, the second picture show us solution in φ, which is not constant anymore. In Figure 20, we illustrate solution in φ and its change in case of $\varepsilon = 1$ for TVS utility. Both solutions in φ (for exponential, as well as for TVS utilty) are smaller in case of $\varepsilon = 1$ then in case of $\varepsilon = 0$.

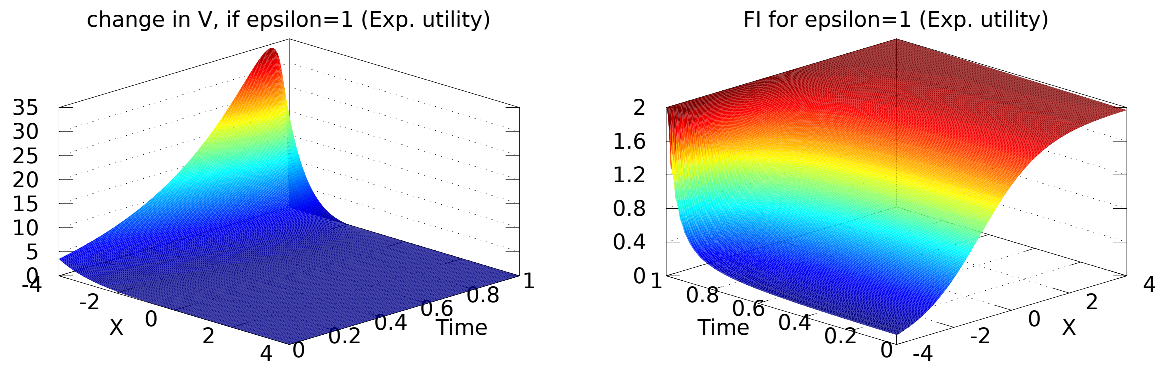

Figure 19: Left: Difference between solutions in $V(x,t)$ of HJB equation with money inflow rate $\varepsilon = 1$ and without it ($\varepsilon = 0$). Right: Solution of HJB equation with $\varepsilon = 1$ in variable $\varphi(x,t)$. In both cases exponential utility function is used.

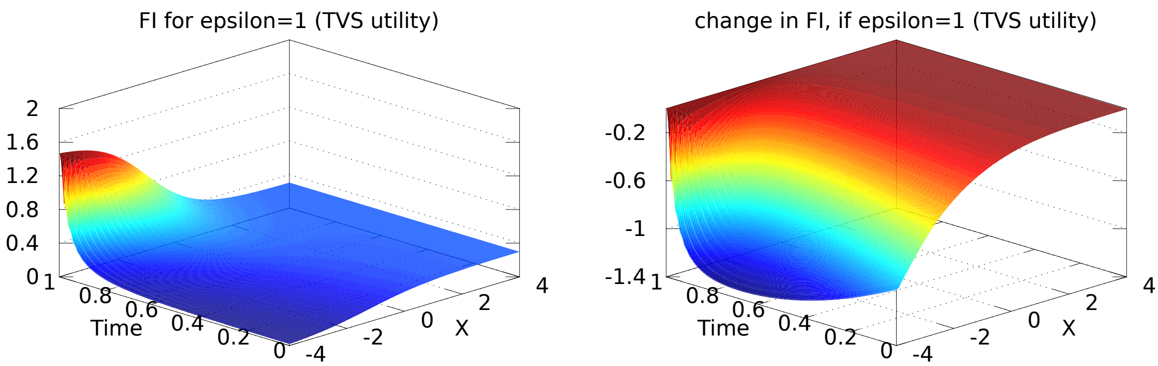

Figure 20: Left: Solution of HJB equation with $\varepsilon = 1$ in variable $\varphi(x,t)$. Right: Difference between solutions in $\varphi(x,t)$ of HJB equation with $\varepsilon = 1$ and with $\varepsilon = 0$. In both cases TVS utility function is used.

Economic interpretation: If there is positive money inflow to the portfolio, investor can achieve greater utility in final time, therefore $V(x,t)$ is higher in this case. The change is most obvious for small portfolio values x (left boundary), since for this values brings the increase of portfolio value due to money inflow biggest increase in utility (because of concavity of utility function). However, the increase is smaller in early times. The reason is, that in earlier times, investor has enough time to improve the utility also without money inflow by using optimal investing strategy. In that case, the additional money inflow will not help much, because, again due to concavity, for larger portfolio values, the increase in value of the portfolio means only small increase in utility. The increase in $V(x,t)$ just before final time is also smaller, because not much time to utilize the money inflow is left. Now, let us try to interpret change of solution in φ. Since the investor has stable inflow of money, he do not need to fear losses so much, and he can do more risky investments (represented by smaller φ). We can also see that most risky investments are done for small portfolio values -left boundary, since the possible improvement of value function is largest there (due to concavity). Near the right boundary increase of portfolio value caused by risky investments will not change the utility as much as the possible decrease in portfolio value -therefore more conservative portfolio composition represented by higher φ is chosen.

Increase in interest rate r

Let us have a look, how the increase in interest rate r affects the shape of the solution. Up to now, we used interest rate $r = 0$, now we will use $r = 1$. Other parameters are set as in benchmark solution ($\varepsilon = 0$). Figure 21 illustrates the differences between solution in V for exponential utility function (or solution in φ for TVS utility function) with $r = 1$ and the benchmark solution with $r = 0$. Again, we can observe increase in value function $V(x,t)$ (for exponential utility) and decrease in solution in $\varphi(x,t)$ for TVS utility function. However, φ for exponential utility will remain constant. We should realize, that the solution in V for exponential utility remain in the form of traveling wave, only the speed changed from $\alpha(\varphi)$ to $\alpha(\varphi) - r$.

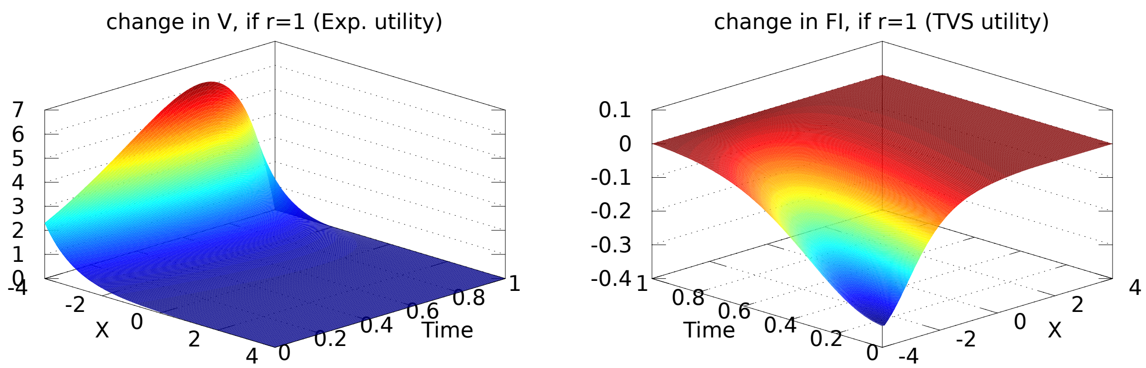

Figure 21: Left: Difference between the solutions in $V(x,t)$ of HJB equation with positive interest rate $r = 1$ and with zero interest rate ($r = 0$) for exponential utility function. Right: Difference between the solutions in $\varphi(x,t)$ of HJB equation with positive interest rate $r = 1$ and with zero interest rate ($r = 0$) for TVS utility function.

Economic interpretation: At first, we should note, that since μ is defined as vector of differences between drifts of assets and risk-free interest rate r, by increasing r we increase drifts of assets ($\mu + r$). This means increase in expected return of assets. Therefore, bigger future portfolio value and bigger utility in final time can be expected, causing increase in value function $V(x,t)$. Again, due to concavity, biggest increase can be seen on the left boundary. This situation is completely similar to the situation of loosening constraints (allowing short positions), since formally, we just change the wave speed. On the other side, the interpretation of change in φ for TVS utility function is similar to the case of changing $\varepsilon = 0$ to $\varepsilon = 1$. We expect stable higher return, so we can make more risky investments. Most risky investments are done for small x (near to left boundary), since in that case, we can improve the utility function by possible high return most remarkably.

Conclusion

Main topic of this work was numerical solving of Hamilton-Jacobi-Bellman (HJB) equation discussed in [9]. This partial differential equation arises from stochastic time-continuous optimal control theory, which is in our case applied to solve portfolio optimization problem. We presented both explicit and implicit methods for solving this equations. Solving this equation is bound with several problems, since the equation is fully nonlinear. The approach we applied to tackle this problem was based on studying standard numerical methods, for example methods for conservative systems (Godunov method, see [16]), or methods for parabolic PDE's (see [21]), and for first order PDE's and then trying to modify this methods so they will be to some extent suitable for our HJB Equation. We implemented this modified methods in Octave and tested them with parameters estimated from real data. For comparison, we tested also method based on Riccati transformation proposed in [9].

One of the most important problems was setting boundary conditions for our numerical scheme. For explicit methods we have the possibility to avoid using boundary conditions at all, by computing in each time layer only values for which we have enough data. By this approach, the computational domain is shrinking, so we need to start with larger computational domain in the first (in our case terminal) layer. We implemented this approach for both Modified Godunov method and simple explicit method, arising just from substituting the derivatives by finite differences. The advantage of this approach is its simplicity and no need to do approximations on the boundary. The disadvantage is that implicit methods which proved to be more stable are not feasible in this case. Another disadvantage is failure of the methods in case of large number of time steps. Therefore, we decided to develop boundary conditions. We presented two reasonable approaches to boundary conditions, based on limit behavior of the solution. We showed, where one of this approaches fails to provide satisfactory outcome, and analyzed the cause of it. We implemented the second approach, which was successful to our methods. This approach is well-reasoned in case of parameter $\varepsilon = 0$. In case of positive parameter ε this approach is legitimate just in case of Modified Godunov method. Using this approach in other methods (with small changes) should be regarded as rather heuristic. Another approach to tackle the problem of positive ε was presented in [9]. However, it can be implemented only in the case of HJB Equation transformed by Riccati transformation.

By testing our methods we focused on three main criteria: Approximation error (and experimental order of convergence), time complexity, and suitability for case of positive ε. We looked at solutions in both original variable $V(x,t)$ and in transformed variable $\varphi(x,t) = 1 - \partial_x^2 V(x,t)/\partial_x V(x,t)$. All tested methods showed very large error of the approximation in $V(x,t)$, and no convergence was present in this case. However, the error of solution in transformed variable $\varphi(x,t)$ was in reasonable bounds and the solution seemed to be converging with better discretization. We also confirmed the experimental order of convergence of the method based on Riccati transformation from [9]. Order of convergence for other methods was decreasing which could be caused by different factors. Despite of decreasing order of convergence, implicit methods based on first order PDE form and parabolic PDE form showed smallest errors in our tests.

However, this methods were also the slowest. The method, that needed least time for computation was Modified Godunov method. For solving HJB equation with $\varepsilon = 1$, implicit methods were again best performing. However, we should note, that they use boundary conditions, that are not well-reasoned, which can lead to biased solution. For cases of large φ, where standard methods often fail to compute solution, we developed method with exponential correction based on parabolic PDE form of HJB equation.

One part of Hamilton-Jacobi-Bellman equation is in fact quadratic programming problem. Solving this problem provide us with information, how the optimal portfolio composition depends on variable $\varphi(x,t)$. Paper [9] presented important result about the form of solution of this quadratic programming problem. However, it was concerned only with constrains in form of standard simplex. In Chapter 2 we generalized and proved this result for any linear constrains. We illustrated portfolio composition for three different set of constrains, each having different economic interpretation.

In Chapter 6, we employed one of our numerical method for solving HJB Equation for different parameters and model constrains. We plotted the solutions and examined the changes in their shapes for different parameters. We interpreted these changes economically.

This topic offers lot of space for further investigation. Since, as we examined, numerical solution in $V(x,t)$ is strongly biased and non-converging, it has more sense to search for solution in variable $\varphi(x,t)$. In that case, Riccati transformation of the original equation proposed in [9] can be useful. Moreover, this solution provide us with the information, that is for the investor most important -the portfolio composition. All methods, that we tested, can be classified as finite difference methods or finite volume methods. Another way would be to use finite elements methods. Also our short analysis of performance of the methods offers lot of space for future improvement. We need to know, where the largest error is generated (boundary conditions, approximation of function $\alpha(\varphi)$), to know, how to improve the methods and achieve more satisfying experimental order of convergence.

References

[1] Applebaum, D.: *Lévy Processes and Stochastic Calculus*, Cambridge University Press, 2009

[2] Bellman, R.: *Dynamic Programming* Princeton University Press, 1957

[3] Finance.yahoo.com: *Yahoo! Finance - Business Finance, Stock Market, Quotes, News*, internet database of historical asset prices http://finance.yahoo.com/

[4] Godunov, S. K.: *A difference method for numerical calculation of discontinuous solutions of the equations of hydrodynamics*, Matematicheskii Sbornik 47(89):3 (1959), 271-306

[5] Halická, M., Brunovský P., Jurča, P.: *Optimálne riadenie*, EPOS, Bratislava, 2009

[6] Halická, M., Jurča, P.: *Optimálne riadenie II*, Lecture text, Comenius Univeristy, Bratislava, 2012

[7] Hamala, M., Trnovská, M.: *Nelineárne programovanie*, EPOS, Bratislava, 2012

[8] Ishimura, N., Ševčovič, D.: *On traveling wave solutions to a Hamilton-Jacobi-Bellman equation with inequality constraints*, Japan Journal of Industrial and Applied Mathematics (2013), 51-67.

[9] Kilianová, S., Ševčovič, D.: *Transformation Method for Solving Hamilton-Jacobi-Bellman Equation for Constrained Dynamic Stochastic Optimal Allocation Problem*, Australian and New Zealand Industrial and Applied Mathematics Journal, Vol 55 (2013), 14-38

[10] Kilianová, S.: *Stochastic Dynamic Optimization Models for Pension Planning*, Disertation Thesis, Faculty of Mathematics, Physics and Informatics, Comenius University, Bratislava, 2008

[11] Kossaczký, I.: *Consistency of Utility Theory with Mean-Variance Optimization under the Assumption of Normally Distributed Returns*, Proceedings of the Student Science Conference 2012, Faculty of Mathematics, Physics and Informatics, Comenius University, Bratislava, 10-13

[12] Kossaczký, I.: *Markowitz multiperiod model*, Bachelor Thesis, Faculty of Mathematics, Physics and Informatics, Comenius University, Bratislava, 2012

[13] Kuhn, H. W., Tucker, A. W.: *Nonlinear programming*, Proceedings of the Second Berkeley Symposium on Mathematical Statistics and Probability, 1950, pp. 481-492

[14] Kútik, P., Mikula, K.: *Finite volume schemes for solving nonlinear partial differential equations in financial mathematics*, Finite Volumes for Complex Applications VI, Problems and Perspectives, Springer Proceedings in Mathematics, volume 4, (2011), pages 643–651.

[15] Lamoš, F., Potocký, R.: *Pravdepodobnosť a matematická štatistika -štatistické analýzy*, Comenius University, Bratislava, 1998

[16] Leveque, R.: *Finite Volume Methods for Hyperbolic Problems*, Cambridge University Press, 2002

[17] Macová, Z., Ševčovič, D.: *Weakly nonlinear analysis of the Hamilton-Jacobi-Bellman equation arising from pension savings management*, International Journal of Numerical Analysis and Modeling, 7(4) 2010, 619-638

[18] Markowitz, H.: *Portfolio selection*, Journal of Finance 7 (1952), 77-91

[19] Melicherčík I.- Olšarová L.- Úradníček V.: *Kapitoly z finančnej matematiky*, EPOS, Bratislava, 2005

[20] von Neumann, J., Morgenstern, O.: *Theory of games and economic behavior*, Princeton university press, 1944

[21] Pulch, R.: *Numerical Analysis and Simulation of Partial Differential Equations* Lecture text for Summer Term 2011, University of Wuppertal, 2011

[22] Seydel, R.: *Tools for Computational Finance*, Springer-Verlag Berlin Heidelberg, 2009

[23] Siede, H.: *Multi-period portfolio optimization with emphasis on a mean-variance criterion*, Doctoral thesis, Universität St. Gallen, 2000

[24] Sondermann, D.: *Introduction to Stochastic Calculus for Finance*, Springer-Verlag Berlin Heidelberg, 2006