# Penalty Methods for Nonlinear Problems in Financial Option Pricing

by

Ruining (Ray) Wu

A thesis submitted in conformity with the requirements
for the degree of Master of Science
Graduate Department of Computer Science
University of Toronto

# Abstract

Penalty Methods for Nonlinear Problems in Financial Option Pricing

Ruining (Ray) Wu
Master of Science
Graduate Department of Computer Science
University of Toronto
2021

We study the pricing of nonlinear problems in computational finance by numerical Partial Differential Equation (PDE) methods. We examine the numerical solution obtained via the use of Policy Iteration methods from Hamilton-Jacobi-Bellman equations and introduce new penalty-like iterative methods to solve the nonlinear equations. We consider the following problems: Unequal Borrowing and Lending rates, Stock Borrowing Fees, Stock Borrowing Fees with American exercise rights, Uncertain Volatility, and Transaction cost models. We demonstrate second-order convergence of the solution and of the Greeks, and for the nonlinear problems we study the number of nonlinear iterations taken as a proxy to the total computational cost. Furthermore, the effects of various details such as different boundary conditions and nonuniform discretization grids are studied. Finally, where applicable, we prove monotonicity of the new penalty-like methods that we have introduced and demonstrate that our numerical implementations uphold this property.

# Acknowledgements

First and foremost, I would like to thank my supervisor Professor Christina C. Christara for her invaluable teaching, supervision, and support throughout my Master's degree, and Professor Kenneth R. Jackson for his teaching and his role as the second reader of this thesis. Additionally, I would also like to thank the Ontario Graduate Scholarship and the Department of Computer Science at University of Toronto for providing me with generous financial support throughout my degree. Finally, I would like to thank my parents for their love and support.

# Contents

# List of Tables

# List of Figures

# Chapter 1

# Introduction

Since the development of the Black-Scholes model [6] in 1973, the practice of option trading and the study of the pricing of financial options have gained mathematical legitimacy around the world, leading to a boom in options trading and the accompanying research. The Black-Scholes model is still used by practitioners today with adjustments. These adjustments are typically nonlinear in nature and thus result in nonlinear partial differential equations, which are typically solved not analytically but with numerical methods, the focus of this thesis.

One such adjustment is the introduction of controls and transformation of the Black-Scholes equation to a controlled PDE. This leads to Hamilton-Jacobi-Bellman equations and solving the associated problems in optimal control. We study the numerical solution of nonlinear partial differential equations (PDEs) in computational finance that arise from optimal control problems, a natural extension of the Black-Scholes model.

The study of optimal control and Hamilton-Jacobi-Bellman (HJB) equations has a long and rich history [2, 3], with numerous applications such as self-driving cars [34] and human navigation [27] in addition to computational finance. Some methods developed for problems in computational option pricing are policy iteration algorithms [14, 17] and a "piecewise constant policy iteration" from [14, 31] among others. In recent years, there is a trend towards the use of deep learning [19] and neural network approaches [8], especially for high-dimensional problems.

In this thesis, we take a finite difference approach to the discretization of derivatives and study options that have only one underlying asset. The main contributions of this thesis are twofold:

1. The formulation of Penalty and Penalty-like methods that efficiently solve the nonlinear PDEs arising from these adjustments. These methods can handle nonlinearity in the solution, first, and second derivatives.

2. Proofs that demonstrate their monotonicity and convergence, when such behaviour is observed.

3. Improved policy iteration methods which incorporate variable timestepping for Amerian options, which highly improves the efficiency of the algorithm and gives rise to more stable convergence of both the solution and the Greeks for American options.

Some of our minor contributions are code profiling results showing the computational time required for our algorithms and the distribution of time spent to justify our cost analysis, the testing of numerical consistency between penalty-like (penalty) methods and policy algorithms and different boundary conditions where applicable, the benefits of using a nonuniform grid in spatial discretization, improving the order of convergence with Richardson extrapolation, and the extension of variable timesteps as suggested in [16] for exotic options with an American-style exercise right.

In addition to the standard problems of European and American option pricing that we study under the Black-Scholes framework, we also study the following nonlinear control problems formulated as Hamilton-Jacobi-Bellman (HJB) equations

- The unequal borrowing and lending interest rate problem [4]

- The above problem with Stock Borrowing Fees [13]

- The Stock Borrowing Fee problem with American exercise rights [14]

- The Uncertain Volatility model [1]

- Transaction cost models: While the initial transaction cost model [24] is identical to the uncertain volatility model, subsequent debate and refinement [22, 38, 25, 35] has led to a slightly different model [37, 35] which we use. The debate on the merits and drawbacks of these models are beyond the scope of this thesis, which focuses on the numerical methods used to solve these models.

- the above with American options [37].

There problems are approximately in order of increasing nonlinearity, that is, more nonlinear iterations are expected to be required to solve the later problems than the earlier problems.

Previous work such as [14] and [17] have used a discrete policy iteration for the numerical solution of these HJB problems. Our approach is a penalty-like (penalty) method similar to [7] and [16]. However, to our knowledge there has not been a previous generalization of penalty matrices to handle nonlinearities in solution and all derivative terms; in [29] a second-derivative nonlinearity is considered, but it is not given in a general form for any derivative and it is not given in the context of American options. In this respect our work is novel.

One of the advantages of penalty method (as opposed to policy method) is that we do not need to enumerate all possibilities of the controls, which is particularly expensive when we have an increasing number of controls.

In this thesis, we start off with an overview of existing algorithms for financial option pricing, such as Crank-Nicolson for European options, the discrete penalty method [16] for American options, and the policy iteration algorithm mentioned earlier. We also discuss details such as the use of nonuniform discretization grids, which are useful in improving the distribution of the error as a function of the spatial variable, and the use and implementation of different types of boundary conditions. Next, the thesis discusses the nonlinear problems and our novel contributions.

In addition, we also use techniques for the penalty iteration method such as [16] and use them to improve the policy method given in [14].

Similar methods to ours have been developed such as [32] for more general problems of HJB variational inequalities. However, our contributions differ from [32] in two aspects:

1. Our method is globally second order convergent while the algorithm proposed in [32] is first order convergent.

2. Our method addresses nonlinear terms involving partial derivatives in a different way, resulting in very efficient iteration techniques.

This thesis is organized as follows: Chapter 2 presents the formulation of the various pricing problems that we consider and solve, both of a control formulation and as a nonlinear PDE formulation. Chapter 3 presents the numerical methods which we use to solve the problems and proves desirable properties of the numerical methods, when applicable. Chapter 4 presents and discusses the results from our numerical experiments, and we conclude with Chapter 5.

# Chapter 2

# Problem Description

This chapter describes the problems that are studied in this thesis: The pricing of European options under the Black-Scholes framework, the pricing of American options as a penalized PDE, and several other option pricing problems under an optimal control framework.

## 2.1 Introduction to Financial Options

A financial option is a contract between two parties, the writer and the holder, in which the writer grants the holder the right to buy or sell an asset for a predetermined price over a specified time interval. Specifically, a call option gives the holder the right to buy the asset for the strike price $K$, and a put option gives the right to sell at the strike price $K$. Options also differ by the permitted time of exercise: For a European option the only permitted time of exercise is at the end $T$ of the prescribed maturity time, whereas for an American option, the holder can exercise the option at any time $t \leq T$.

Since the holder has the choice to not exercise the option if it is not a good financial decision, the option has a nonnegative value associated with it. The accurate computation of this value is a key problem in computational finance.

Consider a call option at the time of expiry. Denote the value of the asset (known as the underlying) as $S$. If the value of $S$ at the end of the time interval is greater than the strike price $K$, then it is beneficial for the holder to exercise the option, and gain a benefit of $S - K > 0$. Conversely, if $S$ is less than $K$, then the holder would not use the contract, and not receive any benefit. Therefore, the payoff $V^*$ of a call option is equal to $\max(S - K, 0)$.

Analogously, the payoff of a put option is $\max(K - S, 0)$.

In our Partial Differential Equation (PDE) models, we start at the known conditions (payoffs) and integrate backwards in time $t$ by applying the transformation $\tau = T - t$. Therefore, the payoffs (which occur at the end of the time of interest) are our initial conditions at $\tau = 0$, and we are solving for the fair price at the initial time $t = 0$ or $\tau = T$.

## 2.2 Black-Scholes and European options

We start with European options, which are considered the simplest: there is only one permitted time of exercise, which is at the time of expiry. In addition, for options with one underlying and call and put payoffs, we have exact closed-form solutions.

The Black-Scholes PDE, which describes the evolution of the value $V(\tau, S)$ of a European option under the Black-Scholes model [6], is

$$V_\tau = \frac{\sigma^2 S^2}{2} V_{SS} + rSV_S - rV \equiv \mathcal{L}(r)V \tag{2.1}$$

where $\sigma$ is the volatility (a measure of the variance in a market), and $r$ is the bank's interest rate. Let also $V^*(S) = V(\tau = 0, S)$ denote the payoff of the option, which also acts as initial condition for the PDE (2.1). Throughout the thesis, $V_\tau$ denotes $\partial V/\partial\tau$, $V_S$ denotes $\partial V/\partial S$, and $V_{SS}$ denotes $\partial^2 V/\partial S^2$.

Several important assumptions are made in the Black-Scholes model:

- random walk: assumption that $S$ follows a Weiner process
- no arbitrage: it is not possible to make instantaneous riskless profit
- frictionless market: no transaction costs to buying or selling assets/options
- fractional cash: transactions can be made with any amount of cash – any real number is permitted

The Black-Scholes equation is "adjusted" – certain terms may be added to it, some terms may be changed – when we are interested in other options, such as American options.

## 2.3 American Options

An American option, unlike a European option, allows the holder to exercise the option at any time up to and including the time of expiry. Since the holder of an American option has more rights, we expect that if all other conditions are equal, the price of an American option is at least that of a European option. Unlike European options, American options do not have a known closed-form solution, which makes them the first type of options that we examine which requires us to use numerical methods.

Figure 2.1 shows the plot of the value of an American Put option along with the payoff and the corresponding European Put option. The payoff function "pushes up" the value of the American option. Without the American early exercise right, the option value would not be pushed up and would be the exact same as the European option's payoff.

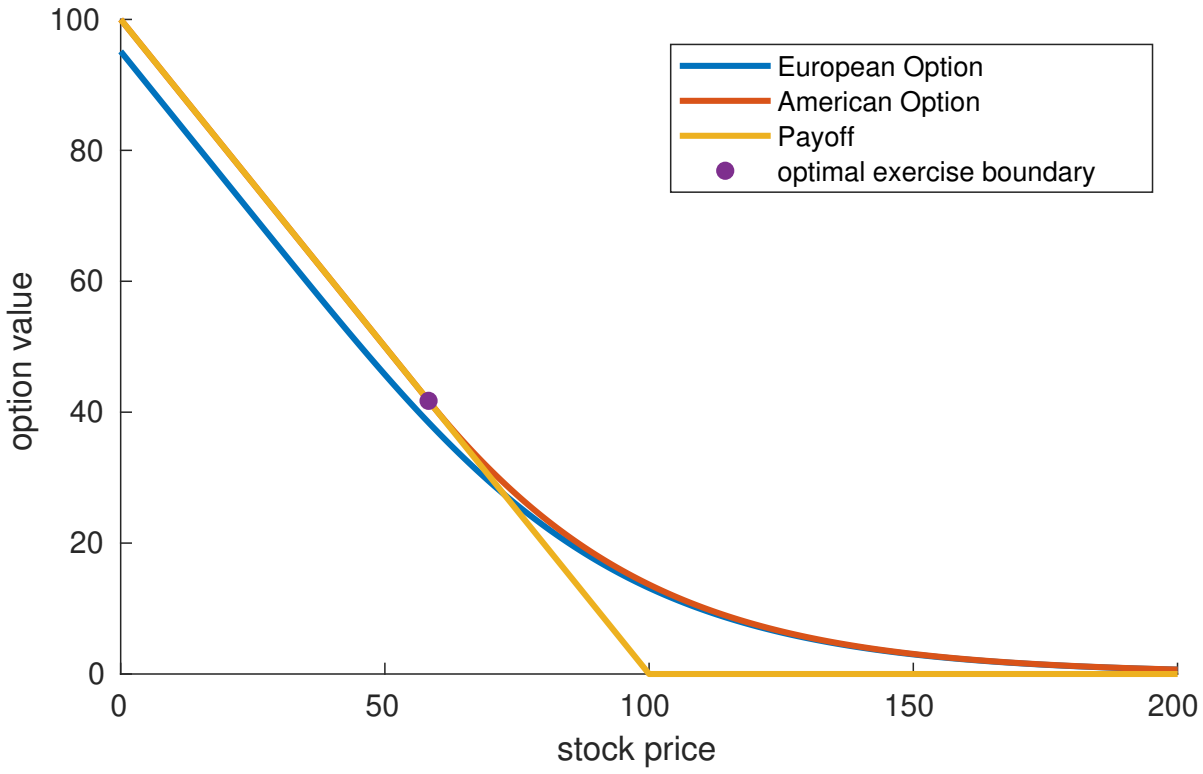


Figure 2.1: Plot of American option value and the optimal exercise boundary

A formulation of the American options pricing problem takes the approach of a linear complementarity problem (LCP). For a full derivation, see Chapter 6 of [36]. For the numerical solution, we transform the LCP into a penalized PDE problem, as introduced in [16].

The LCP for American options is given by

$$V_\tau - \mathcal{L}(r)V \geq 0 \tag{2.2a}$$
$$V - V^* \geq 0 \tag{2.2b}$$
$$(V_\tau - \mathcal{L}(r)V = 0) \vee (V - V^* = 0) \tag{2.2c}$$

with the notation $\vee$ meaning "logical or" of the two statements. All three conditions (2.2a), (2.2b), and (2.2c) must be satisfied.

An alternative way to express the constraints imposed by Equations (2.2) is

$$V_\tau - \mathcal{L}(r)V \geq 0 \text{ and } V - V^* = 0 \tag{2.3a}$$
$$V - V^* \geq 0 \text{ and } V_\tau - \mathcal{L}(r)V = 0 \tag{2.3b}$$

Either condition (2.3a) or (2.3b) must be satisfied. When condition (2.3a) is satisfied, we are in the *exercise region*, where it would be best to exercise the option immediately. This corresponds to the left of the optimal exercise boundary in Figure 2.1. When (2.3b) is satisfied, we are in the continuation region, where it is best to hold on to the option.

The penalized PDE problem of pricing American options is given by

$$V_\tau - \mathcal{L}(r)V = \rho \max(V^* - V, 0), \tag{2.4}$$

or by

$$V_\tau = \frac{\sigma^2 S^2}{2} V_{SS} + rSV_S - rV + \rho \max(V^* - V, 0), \tag{2.5}$$

where $\rho$ is known as the penalty parameter, typically very large.

The idea of the penalty method is that we "penalize" violation of the constraint $V - V^* \geq 0$. As $\rho \to \infty$, the solution satisfies $V \geq V^* - \epsilon$ for some positive $\epsilon \ll 1$.

We solve this with a penalty PDE method, however, it can also be posed as an optimal control problem, which we will introduce in the next section.

## 2.4 Optimal Control

For the type of optimal control problems we are interested in, the general form of the problems is as follows:

$$V_\tau = \sup_{Q \in \hat{Q}} \{a(S,\tau,Q)V_{SS} + b(S,\tau,Q)V_S + c(S,\tau,Q)V + d(S,\tau,Q)\}. \tag{2.6}$$

There is a corresponding formulation which is sometimes used:

$$V_\tau = \inf_{Q \in \hat{Q}} \{a(S,\tau,Q)V_{SS} + b(S,\tau,Q)V_S + c(S,\tau,Q)V + d(S,\tau,Q)\}. \tag{2.7}$$

Note that since the objective function is the same and it is well-known that maximizing and minimizing are interchangeable, these two are equivalent classes of problems. Often, in finance, the $\sup$ formulation corresponds to the short position and the $\inf$ formulation corresponds to the long position.

Equations (2.6) and (2.7) are a type of Hamilton-Jacobi-Bellman (HJB) equations.

Here, $Q$ is called the control and corresponds to parameters we have control over (hence its name). For instance, $Q$ could indicate whether we have exercised an American option early or not. The value of $Q$ is restricted to $\hat{Q}$, which is the set of all permitted states of $Q$.

By choosing different functions of $a, b, c, d$, the optimal control problem corresponds to different types of options. For example, with

$$a = \sigma^2 S^2/2 \tag{2.8a}$$
$$b = rS \tag{2.8b}$$
$$c = -r - \mu/\epsilon \tag{2.8c}$$
$$d = \mu V^*/\epsilon \tag{2.8d}$$

Here, $Q$ is $\mu$ and $\hat{Q} is \{0,1\}$.

and $\mu \in \{0,1\}$, we have the American option pricing problem, with $\epsilon$ corresponding to $1/\rho$ in Equation (2.5). The full HJB equation for the American option pricing problem is

$$V_\tau = \sup_{\mu \in \{0,1\}} \left\{ \frac{\sigma^2 S^2}{2} V_{SS} + rSV_S - rV + \mu \frac{V^* - V}{\epsilon} \right\}. \tag{2.9}$$

For a different set of functions we have the European option, for instance,

$$a = \sigma^2 S^2/2 \tag{2.10a}$$
$$b = rS \tag{2.10b}$$
$$c = -r \tag{2.10c}$$
$$d = 0, \tag{2.10d}$$

which leads to Equation (2.1). Note that $q$ doesn't show up here. This is because the European option can't be controlled; it merely follows the PDE derived by Black and Scholes (2.1).

The optimal control problems (2.6) and (2.7) form the basis for all of our future problems in this chapter.

For the optimal control problems, there are different formulations for the long and the short positions. Computationally, there is no major difference, but when proving monotonicity and convergence of the algorithms, there are difficulties with the long position.

## 2.5 Borrow-Lend

One interesting optimal control problem is that of the pricing of European-style options with unequal borrowing/lending rates [4] with a straddle payoff: the holder has both a put and call option, so the payoff is $V^* = |S - K|$. There are two interest rates: $r_b$ for borrowing and $r_l$ for lending. It is assumed that the borrowing interest rate is higher than the lending interest rate, that is, $r_b > r_l$.

We first present the short position given in [14], which is as follows:

$$V_\tau = \sup_{q \in \{r_l, r_b\}} \left\{ \frac{\sigma^2 S^2}{2} V_{SS} + qSV_S - qV \right\}. \tag{2.11}$$

Forsyth also gives a discontinuous coefficient PDE for the short position:

$$V_\tau = \frac{\sigma 2 S^2}{2} V_{SS} + \rho(V - SV_S)(SV_S - V) \tag{2.12}$$

where

$$\rho(x) = \begin{cases} r_l & \text{if } x \geq 0 \\ r_b & \text{if } x < 0 \end{cases} \tag{2.13}$$

We derive a new formulation of (2.11) based on a nonlinear PDE in the style of the American Penalty problem (2.5).

$$\begin{aligned}\sup_{q\in\{r_l,r_b\}}\left\{\frac{\sigma^2S^2}{2}V_{SS}+qSV_S-qV\right\} &= \frac{\sigma^2S^2}{2}V_{SS}+\max\left\{r_l(SV_S-V),r_b(SV_S-V)\right\}\\ &= \frac{\sigma^2S^2}{2}V_{SS}+r_bSV_S-r_bV+\max\left((r_l-r_b)(SV_S-V),0\right)\\ &= \frac{\sigma^2S^2}{2}V_{SS}+r_bSV_S-r_bV+\max\left((r_b-r_l)(V-SV_S),0\right)\\ &= \frac{\sigma^2S^2}{2}V_{SS}+r_bSV_S-r_bV+(r_b-r_l)\max\left(V-SV_S,0\right).\end{aligned}$$

Thus, the short position is

$$V_\tau=\frac{\sigma^2S^2}{2}V_{SS}+r_bSV_S-r_bV+(r_b-r_l)\max\left(V-SV_S,0\right). \tag{2.14}$$

For the long position of both Equation (2.11) and similar problems, the $\sup$ is replaced by an $\inf$:

$$V_\tau=\inf_{q\in\{r_l,r_b\}}\left\{\frac{\sigma^2S^2}{2}V_{SS}+qSV_S-qV\right\} \tag{2.15}$$

Forsyth also presents a similar discontinuous coefficient PDE for the long position

$$V_\tau=\frac{\sigma 2S^2}{2}V_{SS}+\rho(SV_S-V)(V-SV_S) \tag{2.16}$$

with the same definition of $\rho$.

For the long position, the PDE is derived in a similar way as the short position and is given by

$$V_\tau=\frac{\sigma^2S^2}{2}V_{SS}+r_bSV_S-r_bV+(r_l-r_b)\max\left(SV_S-V,0\right). \tag{2.17}$$

In Chapter 3, we solve Equations (2.14) and (2.17) by a penalty method.

## 2.6 Borrow-Lend with Stock Borrowing Fees

We extend the borrowing/lending model to include stock borrowing fees, which is described in detail in [13]. In essence, this changes the model such that the holder of a short position has

a deduction of $r_f$, known as the stock borrowing fee, on the rate of the proceeds of the short sale. This changes Equation (2.11) to the following:

$$V_\tau = \sup_Q \left\{ \frac{\sigma^2 S^2}{2} V_{SS} + q_3(q_1 S V_S - q_1 V) + (1 - q_3)((r_l - r_f) S V_S - q_2 V) \right\} \tag{2.18}$$

with $Q = (q_1, q_2, q_3)$, $q_1 \in \{r_l, r_b\}$, $q_2 \in \{r_l, r_b\}$, $q_3 \in \{0, 1\}$ for the short position.

For the long position, we change the $\sup$ to an $\inf$:

$$V_\tau = \inf_Q \left\{ \frac{\sigma^2 S^2}{2} V_{SS} + q_3(q_1 S V_S - q_1 V) + (1 - q_3)((r_l - r_f) S V_S - q_2 V) \right\} \tag{2.19}$$

with $Q = (q_1, q_2, q_3)$, $q_1 \in \{r_l, r_b\}$, $q_2 \in \{r_l, r_b\}$, $q_3 \in \{0, 1\}$

Forsyth gives another pair of discontinuous coefficient PDEs for the short and the long positions:

$$V_\tau = \frac{\sigma^2 S^2}{2} V_{SS} + H(V_S)[\rho(V - S V_S)(S V_S - V)] + H(-V_S)[(r_l - r_f) S V_S - \rho(V) V] \tag{2.20}$$

$$V_\tau = \frac{\sigma^2 S^2}{2} V_{SS} + H(-V_S)[\rho(S V_S - V)(V - S V_S)] + H(V_S)[(r_l - r_f) S V_S - \rho(-V) V] \tag{2.21}$$

where $\rho(x)$ is defined the same as in the Borrow-Lend problem, and

$$H(y) = \begin{cases} 1 & \text{if } y \geq 0 \\ 0 & \text{if } y < 0 \end{cases} \tag{2.22}$$

Like for the problem defined by Equation (2.11), we derived a new formulation based on a nonlinear PDE: derived a penalty method to solve a nonlinear PDE for the problem defined by

Equation (2.18)

$$\begin{aligned}
&\sup_{Q}\left\{\frac{\sigma^2S^2}{2}V_{SS}+q_3q_1(SV_S-V)+(1-q_3)\left[(r_l-r_f)SV_S-q_2V\right]\right\}\\
&=\frac{\sigma^2S^2}{2}V_{SS}+\sup_{Q}\left\{q_1(SV_S-V),(r_l-r_f)SV_S-q_2V\right\}\\
&=\frac{\sigma^2S^2}{2}V_{SS}+\max\left\{\max_{q_1}\{q_1(SV_S-V)\},\max_{q_2}\{(r_l-r_f)SV_S-q_2V\}\right\}\\
&=\frac{\sigma^2S^2}{2}V_{SS}+\max\left\{\max_{q_1}\{q_1(SV_S-V)\},(r_l-r_f)SV_S-r_lV\right\}\\
&=\frac{\sigma^2S^2}{2}V_{SS}+\max\left\{r_l(SV_S-V),r_b(SV_S-V),r_l(SV_S-V)-r_fSV_S\right\}\\
&=\frac{\sigma^2S^2}{2}V_{SS}+r_l(SV_S-V)+\max\{(r_b-r_l)(SV_S-V),-r_fSV_S,0\}
\end{aligned}$$

Note that the expression

$$\max_{q_2}\{(r_l-r_f)SV_S-q_2V\} \tag{2.23}$$

is equivalent to $(r_l-r_f)SV_S-r_lV$, since $V$ is always nonnegative.

Thus, the short position PDE is equal to

$$V_\tau=\frac{\sigma^2S^2}{2}V_{SS}+r_l(SV_S-V)+\max\{(r_b-r_l)(SV_S-V),-r_fSV_S,0\} \tag{2.24}$$

The long position has a similar derivation, resulting in

$$V_\tau=\frac{\sigma^2S^2}{2}V_{SS}+r_b(SV_S-V)+\min\{(r_l-r_b)(SV_S-V),-(r_b-r_l+r_f)SV_S,0\} \tag{2.25}$$

In Chapter 3, we develop penalty methods for Equations (2.24) and (2.25).

## 2.7 Adding American Early Exercise

When we add an American Early Exercise feature to (2.18), we get

$$V_\tau = \sup_{\mu,Q}\left\{\frac{\sigma^2S^2}{2}V_{SS} + q_3q_1(SV_S - V) + (1-q_3)((r_l - r_f)SV_S - q_2V) + \mu\frac{V-V^*}{\epsilon}\right\} \tag{2.26}$$

with $Q = (q_1, q_2, q_3)$, $q_1 \in \{r_l, r_b\}$, $q_2 \in \{r_l, r_b\}$, $q_3 \in \{0, 1\}$ and $\mu \in \{0, 1\}$.

However, the long position is claimed to be more interesting in [14]:

$$V_\tau = \sup_{\mu}\inf_{Q}\left\{\frac{\sigma^2S^2}{2}V_{SS} + q_3q_1(SV_S - V) + (1-q_3)((r_l - r_f)SV_S - q_2V) + \mu\frac{V-V^*}{\epsilon}\right\}, \tag{2.27}$$

with $Q = (q_1, q_2, q_3)$, $q_1 \in \{r_l, r_b\}$, $q_2 \in \{r_l, r_b\}$, $q_3 \in \{0, 1\}$ and $\mu \in \{0, 1\}$

Unlike in the two previous cases, this problems seems to be introduced by Forsyth, who only gives the control problems and does not give any nonlinear PDEs.

Due to the $\sup\inf$, this type of equation is known as a Hamilton-Jacobi-Bellman-Issacs (HJBI) equation, and corresponds to a stochastic game: One agent tries to minimize the objective function over the possible choices of $Q$, and the other agent tries to maximize the objective function over $\mu$. It is considered a more difficult problem; however, we have not had computational difficulties by calculating the $\sup\inf$ as it is.

For both the short and long positions of this problem, we can add an American penalty constraint to our penalty formulation used to model the previous problem.

The equations we get are very similar to the penalty problem arising from Equation (2.18) combined with the American early exercise feature from Equation (2.5).

Short position:

$$V_\tau = \frac{\sigma^2S^2}{2}V_{SS} + r_l(SV_S - V) + \max\{(r_b - r_l)(SV_S - V), -r_fSV_S, 0\} + \rho\max(V - V^*, 0) \tag{2.28}$$

Long position:

$$V_\tau = \frac{\sigma^2S^2}{2}V_{SS} + r_b(SV_S - V) + \min\{(r_l - r_b)(SV_S - V), -(r_b - r_l + r_f)SV_S, 0\} + \rho\max(V - V^*, 0) \tag{2.29}$$

In Chapter 3, we solve Equations (2.28) and (2.29) by a double-penalty method, similar to [7] for the pricing of valuation adjustments.

## 2.8 Uncertain Volatility

The Uncertain Volatility model [1] is given as a control problem in [14] and examined as a nonlinear PDE in [29] and more extensively as a control problem in [17].

We solve the equations

$$V_\tau = \sup_{q\in\{\sigma_{\min},\sigma_{\max}\}} \left\{ \frac{q^2 S^2}{2} V_{SS} + rSV_S - rV \right\} \tag{2.30}$$

and

$$V_\tau = \inf_{q\in\{\sigma_{\min},\sigma_{\max}\}} \left\{ \frac{q^2 S^2}{2} V_{SS} + rSV_S - rV \right\} \tag{2.31}$$

which we will refer to as the "best case" and "worst case", respectively.

The payoff function is called a butterfly spread: it is defined by

$$V^*(S) = \begin{cases} 0 & \text{if } S \le K-a \text{ or } S > K+a \\ S-(K-a) & \text{if } K-a < S \le K \\ (K+a)-S & \text{if } K < S \le K+a \end{cases} \tag{2.32}$$

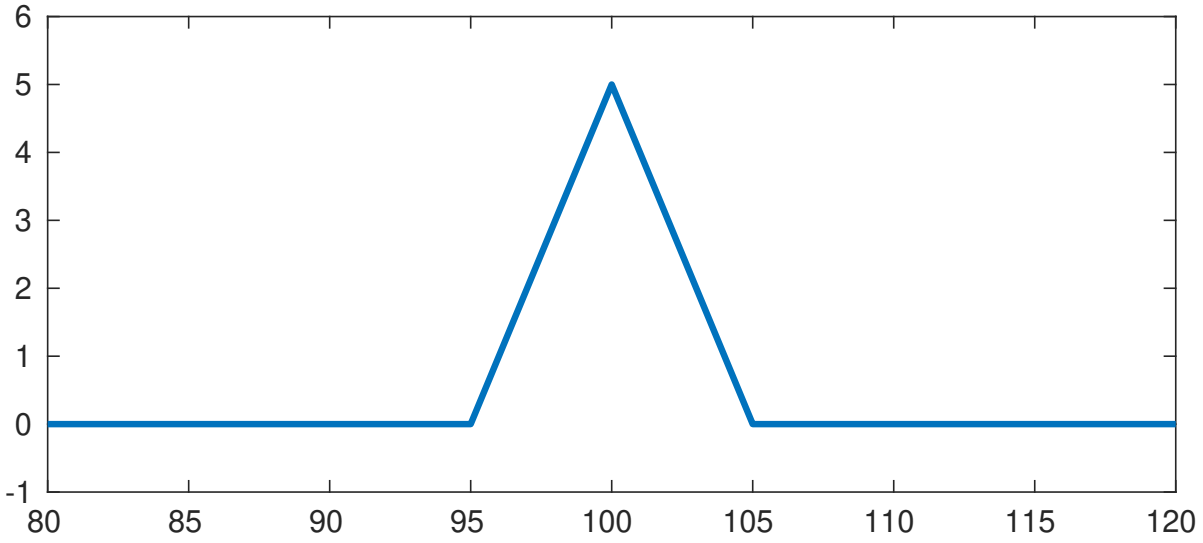


Figure 2.2: Butterfly Spread payoff function, with $K = 100$ and $a = 5$

It can be generated by a portfolio of vanilla call/put options, with two short call/put options at the strike price $K$, and two long call/put options at the adjusted strike prices $K - a$ and $K + a$.

The nonlinear PDE that describes the best case is

$$V_\tau = \frac{\sigma_{\min}^2 S^2}{2} V_{SS} + rSV_S - rV + \max\left\{ \frac{(\sigma_{\max}^2 - \sigma_{\min}^2) S^2}{2} V_{SS}, 0 \right\} \tag{2.33}$$

and the nonlinear PDE that describes the worst case is

$$V_\tau = \frac{\sigma_{\max}^2 S^2}{2} V_{SS} + rSV_S - rV + \max\left\{ \frac{(\sigma_{\min}^2 - \sigma_{\max}^2) S^2}{2} V_{SS}, 0 \right\} \tag{2.34}$$

In Chapter 3, using penalty methods, we solve these two nonlinear PDEs.

## 2.9 Transaction cost Models

Transaction cost models have been introduced in [24], and extensively debated and refined in [22, 38, 25, 35]. It is beyond the scope of the paper to discuss the merits and drawbacks of these models; we use the model given in [37], which is based on [35]. It is given by

$$V_\tau = \frac{\sigma^2 S^2}{2} V_{SS} + rSV_S - rV - \kappa S^2 |V_{SS}| \tag{2.35}$$

where $\kappa < \sigma^2/2$ (to ensure a positive coefficient of the diffusion term) is the transaction cost parameter.

For a transaction cost model with American options, Yousuf [37], following the logic in [16], gives

$$V_\tau = \frac{\sigma^2 S^2}{2} V_{SS} + rSV_S - rV - \kappa S^2 |V_{SS}| + \rho \max(V^* - V, 0) \tag{2.36}$$

where $\rho$ is the same as the one used for American options in [16].

For the cases where the payoff is convex, $V_{SS} \geq 0$ and Equation (2.35) becomes

$$V_\tau = \left( \frac{\sigma^2}{2} - \kappa \right) S^2 V_{SS} + rSV_S - rV, \tag{2.37}$$

the Black-Scholes equation; and Equation (2.36) becomes

$$V_\tau = \left( \frac{\sigma^2}{2} - \kappa \right) S^2 V_{SS} + rSV_S - rV + \rho \max(V^* - V, 0), \tag{2.38}$$

the penalty PDE for American options, as examined in [16]. In both cases, the volatility is adjusted; the new volatility $\sigma'$ is given by

$$\sigma' = \sqrt{\sigma^2 - 2\kappa} \tag{2.39}$$

Which gives rise to the requirement that $\kappa < \sigma^2/2$.

Since the PDE becomes a linear PDE under a convex payoff, we also consider the Butterfly spread payoff as an initial condition for this model.

## 2.10 Greeks

The Greeks are partial derivatives that measure sensitivity of price to changes in the variables. They have important practical uses in risk management and hedging, so we will have a brief mention of some of the most common ones: Delta ($V_S$) and Gamma ($V_{SS}$). We compute the numerical values of these with second-order finite difference formulas.

# Chapter 3

# Numerical Methods

In this chapter, we discuss the numerical methods used to solve the problems previously described.

## 3.1 Time and Space Discretization

In this section we describe the time and space discretizations for the Black-Scholes PDE. This will firstly allow us to numerically price European options; and secondly it serves as a foundation for problems in the later sections.

### 3.1.1 Temporal Discretization

To obtain the solution at the end time $V(\tau = T, S)$ starting from $\tau_0 = 0$, we integrate through time at various timesteps $0 < \tau_1 < \tau_2 < \ldots < T$. Although the description is for uniform timesteps, it applies equally well to nonuniform timesteps. We denote $V(\tau_j, \cdot)$ as $V^j$, and we use $\Delta\tau$ to denote the step size taken. In our descriptions we assume a constant step size, however, the description doesn't change for variable step sizes.

The time-stepping scheme that we use in this thesis is one that is proposed by Crank and Nicolson in 1947 [12]. We include a brief explanation below (superscripts denote timesteps):

Suppose that the PDE of interest is

$$V_\tau = F(\tau, S, V, V_S, V_{SS}) \tag{3.1}$$

As we take a step of size $\Delta\tau$ forwards from $V^{j-1}$ to $V^j$, the solution at time $j-1$ to time $j$, the Crank-Nicolson (CN) timestepping method solves the following system of equations for the solution at the next timestep

$$\frac{V^j - V^{j-1}}{\Delta\tau} = \frac{1}{2}\left(F^j(\tau, S, V, V_S, V_{SS}) + F^{j-1}(\tau, S, V, V_S, V_{SS})\right), \tag{3.2}$$

where $F^j(\tau, S, V, V_S, V_{SS})$ denotes $F(\tau^j, S, V^j, V_S^j, V_{SS}^j)$.

Another timestepping method is

$$\frac{V^j - V^{j-1}}{\Delta\tau} = F^j(\tau, S, V, V_S, V_{SS}) \tag{3.3}$$

which is the implicit Euler or Backwards Euler (BE) method.

It is useful to examine BE and CN together, as a weighted sum parametrized by $\theta$. A general form is

$$\frac{V^j - V^{j-1}}{\Delta\tau} = \theta F^j(\tau, S, V, V_S, V_{SS}) + (1-\theta)F^{j-1}(\tau, S, V, V_S, V_{SS}), \tag{3.4}$$

known as a family of $\theta$-methods.

Note that as long as $\theta > 0$, this is an *implicit* method: the solution must be obtained by solving a possibly nonlinear system of equations instead of by direct calculation as in explicit methods, which have restrictions on their step size for stability requirements. Implicit methods frequently have less stringent requirements than explicit methods and often are unconditionally stable.

This allows us to take larger timesteps at the cost of solving a system of linear equations. Fortunately, since the computation of derivative values only depend on adjacent nodes, the matrix $A$ which computes the spatial discretization of $\mathcal{L}$ is tridiagonal and we can solve the systems of equations in linear time; an example is Algorithm 2.8 in Chapter 2 of [20].

In a subsequent section, we will also show that under certain conditions the matrix $A$ is diagonally dominant; however, we note that such a result has already been proven in [9] and we extend it to other matrices arising from the nonlinear problems.

Compared with CN, which is second order accurate, BE is only first order accurate. However, BE has damping properties which turns out to be helpful because of the nondifferentiable nature of the initial conditions (there is a cusp at the strike price $K$). After applying a constant number of BE timesteps and the solution curve has been sufficiently smoothed out, we can then apply CN timestepping for the remaining timesteps. This technique of a few iterations of

BE followed by CN is known as Rannacher smoothing [30], and it does not affect the overall second-order accuracy of CN. The original Rannacher smoothing technique divides the initial timestep into two equal substeps, and instead of taking a CN step over the entire timestep, we take one BE step over each substep. However, it has been shown in [10, 18] that the best Rannacher smoothing for convergence is when we divide the first two timesteps into four substeps, and take BE steps over each substep.

The effects of Rannacher smoothing may not be noticeable when we look at the value itself, however, when we look at the "Greeks" we notice that the initial non-smoothness has been exacerbated when we do not use Rannacher smoothing, and smoothed out when we implement Rannacher smoothing (see Figure 3.1).

### 3.1.2 Spatial Discretization

Recall that the domain for $S$, the price of the underlying asset, is theoretically any nonnegative amount. However, to make computation practical, we typically truncate the semi-infinite domain $[0, \infty)$ to a finite domain $[0, S_{\max}]$. In our numerical experiments, we truncate to $[0, 10K]$, where $K$ is the strike price.

Recall also the Black-Scholes equation:

$$V_\tau = \frac{\sigma^2 S^2}{2} V_{SS} + rSV_S - rV \equiv \mathcal{L}(r)V \tag{3.5}$$

We will directly describe the discretization on an arbitrary grid, where the continuous domain $[0, S_{\max}]$ is represented with a finite number of points denoted as $S_0 = 0 < S_1 < S_2 < \cdots < S_N = S_{\max}$. Define $v$ to denote the vector of approximate values at the gridpoints $\{S_i\}_{i=0}^N$ that is calculated at each timestep. To calculate the partial derivatives $V_S$ and $V_{SS}$, we use the $(N+1) \times (N+1)$ tridiagonal matrices $T_1$ and $T_2$ defined based on formulas (3.6). These matrices are defined such that $T_1 v$ and $T_2 v$ approximate the values of $V_S$ and $V_{SS}$ respectively at the gridpoints.

We consider uniform grids only to demonstrate the superiority of nonuniform grids in lessening of discretization error.

One reason for using a nonuniform grid is the fact that not all regions of the spatial domain have equal significance. The region near the strike price is more important because the underlying is more likely to be near that price. A second reason is that the graph of the value away from the

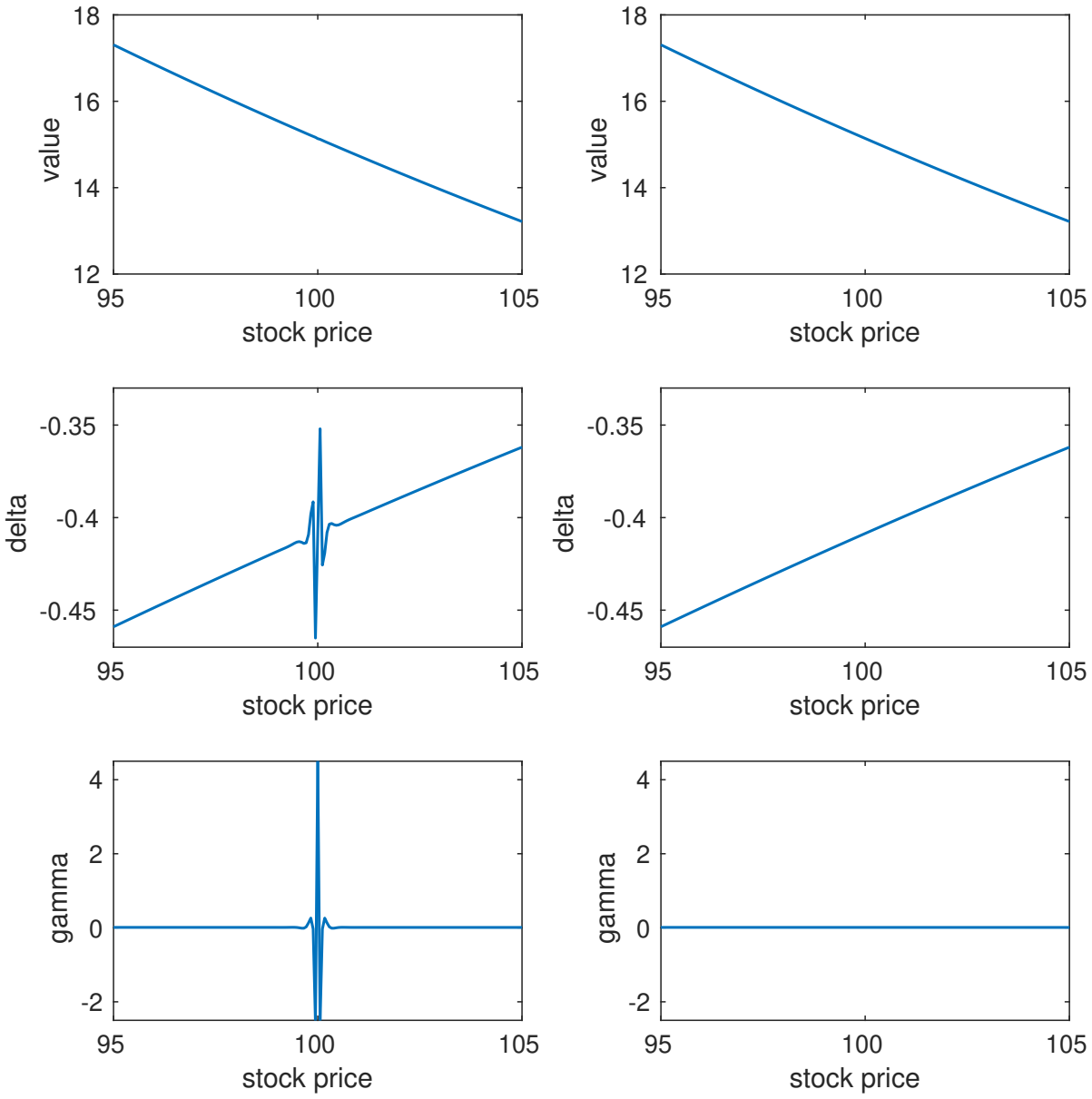


Figure 3.1: European Put option, $K = 100$, $\sigma = 0.8$, $r = 0.05$, $T = 0.25$. Grid: $N = 1000$, $\Delta\tau = 1/1600$, Clarke-Parrott with $a = 0.38$. *Left*: No Rannacher smoothing. *Right*: Rannacher smoothing over 1 timesteps. *Top*: Option value ($V$). *Middle*: Delta ($V_S$). *Bottom*: Gamma ($V_{SS}$)

strike price is almost linear; therefore fewer points are required for the same level of accuracy. An example of the benefits of using a nonuniform grid can be seen in Figure 3.2: the maximum error located near the strike price $K$ is reduced by almost an order of magnitude with otherwise equal parameters.

Although any grid that is not a uniform grid is a nonuniform grid, in practice we restrict the

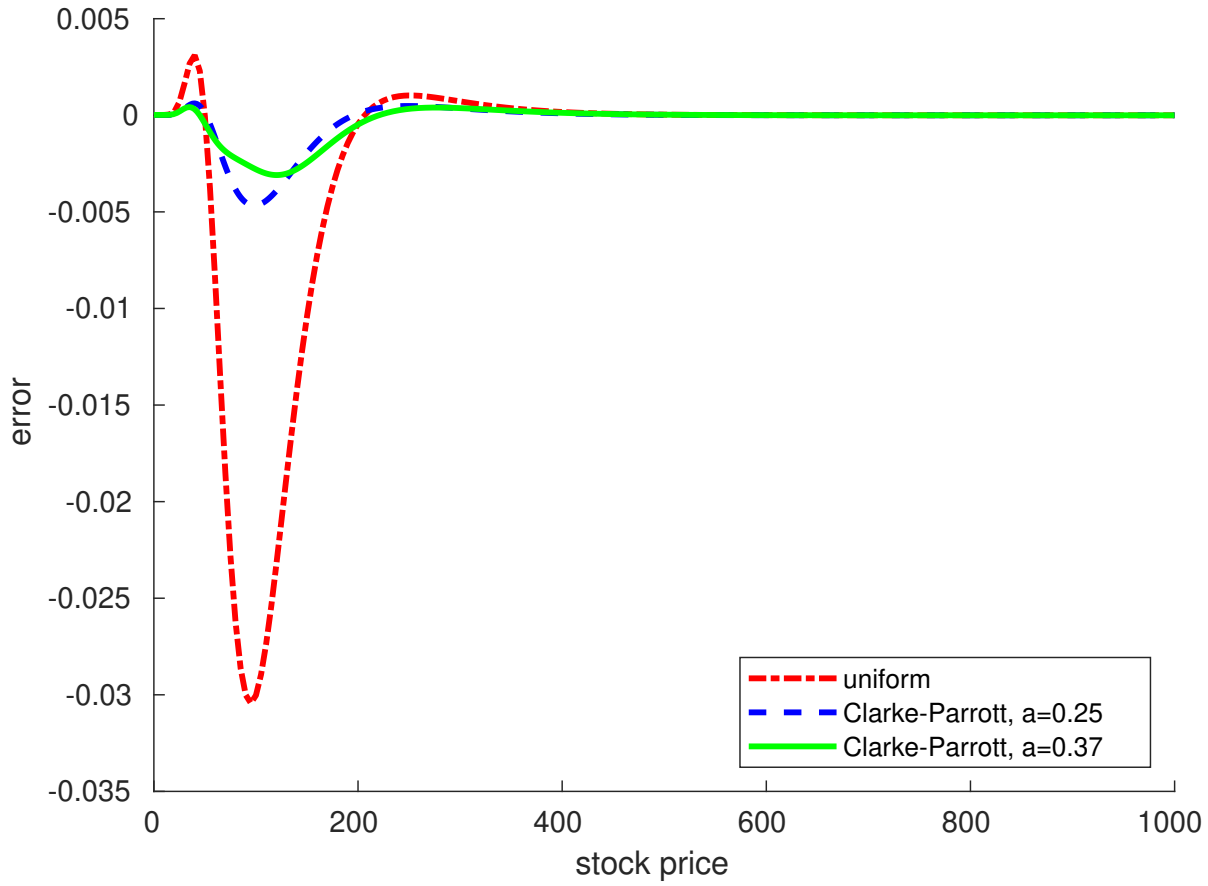


Figure 3.2: Error of European option at the strike price is reduced significantly with the use of a nonuniform grid

nonuniform grids of interest to those that can be generated by applying a mapping function to a uniform grid. By imposing certain requirements on the mapping function, we can obtain desired convergence results.

The mapping functions that we use are defined as $g\colon [0,1] \to [0,1]$ and must satisfy the following requirements (we scale the resulting grid as needed):

1. $g$ has fixed endpoints: $g(0) = 0$, $g(1) = 1$

2. $g$ is monotonically increasing, continuous, and has two continuous derivatives

The function $g$ is plotted in Figure 3.3 with different values of $a$.

Let the grid be defined by $S_i$ $(0 \leq i \leq n)$ satisfying $S_0 = 0 < S_1 < S_2 < \ldots < S_{n-1} < S_n = S_{\max}$. Define $h_i = S_i - S_{i-1}$. Let $V_i$ be the value of $V(\cdot, S_i)$ at an arbitrary point in time. We show the formulas for approximating $V_S$ and $V_{SS}$ on arbitrary gridpoints, where $[V_S]_i$ denotes $V_S$ at $S_i$.

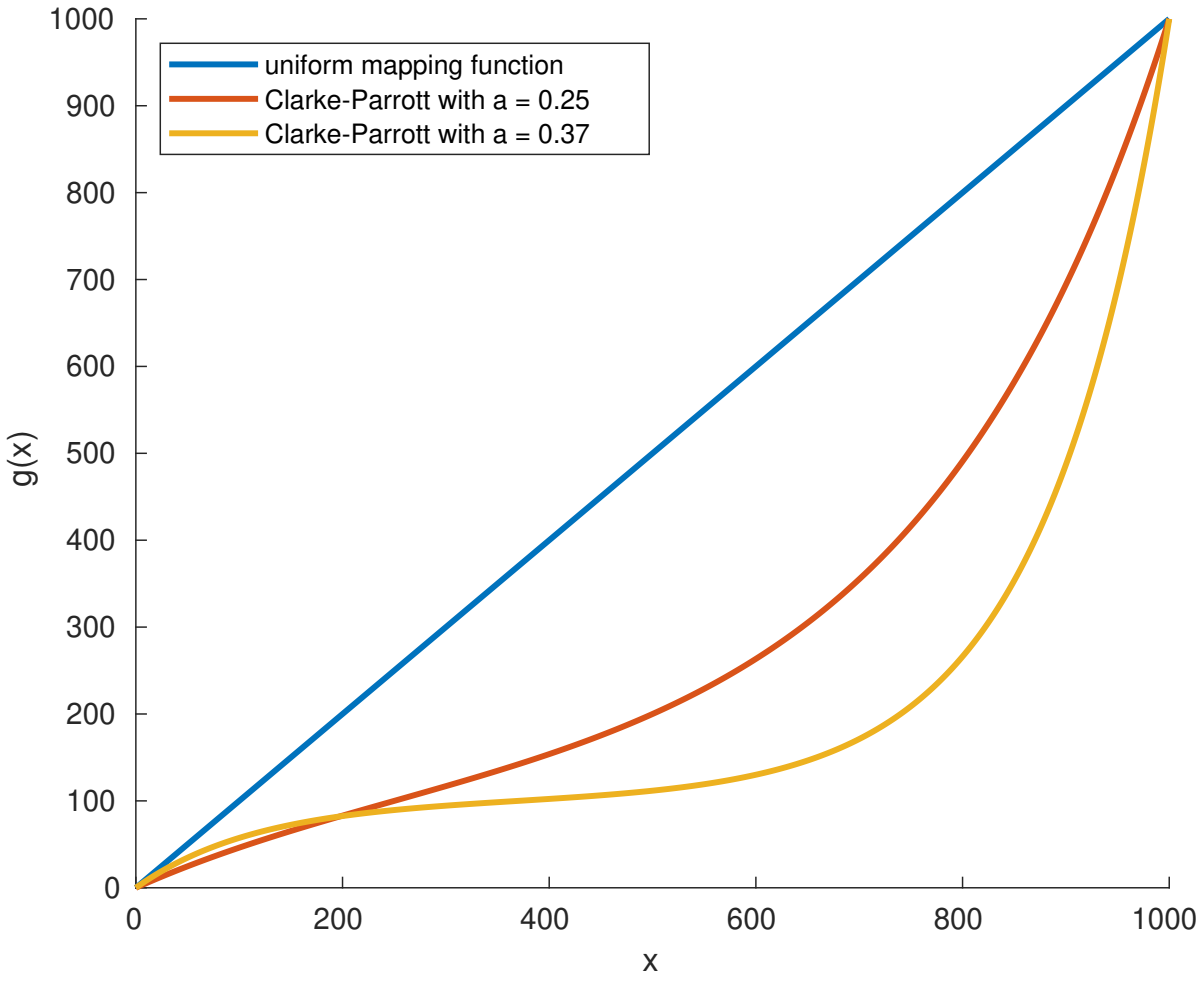


Figure 3.3: Plot of mapping functions with various values of $a$

More specifically, the nonuniform centered differences are computed with the following approximations:

$$[V_S]_i \approx \frac{-h_{i+1}}{h_i(h_i+h_{i+1})}V_{i-1} + \frac{h_{i+1}-h_i}{h_i h_{i+1}}V_i + \frac{h_i}{h_{i+1}(h_{i+1}+h_i)}V_{i+1} \tag{3.6a}$$

$$[V_{SS}]_i \approx \frac{2}{h_i(h_i+h_{i+1})}V_{i-1} + \frac{-2}{h_i h_{i+1}}V_i + \frac{2}{h_{i+1}(h_i+h_{i+1})}V_{i+1}. \tag{3.6b}$$

For the first derivative, we also use backward differences, computed by the following expression

$$[V_S]_i \approx \frac{V_i - V_{i-1}}{h_i} \tag{3.7}$$

However, this is only used on the right endpoint at $S = S_{\max}$ when we implement linear boundary conditions (see Section 3.1.2). Therefore, $i = N$ in Equation (3.7) for all usage in

this thesis.

Using Equations (3.6) and (3.7), we can define the tridiagonal matrices $T_1$ and $T_2$:

$$T_1 = \begin{bmatrix} 0 & 0 & 0 & 0 & \cdots & 0 \\ \frac{-h_2}{h_1(h_1+h_2)} & \frac{h_2-h_1}{h_1h_2} & \frac{h_1}{h_2(h_2+h_1)} & 0 & \cdots & 0 \\ 0 & \frac{-h_3}{h_2(h_2+h_3)} & \frac{h_3-h_2}{h_2h_3} & \frac{h_2}{h_3(h_3+h_2)} & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & \cdots & 0 & \frac{-h_N}{h_{N-1}(h_{N-1}+h_N)} & \frac{h_N-h_{N-1}}{h_{N-1}h_N} & \frac{h_{N-1}}{h_N(h_N+h_{N-1})} \\ 0 & 0 & 0 & 0 & \cdots & 0 \end{bmatrix} \tag{3.8}$$

$$T_2 = \begin{bmatrix} 0 & 0 & 0 & 0 & \cdots & 0 \\ \frac{2}{h_1(h_1+h_2)} & \frac{-2}{h_1h_2} & \frac{2}{h_2(h_1+h_2)} & 0 & \cdots & 0 \\ 0 & \frac{2}{h_2(h_2+h_3)} & \frac{-2}{h_2h_3} & \frac{2}{h_3(h_2+h_3)} & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & \cdots & 0 & \frac{2}{h_{N-1}(h_{N-1}+h_N)} & \frac{-2}{h_{N-1}h_N} & \frac{2}{h_N(h_{N-1}+h_N)} \\ 0 & 0 & 0 & 0 & \cdots & 0 \end{bmatrix} \tag{3.9}$$

To compute the discretization of the Black-Scholes operator $\mathcal{L}(r)V$ in Equation (3.5), we define the diagonal matrices $W_0$, $W_1$ and $W_2$, where the diagonal entries are defined by the following:

$$[W_0]_{ii} = r \tag{3.10}$$

$$[W_1]_{ii} = rS_{i-1} \tag{3.11}$$

$$[W_2]_{ii} = \frac{\sigma^2 S_{i-1}^2}{2} \tag{3.12}$$

Note that since matrices are always indexed from 1 to $N+1$, and the nodes are indexed from 0 to $N$, the indices will have a difference of one in Equations (3.11) and (3.12).

Then, define

$$A = W_0 + W_1 T_1 + W_2 T_2 + B, \tag{3.13}$$

where $W_0$, $W_1$, $W_2$, $T_1$, $T_2$ are defined above, and $B$ is the matrix which defines the boundary conditions. In the case of Dirichlet boundary conditions on both endpoints, $B_{1,1} = B_{N+1,N+1} = 1$. In the case of Dirichlet boundary conditions on the left endpoint and linear boundary conditions on the right endpoint,

$$B_{1,1} = 1 \tag{3.14a}$$

$$B_{N+1,N} = rS_{N+1}\left(\frac{1}{h_N}\right) \tag{3.14b}$$

$$B_{N+1,N+1} = rS_{N+1}\left(\frac{-1}{h_N}\right) - rS_{N+1} \tag{3.14c}$$

The other entries of the matrix $B$ are zero.

Applying $\theta$-timestepping to Equation (3.5), we have

$$\begin{aligned}
\frac{V^j - V^{j-1}}{\Delta\tau} &= \theta A V^j + (1-\theta) A V^j \\
V^j - V^{j-1} &= \theta \Delta\tau A V^j + (1-\theta)\Delta\tau A V^{j-1} \\
V^j - \theta\Delta\tau A V^j &= V^{j-1} + (1-\theta)\Delta\tau A V^j \\
(I - \theta\Delta\tau A)V^j &= (I + (1-\theta)\Delta\tau A)V^{j-1}
\end{aligned}$$

We solve the linear system

$$(I - \theta\Delta\tau A)V^j = (I + (1-\theta)\Delta\tau A)V^{j-1} \tag{3.15}$$

for the unknown $V^j$ at each timestep to advance from time $j-1$ to time $j$.

### Specific nonuniform grid

The nonuniform grid that we use is introduced in [11], defined by the inverse of the function

$$g(x_i) = K\left(1 + \frac{\sinh(b(x_i - a))}{\sinh(ba)}\right) \tag{3.16}$$

on the interval $[0, 1]$, where $x_i$ are the gridpoints, and $a$, $b$ are fixed parameters. The parameter $a$ is a measure of concentration of gridpoints near the strike price; higher values of $a$ have increased number of points near the strike price. The other parameter $b$ is used to ensure that $g(1) = 1$. After we solve for the grid points, we scale this grid to the proper spatial domain (i.e. $S_i = S_{\max} x_i$).

In practice, we would solve Equation (3.17) numerically for $b$ after choosing an appropriate $a$. In this thesis we set $a = 0.37$ for most of our problems where we use this family of nonuniform grids.

$$S_{\max} = K \left( 1 + \frac{\sinh(b(1-a))}{\sinh(ba)} \right) \tag{3.17}$$

This nonuniform grid ensures that there exists a point on the only non-smooth point in the initial conditions, which is the strike price for call, put, and straddle payoffs. This ensures that the we converge to the correct solution [28].

However, for when we are solving the Uncertain Volatility problem, we have the Butterfly Spread payoff function and there is no way to ensure that we can place nodes located at the three strike prices $K_1$, $K_2$, and $K_3$. Therefore, we use a uniform grid with an appropriate number of subintervals to ensure that each of the three non-smooth points coincides with a node.

Differentiating the function $g(S)$ twice would show that it has a bounded second derivative. Therefore, as we will show in the next section, the discretization of the diffusion ($\frac{1}{2}\sigma^2 S^2 V_{SS}$) term in the Black-Scholes PDE converges in second order.

### Boundary Conditions

As a result of truncating the semi-infinite spatial domain, we have to consider the two endpoints of the interval: $S = 0$ and $S = S_{\max}$.

For the left side of the boundary corresponding to $S = 0$, we always use Dirichlet boundary conditions

$$V(\tau, S = 0) = \text{given constant} \tag{3.18}$$

They can be derived by setting $S = 0$ in the PDE of interest and solving for $V$ over time. Since the $V_S$ and $V_{SS}$ terms all have an $S$ coefficient in all of our PDEs, they are zeroed out and we

only have

$$V_\tau = (\text{constants})V \tag{3.19}$$

which can be solved for exactly.

For the right side of the boundary, we use Dirichlet boundary conditions for some simple problems to check agreement with linear boundary conditions

$$V_{SS}(\tau, S = S_{\max}) = 0 \tag{3.20}$$

in our implementations.

Linear boundary conditions are implemented by setting the diffusion term equal to zero on the boundary and discretizing the first derivative $V_S$ according to the backward difference formula (3.7).

For more complicated problems where we don't necessarily want to derive Dirichlet boundary conditions we use linear boundary conditions. In addition to verifying the correctness of linear boundary conditions, the use of Dirichlet boundary conditions allows us to analyze properties such as diagonal dominance of the matrix arising from discretization. This is not possible with linear boundary conditions, because the lack of a diffusion term and the negative coefficient in the $V$ term make diagonal dominance impossible.

### 3.1.3 Order of Residual Terms

By Taylor Series expansion, we can show that for our approximations to the first derivative,

$$\begin{aligned}\text{residual} &= \frac{h_i}{h_{i+1}(h_{i+1}+h_i)}\frac{h_{i+1}^3}{6}V_{SSS}(S_i) - \frac{-h_{i+1}}{h_i(h_i+h_{i+1})}\frac{h_i^3}{6}V_{SSS}(S_i) + \mathcal{O}(\max(h_i, h_{i+1})^2) \\ &= \mathcal{O}(h^2)\end{aligned}$$

The stencil for the second derivative is also second order, if we make assumptions on the

spacing of the points. The residual is

$$\begin{aligned}\text{residual} &= V_{SSS}(S_i)\left[\frac{2}{h_{i+1}(h_i+h_{i+1})}\frac{h_{i+1}^3}{6} - \frac{2}{h_i(h_i+h_{i+1})}\frac{h_i^3}{6}\right] + \mathcal{O}(\max(h_i,h_{i+1})^2)\\ &= V_{SSS}(S_i)\left[\frac{h_{i+1}^2}{3(h_i+h_{i+1})} - \frac{h_i^2}{3(h_i+h_{i+1})}\right] + \mathcal{O}(\max(h_i,h_{i+1})^2)\end{aligned}$$

Therefore,

$$\text{residual} = \frac{1}{3}V_{SSS}(S_i)(h_{i+1}-h_i) + \mathcal{O}(\max(h_i,h_{i+1})^2) \tag{3.21}$$

The only term that may cause a problem for second order convergence of Equation (3.21) is $h_{i+1} - h_i$. To determine under what assumptions it is second order, let $g\colon [0,1] \to [0,1]$ be the mapping function; and let the original points be $w-h$, $w$, $w+h$. The new points are

$$g(w-h) = S_i - h_i \tag{3.22a}$$

$$g(w) = S_i \tag{3.22b}$$

$$g(w+h) = S_i + h_{i+1} \tag{3.22c}$$

The difference can be calculated as follows:

$$\begin{aligned}h_{i+1} - h_i &= (S_i + h_{i+1} - S_i) + (S_i - h_i - S_i)\\ &= (g(w+h) - g(w)) + (g(w-h) - g(w))\\ &= \left[hg'(w) + \frac{h^2}{2}g''(w)\right] + \left[-hg'(w) + \frac{h^2}{2}g''(w)\right] + \mathcal{O}(h^3)\\ &= h^2 g''(w) + \mathcal{O}(h^3)\end{aligned}$$

Therefore, the residual is $\mathcal{O}(h^2)$ as long as $g$ has bounded 2nd derivative.

## 3.2 Policy Iteration

The policy iteration algorithm is applied to all HJB problems, making the analysis for the problems more or less equivalent.

Recall the HJB equation

$$V_\tau = \sup_Q \{a(S,\tau,Q)V_{SS} + b(S,\tau,Q)V_S + c(S,\tau,Q)V + d(S,\tau,Q)\}$$

We modify the definition of $A$, so instead of Equation (3.13), it is now a function of the control $Q$ ($A = A(Q)$) and is defined by

$$A(Q) = W_0(Q) + W_1(Q)T_1 + W_2(Q)T_2 + B, \tag{3.23}$$

Additionally, define $D(Q)$ by the vector that has the values $D_i = d(S_i, Q_i)$.

We want to solve the system

$$(I - \theta\Delta\tau A(Q^j))v^j - \theta\Delta\tau D(Q^j) = (I + (1-\theta)\Delta\tau A^{j-1})v^{j-1} + (1-\theta)\Delta\tau D^{j-1} \tag{3.24}$$

subject to

$$Q_i = \arg\sup_{Q\in\hat{Q}} [A^j(Q)v^j + D(Q)]_i \tag{3.25}$$

To solve Equation (3.24) subject to condition (3.25), we use an iterative process that alternates between a maximization step and a solving step.

Let $Q^{j,k}$ be defined as the elementwise argmax of the objective function

$$Q_i^{j,k} = \arg\sup_{\mu\in\{0,1\}} \left[A(Q^{j,k})v^{j,k-1} + D(Q^{j,k})\right]_i \tag{3.26}$$

Computing $Q^{j,k}$ from Equation (3.26) is known as the *maximization step*.

Then, we solve

$$(I - \theta\Delta\tau A(Q^{j,k}))v^{j,k} = (I + (1-\theta)\Delta\tau A^{j-1})v^{j-1} + \theta\Delta\tau D(Q^{j,k}) + (1-\theta)\Delta\tau D^{j-1}. \tag{3.27}$$

Solving for $v^{j,k}$ from Equation (3.27) is known as the *evaluation step*.

These two steps are key to the policy iteration algorithm and can be seen in numerous sources including [33]. Algorithm 1 gives the high-level description of the policy iteration algorithm for all optimal control problems, including American options.

**Algorithm 1** Policy Iteration for HJB PDEs with $\theta$ timestepping

**Require:** Solve $(I - \theta\Delta\tau A(Q^{j,k}))v^{j,k} = g^{j-1} + \theta\Delta\tau D(Q^{j,k})$
subject to $Q_i = \arg\sup_{Q\in\hat{Q}}[A(Q^j)v^j + D(Q^j)]_i$
where $g^{j-1} = (I + (1-\theta)\Delta\tau A^{j-1})v^{j-1} + (1-\theta)\Delta\tau D^{j-1}$

Initialize $v^{j,0} = v^{j-1}$ and $Q^{j,0} = Q^{j-1}$
**for** $k = 1, \ldots, maxit$ **do**
  Solve $(I - \theta\Delta\tau A(Q^{j,k-1}))v^{j,k} = g^{j-1} + \theta\Delta\tau D^j(Q^{j,k-1})$
  Compute $Q_i^{j,k} = \arg\sup_{Q\in\hat{Q}}[A(Q)v^{j,k} + D(Q)]_i$
  **if** stopping criterion satisfied **then**
    Break
  **end if**
**end for**
Set $v^j = v^{j,k}$ and $Q^j = Q^{j,k}$

The convergence criteria that we use throughout our problems are the convergence of controls: when

$$Q^{j,k} = Q^{j,k+1} \tag{3.28}$$

the iteration is terminated.

Using Condition (3.28) is an improvement over the algorithm described in [14] because the stopping criterion in the latter forces a minimum of two iterations per timestep, but we can do less without deterioration of the solution quality. We later present an improved policy iteration algorithm that handles American exercise rights in a different way and allows variable timestepping and stable order of convergence

The iteration is guaranteed to converge [33], however, the authors provide a worst-case example that can take as many as the number of states to converge. This is not an issue in the problems we are solving for because the authors assume a cold start, but in PDE problems, the solution value does not change greatly per timestep, so we can always use the value at the previous timestep as a warm start.

## 3.3 Penalty Methods

The penalty methods that we introduce are inspired by [7] and [16], which address nonlinearity arising from valuation adjustments and American exercise rights, respectively.

Our nonlinear PDEs from the Borrow-Lend, Stock Borrowing Fees, and Uncertain Volatility problems are of the general form

$$V_\tau = \mathcal{L}(r)V + \text{nonlinear term} \tag{3.29}$$

where the "nonlinear term" is either a maximum or a minimum taken over two or more linear terms involving partial derivatives of $V$ with respect to $S$.

### 3.3.1 American options

For American options, the discrete penalty method is introduced in [16].

Recall that the American Penalty PDE is defined as

$$V_\tau = \frac{\sigma^2 S^2}{2} V_{SS} + rSV_S - rV + \rho \max(V^* - V, 0) \tag{3.30}$$

The challenge in this is the discretization of the penalty term $\rho \max(V^* - V, 0)$. Everything else is the same as for European options.

Recall for European options we solve the system

$$(I - \theta \Delta\tau A)v^j = (I + (1-\theta)\Delta\tau A)v^{j-1} \tag{3.31}$$

For American options, instead of solving a linear system we have to solve Equation (3.33), which is nonlinear, and for that we will use an iterative method.

Define the penalty matrix $P_A$ as a diagonal matrix with the following entries:

$${P_A}_{i,i}^j = P_A(V^j)_{i,i} = \begin{cases} \rho & \text{if } V_i^j < V_i^* \\ 0 & \text{otherwise} \end{cases} \tag{3.32}$$

And we solve the following nonlinear system

$$(I - \theta \Delta\tau A)v^j = (I + (1-\theta)\Delta\tau A)v^{j-1} + P_A(v^j)(v^* - v^j) \tag{3.33}$$

with a iterative method as described in Algorithm 2

**Algorithm 2** Penalty Iteration scheme for timestepping from $j-1$ to $j$

**Require:** Solve $[(I-\theta\Delta\tau A)+P_A(v^j)]v^j = g^j + P_A(v^j)V^*$
where $g^j = (I+(1-\theta)\Delta\tau A)v^{j-1}$

1: Initialize $v^{j,0} = v^{j-1}$ and $P^{j,0} = P(v^{j-1})$
2: **for** $k = 1, \ldots, maxit$ **do**
3: Solve $[(I-\theta\Delta\tau A)+P_A^{j,k-1}]v^{j,k} = g^j + P_A^{j,k-1}V^*$
4: Compute $P_A^{j,k} = P_A(v^{j,k})$
5: **if** stopping criterion satisfied **then**
6: Break
7: **end if**
8: **end for**
9: Set $v^j = v^{j,k}$

There are two different termination conditions. The first, from [14], is

$$\max_i \frac{|v_i^{k+1} - v_i^k|}{\max(1, v_i^{k+1})} \leq tol, \tag{3.34}$$

with $tol$ chosen to be sufficiently small such as $10^{-8}$. The other stopping criterion is

$$P_A^{j,k} = P_A^{j,k+1}. \tag{3.35}$$

The benefits of Condition (3.35) is that if we have converged at the $k$-th iteration, we do not need to compute the next iteration to check convergence. Only checking for equivalence of the penalty matrix is sufficient. The first stopping criterion (Condition (3.34)) is a typical criterion for iterative methods borrowed from [16].

The choice of $\rho$ is determined by the relative accuracy required. We follow the analysis provided in [16]. In

$$[(I-\theta\Delta\tau A)+P_A^j]v^j = g^j + P_A^j V^*, \tag{3.36}$$

assume that $(I-\theta\Delta\tau A)v^{j,k}$ and $g^j$ are bounded independently of $\rho$. Then, as $\rho \to \infty$, for the nodes that violate the constraint $v_i^j > v_i^*$, Equation (3.36) reduces to

$$(1+\rho)v_i^j = 1 + \rho v_i^*, \tag{3.37}$$

Then, we can write $v_i^j$ approximately in terms of $v_i^*$,

$$v_i^j = \frac{1}{1+\rho} + \frac{\rho}{1+\rho} v_i^* \approx \frac{\rho}{1+\rho} v_i^* \tag{3.38}$$

Finally, we take the difference to get

$$v_i^* - v_i^j \approx (1 - \frac{\rho}{1+\rho}) v_i^* = (\frac{1}{1+\rho}) v_i^* \tag{3.39}$$

Or in terms of relative precision,

$$\frac{v_i^* - v_i^j}{v_i^*} \approx \frac{1}{1+\rho} \approx \rho^{-1}. \tag{3.40}$$

Hence, the choice of $\rho$ should be approximately the inverse of the amount of the relative accuracy required.

### 3.3.2 Borrow-Lend

Recall the PDE for short position of the Borrow-Lend problem:

$$V_\tau = \mathcal{L}(r_b)V + (r_b - r_l)\max(V - SV_S, 0) \tag{3.41}$$

The challenge in the discretization of this PDE is the $\max$ term, since $\mathcal{L}(r_b)V$ is simply the Black-Scholes operator. We have two algorithms for discretization, one that is a direct analogy of the American penalty PDE problem and the other is a modification.

Recall that $S_0 < S_1 < \ldots < S_N$ are the spatial discretization points, with $S_0 = 0$ and $S_N = S_{\max}$. Recall also that $v^j$ is the vector of approximate $V(\tau_j, \cdot)$ values computed at $S_i$, $i = 1, \ldots, N$, and $T_1$, $T_2$ are the $N+1 \times N+1$ tridiagonal matrices arising from the discretization of $V_S$ and $V_{SS}$ respectively, by centered differences at $S_i$, $i = 1, \ldots, N-1$, and by backward differences at the far-end point. Note that it doesn't matter how we compute the derivative at $S_0 = 0$, since during computation we would multiply it by zero anyways. Let $D_S$ be the $(N+1) \times (N+1)$ diagonal matrix diag$\{S_0, S_1, \ldots, S_N\}$.

Let $P_d^j = P_d(v^j)$ be the diagonal matrix defined by

$$[P_d]_{ii} = \begin{cases} r_b - r_l & \text{if } v_i^j - [D_S T_1 v^j]_i > 0 \\ 0 & \text{otherwise.`} \end{cases} \tag{3.42}$$

Then, $P_d(V - D_S T_1 V)$ calculates $\max(V - SV_S, 0)$.

Define the tridiagonal penalty matrix $P = P_d(I - D_S T_1)$. Then, $PV$ calculates $\max(V - SV_S, 0)$.

For the long position, the PDE is

$$V_\tau = \mathcal{L}(r_b)V + (r_l - r_b)\max(SV_S - V, 0) \tag{3.43}$$

The penalty matrix $P_d$ is now defined as

$$[P_d]_{ii} = \begin{cases} r_l - r_b & \text{if } [D_S T_1 v^j]_i - v_i^j > 0 \\ 0 & \text{otherwise.`} \end{cases} \tag{3.44}$$

and the tridiagonal penalty matrix $P$ has a similar definition i.e. $P = P_d(D_S T_1 - I)$.

Since the tridiagonal penalty iteration performs better than the diagonal penalty iteration (see chapter 4), we only have tridiagonal penalty methods for the more complicated Stock Borrowing Fee problem.

### 3.3.3 Stock Borrowing Fees

Recall the PDE of interest for the short position:

$$V_\tau = \mathcal{L}(r_l)V + \max\{(r_b - r_l)(SV_S - V), -r_f SV_S, 0\} \tag{3.45}$$

Let now $A$, $P_1$ and $P_2$ the tridiagonal matrices arising from the discretization of $\mathcal{L}(r_l)V$, $(r_b - r_l)(SV_S - V)$ and $-r_f SV_S$, respectively. Note that $P_1 = (r_b - r_l)(D_S T_1 - I)$ and $P_2 =$

$-r_f D_S T_1$. Define the tridiagonal penalty matrix $P = P(v^j)$ by

$$P_{i,:} = \begin{cases} 0 & \text{if } [P_1 v^j]_i \leq 0 \text{ and } [P_2 v^j]_i \leq 0 \\ [P_1]_{i,:} & \text{if } [P_1 v^j]_i > 0 \text{ and } [P_1 v^j]_i > [P_2 v^j]_i \\ [P_2]_{i,:} & \text{if } [P_2 v^j]_i > 0 \text{ and } [P_1 v^j]_i \leq [P_2 v^j]_i, \end{cases} \tag{3.46}$$

where the colon notation $P_{i,:}$ means the entire $i$-th row of $P$ and is borrowed from MATLAB.

Long positions are similar: the PDE becomes

$$V_\tau = \mathcal{L}(r_b)V + \min\{(r_l - r_b)(SV_S - V), -(r_b - r_l + r_f)SV_S, 0\} \tag{3.47}$$

and the nonlinear term to be discretized is

$$\min\{(r_l - r_b)(SV_S - V), -(r_b - r_l + r_f)SV_S, 0\} \tag{3.48}$$

In similar fashion as the short position, let $P_1$ and $P_2$ be the tridiagonal matrices arising from the discretization of $(r_l - r_b)(SV_S - V)$ and $-(r_b - r_l + r_f)SV_S$, respectively. Note that $P_1 = (r_l - r_b)(D_S T_1 - I)$ and $P_2 = -(r_b - r_l + r_f)D_S T_1$. Define the tridiagonal penalty matrix $P = P(v^j)$ by

$$P_{i,:} = \begin{cases} 0 & \text{if } [P_1 v^j]_i > 0 \text{ and } [P_2 v^j]_i > 0 \\ [P_1]_{i,:} & \text{if } [P_1 v^j]_i < 0 \text{ and } [P_1 v^j]_i < [P_2 v^j]_i \\ [P_2]_{i,:} & \text{if } [P_2 v^j]_i < 0 \text{ and } [P_1 v^j]_i > [P_2 v^j]_i, \end{cases} \tag{3.49}$$

For the Stock Borrowing Fee problem with American exercise rights,

$$V_\tau = \mathcal{L}(r_l)V + \max\{(r_b - r_l)(SV_S - V), -r_f SV_S, 0\} + \rho \max(V^* - V, 0) \tag{3.50}$$

and we can apply both this penalty matrix and $P_A$ defined earlier.

### 3.3.4 Uncertain Volatility

For the Uncertain Volatility Model, our nonlinear PDEs are

$$V_\tau = \frac{\sigma_{\min}^2 S^2}{2} V_{SS} + rSV_S - rV + \max\left\{ \frac{(\sigma_{\max}^2 - \sigma_{\min}^2)S^2}{2} V_{SS}, 0 \right\} \tag{3.51}$$

for the best case, and

$$V_\tau = \frac{\sigma_{\max}^2 S^2}{2} V_{SS} + rSV_S - rV + \max\left\{ \frac{(\sigma_{\min}^2 - \sigma_{\max}^2)S^2}{2} V_{SS}, 0 \right\} \tag{3.52}$$

for the worst case.

The tridiagonal penalty matrix $P = P(v^j)$ is defined by

$$P_{i,:} = \begin{cases} \frac{1}{2}(\sigma_d)[D_S^2 T_2]_{i,:} & \text{if } [D_S^2 T_2 v^j]_i > 0 \\ 0 & \text{otherwise,} \end{cases} \tag{3.53}$$

where $\sigma_d$ is $\sigma_{\max}^2 - \sigma_{\min}^2$ in the case of solving for the best case, and $\sigma_{\min}^2 - \sigma_{\max}^2$ when solving for the worst case.

### 3.3.5 Transaction Cost models

Although the original transaction cost model [24] is mathematically equivalent to an uncertain volatility model, the models used in [37] differ slightly. Recall that the nonlinear term is $-\kappa S^2 |V_{SS}|$.

Hence, the matrix arising from the nonlinearity in transaction costs is similar to the matrix arising from the uncertain volatility problem. The tridiagonal penalty matrix $P = P(v^j)$ is defined by

$$P_{i,:} = \begin{cases} -\kappa[D_S^2 T_2]_{i,:} & \text{if } [D_S^2 T_2 v^j]_i > 0 \\ \kappa[D_S^2 T_2]_{i,:} & \text{otherwise,} \end{cases} \tag{3.54}$$

It is easily seen that $Pv$ computes $-\kappa S^2 |V_{SS}|$

### 3.3.6 Algorithm Descriptions

Now, we are ready to give a general form of the penalty algorithms developed throughout the previous section.

At the $j$th timestep, the PDEs (3.41), (3.45), (3.51), and (3.52) can be discretized as

$$(I - \theta\Delta\tau(A + P(v^j)))v^j = (I + (1-\theta)\Delta\tau(A + P(v^{j-1})))v^{j-1} \tag{3.55}$$

If options with American exercise rights are involved, such as the PDEs (3.50) and the related long position, we solve a system similar to (3.33)

$$(I - \theta\Delta\tau(A + P(v^j)))v^j = (I + (1-\theta)\Delta\tau(A + P(v^{j-1})))v^{j-1} + P_A(v^j)(v^* - v^j) \tag{3.56}$$

Let $P$ denote the penalty-like matrix arising from the max terms involving partial derivatives, and let $P_A$ denote the penalty matrix arising from the American constraint, if considering options with American exercise rights. If not, then disregard that matrix entirely. Then, the general form of the penalty iteration algorithm becomes Algorithm 3

**Algorithm 3** General form of tridiagonal penalty iteration at step $j$, with $\theta$-timestepping

**Require:** Solve $[(I - \theta\Delta\tau(A + P(v^j))) + P_A(v^j)]v^j = g^j + P_A(v^j)V^*$
where $g^j = (I + (1-\theta)\Delta\tau(A + P(v^{j-1})))v^{j-1}$

Initialize $v^{j,0} = v^{j-1}$ and $P^{j,0} = P(v^{j-1})$
**for** $k = 1, \ldots, maxit$ **do**
  Solve $[(I - \theta\Delta\tau(A + P^{j,k-1})) + P_A^{j,k-1}]v^{j,k} = g^j + P_A^{j,k-1}V^*$
  **if** first stopping criterion satisfied **then**
    Break
  **end if**
  Compute $P^{j,k} = P(v^{j,k})$, $P_A^{j,k} = P_A(v^{j,k})$
  **if** second stopping criterion satisfied **then**
    Break
  **end if**
**end for**
Set $v^j = v^{j,k}$

The first stopping criterion is

$$\max_i \left( \frac{|v_i^{j,k-1} - v_i^{j,k}|}{\max(scale, |v_i^{j,k}|)} \right) < tol \tag{3.57}$$

with $scale = 1$ and $tol = 10^{-5}$.

The second stopping criterion is

$$P^{j,k} = P^{j,k-1} \text{ and } P_A^{j,k} = P_A^{j,k-1} \tag{3.58}$$

When either condition is satisfied the iteration is terminated.

The second stopping criterion is analogous to the equivalence of controls for the policy iteration. The first stopping criterion is borrowed from [14] and is used to prevent excess iterations in the case of degenerate problems that behave like a simpler problem, such as a transaction cost model with a Put payoff.

Additionally, we have a "diagonal penalty method" (Algorithm 4) which is closer in spirit to the original penalty method as described in [16]. However, the "tridiagonal penalty method" should be preferred due to better convergence properties. Notably, it takes far fewer iterations, and is monotone except for certain cases, whereas the diagonal penalty method is never monotone and a minimum of two iterations are required per timestep.

**Algorithm 4** Diagonal penalty iteration for the Borrow-Lend problem at step $j$, with $\theta$-timestepping

**Require:** Solve $(I - \theta\Delta\tau(A + P_d^j - P_d^j D_S T_1))v^j = g^j$
where $g^j = (I + (1-\theta)\Delta\tau(A + P_d^{j-1} - P_d^{j-1} D_S T_1))v^{j-1} + bnd^j$

Initialize $v^{j,0} = v^{j-1}$ and $P_d^{j,0} = P_d(v^{j-1})$
**for** $k = 1, \ldots, maxit$ **do**
  Solve $(I - \theta\Delta\tau(A + P_d^{j,k-1}))v^{j,k} = g^j - \theta\Delta\tau P_d^{j,k-1} D_S T_1 v^{j,k-1}$
  **if** stopping criterion satisfied **then**
    Break
  **end if**
  Compute $P_d^{j,k} = P_d(v^{j,k})$
**end for**
Set $v^j = v^{j,k}$

The single stopping criterion used is the equivalence of penalty matrices, since we do not have degenerate cases for the Borrow-Lend problem.

## 3.4 Improved Policy Iteration Algorithm for American exercise rights

Due to the fact that the variable timesteps for American exercise rights do not work with Algorithm 1, we present a modified policy iteration algorithm specifically for problems with American exercise rights.

The motivation can be found by analyzing the difference between the Penalty Iteration algorithm [16] and the policy iteration algorithm in [14] for American Put options. It can be determined that the maximization step is equivalent to calculating the penalty matrix; however, the difference lies in the solution of the linear system. It can be shown that solving for the next iterate $v^{j,k}$ in the policy iteration is solving

$$(\mathbf{I} - \theta\Delta\tau A + P_A^{j,k-1})v^{j,k} = (\mathbf{I} + (1-\theta)\Delta\tau A)v^{j-1} + P_A^{j,k-1}v^* + P_A^{j-1}(v^* - v^{j-1}) \quad (3.59)$$

when $\theta = 1/2$.

In contrast, the Penalty Iteration algorithm [16] solves

$$(\mathbf{I} - \theta\Delta\tau A + P_A^{j,k-1})v^{j,k} = (\mathbf{I} + (1-\theta)\Delta\tau A)v^{j-1} + P_A^{j,k-1}v^* \quad (3.60)$$

regardless of what the value of $\theta$ is.

Let the diagonal matrix $R(\mu)$ be defined with the diagonal elements as $\rho\mu$ (recall that $\mu$ is a vector of controls). This leads us to Algorithm 5.

The first stopping criterion is the same as in Algorithm 3 which is Condition (3.57). For the second stopping criterion, if we use a similar condition as the one used in Algorithm (1) based on equivalence of controls

$$\mu^{j,k} = \mu^{j,k-1} \text{ and } Q^{j,k} = Q^{j,k-1} \quad (3.61)$$

then we run into problems with oscillations that can lead to infinite iterations. Instead, we use

**Algorithm 5** Policy Iteration for HJB PDEs with American exercise rights with $\theta$ timestepping

**Require:** Solve $(\mathbf{I} - \theta\Delta\tau A(Q^j))v^j = g^{j-1}$
subject to $Q_i^j = \arg\sup_{Q\in\hat{Q}}[A(Q^j)v^j]_i$ and $\mu_i^j = \arg\sup_{\mu\in\{0,1\}}[R(\mu^j)(v^* - v)]_i$
where $g^{j-1} = (\mathbf{I} + (1-\theta)\Delta\tau A^{j-1})v^{j-1}$
Initialize $v^{j,0} = v^{j-1}$, $\mu^{j,0} = \mu^{j-1}$, and $Q^{j,0} = Q^{j-1}$
**for** $k = 1, \ldots, maxit$ **do**
  Solve $(\mathbf{I} - \theta\Delta\tau A(Q^{j,k-1}) + R(\mu^{j,k-1}))v^{j,k} = g^{j-1} + R(\mu^{j,k-1})v^*$
  **if** first stopping criterion satisfied **then**
    Break
  **end if**
  Compute $Q_i^{j,k} = \arg\sup_{Q\in\hat{Q}}[A(Q)v^{j,k}]_i$, $\mu_i^{j,k} = \arg\sup_{\mu\in\{0,1\}}[R(\mu)(v^* - v)]_i$
  **if** second stopping criterion satisfied **then**
    Break
  **end if**
**end for**
Set $v^j = v^{j,k}$, $\mu^j = \mu^{j,k}$, and $Q^j = Q^{j,k}$

a stopping criterion similar to Condition (3.57)

$$\mu^{j,k} = \mu^{j,k-1} \text{ and } \max_i \left( \frac{|A(Q^{j,k})v^{j,k} - A(Q^{j,k-1})v^{j,k}|}{\max(scale, A(Q^{j,k})v^{j,k})} \right) < tol. \tag{3.62}$$

The idea of this is to prevent excess iterations (that can possibly lead to infinite iterations) in the case of degenerate problems.

## 3.5 Diagonal Dominance, Monotonicity, and Convergence

In this section, we prove that the matrices we use are strictly diagonally dominant (assuming Dirichlet boundary conditions are used), with positive diagonal entries, and nonpositive off-diagonal entries.

This shows the monotonicity [26] of the matrices, which helps to prove the monotonicity of the iterates in the penalty method. Monotonicity of the iterates is used with stability and consistency to prove convergence.

### 3.5.1 Black-Scholes operator

First we consider the Black-Scholes PDE:

$$V_\tau = \frac{\sigma^2 S^2}{2} V_{SS} + rSV_S - rV \equiv \mathcal{L}(r)V \tag{3.63}$$

After we apply CN discretization, we are interested in the matrix $I - \theta\Delta\tau(A + P)$, where $A$ is the discretization matrix of $\mathcal{L}$ and $P$ is the penalty matrix. By showing that $A + P$ is strictly diagonally dominant with negative diagonal entries, we establish that $I - \theta\Delta\tau(A+P)$ is strictly diagonally dominant with positive diagonal entries.

We consider a grid $0 = S_0 < S_1 < S_2 < \cdots < S_n = S_{\max}$, and define $h_i = S_i - S_{i-1}$

Using second-order finite differences for $V_{SS}$ and $V_S$, we get the following for each row $i$ of $A$:

$$A_{i,i-1} = \frac{\sigma^2 S_i^2}{h_i(h_i + h_{i+1})} - \frac{rS_i h_{i+1}}{h_i(h_i + h_{i+1})} \tag{3.64a}$$

$$A_{i,i} = -\frac{\sigma^2 S_i^2}{h_i + h_{i+1}} + \frac{(h_{i+1} - h_i)rS_i}{h_i h_{i+1}} - r \tag{3.64b}$$

$$A_{i,i+1} = \frac{\sigma^2 S_i^2}{h_{i+1}(h_i + h_{i+1})} + \frac{rS_i h_i}{h_{i+1}(h_i + h_{i+1})} \tag{3.64c}$$

Since $A$ is tridiagonal, the strict diagonal dominance is established by showing that

$$|A_{i,i}| - |A_{i,i-1}| - |A_{i,i+1}| > 0 \tag{3.65}$$

We make the assumption that

$$h_{i+1} < \sigma^2 S_i / r \tag{3.66}$$

which is the condition for making diagonal entries negative and the off-diagonal entries positive.

Since

$$h_{i+1} = S_{i+1} - S_i, \tag{3.67}$$

the inequality (3.66) becomes

$$S_{i+1} < (1 + \sigma^2/r)S_i \tag{3.68}$$

which states that the relative increase in each of the gridpoints cannot exceed $\sigma^2/r$.

Then,

$$\begin{aligned} |A_{i,i}| - |A_{i,i-1}| - |A_{i,i+1}| &= -A_{i,i} - A_{i,i-1} - A_{i,i+1} \\ &= -(A_{i,i} + A_{i,i-1} + A_{i,i+1}) \end{aligned}$$

Since $A$ arises from discretization formulas involving $V_{SS}$ and $V_S$, if we take their sum of the entries, the lower-order terms which compute the derivatives sum to $0$ and we just end up with $-r$ as the sum. Since (3.65) is satisfied, we have shown diagonal dominance of the matrix $A$.

This result can also be seen from the fact that a constant function has zero derivative. This is because computing the sum of the entries is equivalent to multiplication with a vector of ones.

### 3.5.2 Diagonal/Tridiagonal Penalty Matrix for Borrow-Lend

Adding the diagonal penalty matrix $P$ only adds $r_b - r_l$ to some diagonal entries of the matrix $A$, which doesn't affect the diagonal dominance. The rows simply sum to $-r_l$ instead of $-r_b$.

The tridiagonal penalty matrix $P$ is like the matrix $A$ except with no second derivative terms and some rows set to 0. For the nonzero rows, they still have negative entries on the diagonals and positive entries on the non-diagonals. Since $P$ arises from discretization, when the sum is taken we are left only with the $(r_b - r_l)V$ terms. If we add $P$ and $A$, since the signs of both the diagonal and off-diagonal entries match, the rows of $A + P$ sum to $-r_b$ if the corresponding row of $P$ is zero, and sum to $-r_l$ if the corresponding row of $P$ is nonzero. Therefore, diagonal dominance is preserved.

### 3.5.3 Tridiagonal penalty matrix for Stock Borrowing Fees

We prove diagonal dominance of the linear system in the penalty method.

Recall that the PDE (3.45) is

$$V_\tau = \mathcal{L}(r_l)V + \max\{(r_b - r_l)(SV_S - V), -r_f SV_S, 0\}. \tag{3.69}$$

The matrix of interest is

$$(I - \theta\Delta\tau(A + P(v^j))) \tag{3.70}$$

We need to consider the addition of the penalty term. There are 3 cases to consider, corresponding to when each of the terms in the $\max$ are the largest.

We have already proved the cases for when either $(r_b - r_l)(SV_S - V)$ is the largest or $0$ is the largest. They are covered in the Borrow-Lend and the plain European cases, respectively.

So there is only the case when $-r_f SV_S$ is the maximum left to consider.

Since the sign is negative, the terms added to the matrix $A$ by the penalty matrix $P$ goes in the opposite direction of the existing terms. Therefore, the diagonal dominance is enhanced.

For the long position, the term under consideration is

$$\min\{(r_l - r_b)(SV_S - V), -(r_b - r_l + r_f)SV_S, 0\} \tag{3.71}$$

and the analysis is the same: the term $-(r_b - r_l + r_f)$ is always negative since $r_b > r_l$. Therefore, the diagonal dominance is not weakened when the other two terms are the maximum, and is enhanced when stock borrowing fee takes effect.

### 3.5.4 Uncertain Volatility

For the penalty matrix arising from the uncertain volatility model, only a discretization involving the 2nd derivative $V_{SS}$ is computed for the nonzero rows. In the case of computing the maximum with a positive coefficient, the coefficients sum to zero and the signs have the same signs as $A$ does – which ensures that diagonal dominance is preserved. In the case of a negative coefficient, the coefficient has smaller magnitude than than that for the corresponding terms in $A$, and hence, diagonal dominance is also preserved.

### 3.5.5 Transaction Cost model

For the penalty matrix arising from the transaction cost model, the rows all follow the same logic as that for the matrix arising from the Uncertain Volatility model. Whether the row is positive or negative, it does not affect the diagonal dominance of $A$ as long as $\kappa < \sigma^2/2$.

### 3.5.6 Monotonicity and Convergence

Note that we have proved strict diagonal dominance of $A$ or $A + P$ with negative diagonal entries and nonnegative off-diagonal entries for the discretization matrices used in European options, Borrow-Lend problem, and Stock Borrowing Fees.

The linear system we are solving for is $I - \theta\Delta\tau A$ or $I - \theta\Delta\tau(A + P)$. Therefore, since $A$ and $A + P$ are diagonally dominant with negative diagonal entries and nonnegative off-diagonal entries, $I - \theta\Delta\tau A$ and $I - \theta\Delta\tau(A + P)$ are diagonally dominant with positive diagonal entries and nonnegative off-diagonal entries. In other words, $I - \theta\Delta\tau A$ and $I - \theta\Delta\tau(A + P)$ are monotone. A proof of the fact that a strictly diagonally dominant matrix with positive diagonal entries and non-positive off-diagonal entries is monotone can be found in [5]. Using this fact, we can show that some of the discrete penalty-like iteration converges monotonically to the unique solution in a finite number of iterations similar to Theorem 1 of [7].

### 3.5.7 American Penalty Matrix

Penalty matrices arising from American options are treated differently than those which arise from nonlinearity involving the partial derivatives such as in Borrow-Lend and Stock Borrowing Fees. In particular, the matrix under question is

$$I - \theta\Delta\tau(A + P) + P_A \tag{3.72}$$

where $P_A$ denotes the penalty matrix arising from the American exercise feature. Since $P_A$ is a diagonal matrix that has only positive or zero entries, it can only enhance the diagonal dominance (and by extension, the monotonicity) of the matrix $I - \theta\Delta\tau(A + P)$. Therefore, adding an American penalty matrix can only enhance the diagonal dominance of the linear system under consideration, and the arguments about diagonal dominance and monotonicity all hold.

## 3.6 Algorithm Convergence

We will show that under certain conditions, the family of tridiagonal penalty methods (Algorithm 3) converge monotonically to the unique solution in a finite number of iterations.

We make use of the fact that if $B$ is a diagonally dominant matrix with positive diagonal entries and negative off-diagonal entries, then $B$ is a monotone matrix, meaning:

1. $B^{-1}$ exists
2. $B^{-1}_{ij} > 0$ for all $i$, $j$.

Therefore, when solving the system $Bx = y$ for $x$, if $y < 0$ then $x < 0$; and if $y > 0$ then $x > 0$.

### 3.6.1 Monotonicity

We wish to prove the following results:

1. When we have penalty matrices that compute the maximum with a positive coefficient, the iterates are monotone increasing.
2. Similarly, when penalty matrices compute the minimum with a positive coefficient, or the maximum with a negative coefficient, the iterates are monotone decreasing.
3. Since the American constraint causes the iterates to be monotone increasing, the iterates are monotone increasing if the penalty matrices compute the maximum with a positive coefficient (in the first case).

We will prove all three statements. For the penalty iteration, we have to solve

$$[I - \theta\Delta\tau(A + P^{j,k-1})]v^{j,k} = g^j \tag{3.73}$$

at each penalty iteration for $v^{j,k}$, where $P$ is a matrix defined based on the maximum of several terms.

We want to show that $v^{j,k+1} \geq v^{j,k}$. Hence, consider the next penalty iteration

$$[I - \theta\Delta\tau(A + P^{j,k})]v^{j,k+1} = g^j \tag{3.74}$$

and write the equation in the previous iteration (3.73) as

$$[I - \theta\Delta\tau(A + P^{j,k})]v^{j,k} = g^j + \theta\Delta\tau(P^{j,k-1} - P^{j,k})v^{j,k} \tag{3.75}$$

Therefore,

$$[I - \theta\Delta\tau(A + P^{j,k})](v^{j,k+1} - v^{j,k}) = \theta\Delta\tau(P^{j,k} - P^{j,k-1})v^{j,k} \tag{3.76}$$

Since the penalty matrix $P^{j,k}$ is defined based on $v^{j,k}$ and is the maximum with a positive coefficient, then $P^{j,k}v^{j,k} \geq P^{j,k-1}v^{j,k}$.

For the second case, we will also reach Equation (3.76). However, since $P$ is now defined as the minimum of the matrix-vector product $P^{j,k}v^{j,k}$ over some terms (or equivalently, the maximum over the terms with a negative coefficient), we have $P^{j,k}v^{j,k} \leq P^{j,k-1}v^{j,k}$.

Since $[I - \theta\Delta\tau(A + P^{j,k})]$ is a monotone matrix, it follows that

$$v^{j,k+1} - v^{j,k} \geq 0 \text{ if } \theta\Delta\tau(P^{j,k} - P^{j,k-1})v^{j,k} \geq 0 \tag{3.77}$$

$$v^{j,k+1} - v^{j,k} \leq 0 \text{ if } \theta\Delta\tau(P^{j,k} - P^{j,k-1})v^{j,k} \leq 0 \tag{3.78}$$

Therefore, in the first case the iterates are monotonically increasing, and in the second case the iterates are monotonically decreasing.

Since adding in the American penalty matrix causes the iterates to increase on each iteration [16], it wouldn't change the monotonicity if the iterates are monotonically increasing otherwise, but if the iterates are monotonically decreasing otherwise then adding in the American penalty matrix breaks the monotone conditions.

### 3.6.2 Proof of algorithm termination

The basic idea of the proof is to partition the nodes into two or three sets depending on the nonlinear problem we are considering based on the penalty matrix. Since the iterates are increasing/decreasing monotonically, any node that has the entry of the matrix-vector product $[Pv]_i$ maximized/minimized does not change on the next iteration. Additionally, if no nodes change sets, then the penalty matrices between iterations are equal, and Equation (3.35) is satisfied, terminating the iteration.

Since there is only a finite number of nodes, and a finite number of states each node can have, and the iterates are monotone, it follows that algorithm terminates in at most $(N + 1)(m - 1)$ iterations, where $N$ is the number of subintervals and $m$ is the number of states.

However, our empirical results show that most of the time the number of iterations taken per timestep is one or two, with the average being closer to one for most problems.

For problems with an additional American penalty matrix, the proof of algorithm termination doesn't change as long as the monotonicity arguments in Section 3.6.1 hold, since the results regarding the algorithm termination from [16] hold as well.

### 3.6.3 Uniqueness of solution

Recall that we have to solve the nonlinear system

$$[I - \theta\Delta\tau(A + P^{j,k-1})]v^{j,k} = g^j \tag{3.79}$$

with emphasis on the nonlinearity between the penalty matrix $P^{j,k-1}$ and $v^{j,k}$

Suppose that there exist two solutions, $(P_1, v_1)$ and $(P_2, v_2)$. Then,

$$[I - \theta\Delta\tau(A + P_1)]v_1 = g^j \tag{3.80}$$

and

$$[I - \theta\Delta\tau(A + P_2)]v_2 = g^j \tag{3.81}$$

We can rewrite (3.80) as

$$[I - \theta\Delta\tau(A + P_2)]v_1 + \theta\Delta\tau(P_2 - P_1)v_1 = g^j \tag{3.82}$$

and take the difference to get

$$[I - \theta\Delta\tau(A + P_2)](v_2 - v_1) = \theta\Delta\tau(P_2 - P_1)v_1 \tag{3.83}$$

Here, we apply the results on monotonicity earlier to argue that in the case of the penalty matrix being based on a maximum, then the RHS vector is always nonpositive, and if the penalty matrix is based on a minimum, then the RHS vector is always nonnegative. This leads us to conclude that $v_1 \geq v_2$ in the former case and $v_1 \leq v_2$ in the latter case.

Note that we could have started with (3.81) to get

$$[I - \theta\Delta\tau(A + P_1)](v_1 - v_2) = \theta\Delta\tau(P_1 - P_2)v_2 \tag{3.84}$$

and obtained the reverse conclusions, that is, $v_1 \geq v_2$ in the former case and $v_2 \geq v_1$ in the latter case. Therefore, $v_1 = v_2$ and $P_1 = P_2$, and the solution to the nonlinear problem (3.79)

is unique.

For problems with an additional American penalty matrix, the proof of uniqueness follows the same process in [16], assuming again that the results in Section 3.6.1 hold.

To conclude this section, we note for the tridiagonal treatment of the penalty-like matrix involving derivative terms, the family of penalty-like and double penalty methods converge monotonically to the unique solution in a finite number of iterations, with the exception of the double penalty method for the long position of Stock Borrowing Fees with American options due to the differing sign between the penalty matrices.

Additionally, where the payoff is convex, for an American Transaction Cost Model with nonzero transaction cost, there is no guarantee of monotone convergence. However, for concave payoffs, the analyses in Sections 2.9 and 3.6.1 are reversed and the iterates are monotonically increasing due to the same sign of the penalty matrix arising from transaction costs and the American constraint.

# Chapter 4

# Numerical Experiments

In this chapter, we present and discuss the results from our numerical experiments. We start off with the basic problem of European options where we have exact solutions before moving on to more complicated problems where we don't have exact solutions.

## 4.1 Introduction

We first discuss the basics of our presentation: Convergence rates, Computational cost, and Error balancing between space and time.

### 4.1.1 Convergence Rates

When we have the exact solution, the observed rate of convergence is

$$\log_2 \left( \frac{\text{error}_n}{\text{error}_{2n}} \right) \tag{4.1}$$

Since both Crank-Nicolson and centered finite differences are second-order accurate, it is important that this number is close to 2.

For the cases where an exact analytical solution is not available, we test the convergence rate

based on three values, and the observed rate of convergence is

$$\log_2 \left( \frac{\text{difference}_n}{\text{difference}_{2n}} \right) \tag{4.2}$$

where

$$\text{difference}_n = v_{(2n)} - v_{(n)}. \tag{4.3}$$

The above convergence analysis is computed at the strike price, because

1. The error as a function of the underlying $S$ is often greatest at or near the strike price

2. The region near the strike price is of the most practical significance.

In addition to the convergence of the value, we also examine the "Greeks" which have important hedging uses. Ideally, we want the "Greeks" to be smooth, and we also test that the "Greeks" converge in second order.

We will start by showing some results from computation over a uniform grid, but after that we will consider only nonuniform grids due to their superior performance.

### 4.1.2 Error Balancing

Error balancing is when we take into consideration both the error arising from spatial discretization and temporal discretization. By finding an appropriate balance between the time and space error, we reduce the error to an acceptable amount while not incurring excessive computational cost due to the number of timesteps, or due to large number of gridpoints.

All problems were tested on spatial domain $[0, 10K]$, with $K = 100$ and time domain $[0, 1]$, with the exception of the Uncertain Volatility problem.

For the problems tested, when we have an American early exercise right, we use the variable timestepping algorithm described in [16] and used in [9].

For problems without an American early exercise, we use a constant timestep and keep the ratio of the number of nodes to the number of timesteps at $4 : 1$. Increasing the ratio beyond that decreases the quality of the solution substantially.

### 4.1.3 Total Computation Cost

Although in practice there are other costs to computing the discretized equations, we view the solving of the linear systems arising from the implicit time discretization as the primary computational cost. We are justified in making this assumption, as can be seen in Figure 4.1: the majority of the computation time is in the `solve` subroutine, which has over half of the computational time and perhaps more importantly, scales linearly with the product of the number of timesteps and the size of the grid.

Profile Summary

*Generated 28-Jul-2020 10:50:03 using performance time.*

| Function Name | Calls | **Total Time** | Self Time* | Total Time Plot (dark band = self time) |
|---|---|---|---|---|
| plot_europeanProfile | 1 | 46.313 s | 0.038 s | |
| CrankNicolson | 1 | 46.275 s | 5.868 s | |
| CrankNicolson>solve | 6402 | 37.793 s | 37.793 s | |
| nonuniformcfdFull | 1 | 2.614 s | 2.578 s | |
| spdiags | 3 | 0.036 s | 0.036 s | |

Figure 4.1: Code profile for European options, $N = 25600$ and $\Delta\tau = 1/6400$, $T = 1$

For linear problems such as European options, the number of linear systems we need to solve is just the number of timesteps. For nonlinear problems such as every other problem in this thesis, it is the total number of solves to obtain the solution in the case of the policy iteration algorithms, and the total number of penalty iterations in the case of the penalty iteration algorithms.

Since the solution of the linear systems is linear in the size of the system, we scale the number of iterations by the size of the linear system to account for this.

We are interested in comparing the total cost to answer questions such as "What is the most efficient choice of the timestep size and $N$ to obtain a solution accurate to 5 decimal places?".

However, for nonlinear problems solved with penalty methods, the primary source of computational time is in both the solving of linear systems and the computation of the penalty term, as can be seen in Figure 4.2. For nonlinear problems solved with policy iteration, we have a lot of overhead because the matrix $A$ is dependent on the control which changes over time and

needs updating. For both of these cases, the procedure is required every time before we solve a new linear system, so the number of solves is still reflective of the total cost.

Profile Summary
*Generated 28-Jul-2020 11:38:52 using performance time.*

| Function Name | Calls | **Total Time** | Self Time* | Total Time Plot (dark band = self time) |
|---|---|---|---|---|
| plot_uvprofile | 1 | 7.257 s | 0.054 s | |
| penaltyIterationUV | 1 | 7.203 s | 1.854 s | |
| penaltyIterationUV>solve | 2037 | 3.037 s | 3.037 s | |
| penaltyIterationUV>getPenaltyTerm | 2039 | 1.916 s | 1.916 s | |
| nonuniformcfdFull | 2 | 0.397 s | 0.373 s | |
| spdiags | 6 | 0.024 s | 0.024 s | |

Figure 4.2: Code profile from Uncertain Volatility (penalty), $N = 1600$ and $\Delta\tau = 1/400$, $T = 1$

Profile Summary
*Generated 28-Jul-2020 11:39:06 using performance time.*

| Function Name | Calls | **Total Time** | Self Time* | Total Time Plot (dark band = self time) |
|---|---|---|---|---|
| plot_uvprofile | 1 | 7.523 s | 0.059 s | |
| policyIterationUV | 1 | 7.464 s | 3.424 s | |
| policyIterationUV>solve | 2095 | 3.088 s | 3.088 s | |
| policyIterationUV>getA | 2 | 0.531 s | 0.004 s | |
| nonuniformcfdFull | 2 | 0.510 s | 0.471 s | |
| policyIterationUV>policyMax | 2096 | 0.422 s | 0.422 s | |
| spdiags | 6 | 0.039 s | 0.039 s | |
| plot_uvprofile>@(x,t,Q)Q.^2.*x.^2/2 | 2 | 0.017 s | 0.017 s | |
| plot_uvprofile>@(x,t,Q)-r*ones(size(x)) | 2 | 0.000 s | 0.000 s | |
| plot_uvprofile>@(x,t,Q)r*x | 2 | 0.000 s | 0.000 s | |

Figure 4.3: Code profile from Uncertain Volatility (policy), $N = 1600$ and $\Delta\tau = 1/400$, $T = 1$

## 4.2 Convergence of Solution

Here we discuss the convergence of the solution.

### 4.2.1 European Put

Here, we show and discuss the results for the European options. We examine the Put option with parameters in Table 4.1

| Description | Symbol | Value |
|---|---|---|
| Maturity time | $T$ | 1 |
| Strike Price | $K$ | 100 |
| Space Truncation boundary | $S_{\max}$ | 1000 |
| Volatility | $\sigma$ | 0.30 |
| Interest rate | $r$ | 0.05 |

Table 4.1: Numerical values of parameters for pricing European Put option

Tables 4.2 and 4.3 show the comparison between the uniform and nonuniform grid. As can be seen, the nonuniform grid reduces the error by a factor of around 27. Therefore, in the future we will use nonuniform gridpoints for the remaining option pricing problems.

| Nodes | Tstep | Value | Error | Rate | Total Cost |
|---|---|---|---|---|---|
| 51 | 52 | 8.643955 | 7.10e-01 | — | 2.60e+03 |
| 101 | 102 | 9.191791 | 1.62e-01 | 2.13 | 1.02e+04 |
| 201 | 202 | 9.314232 | 4.00e-02 | 2.02 | 4.04e+04 |
| 401 | 402 | 9.344243 | 9.95e-03 | 2.01 | 1.61e+05 |
| 801 | 802 | 9.351711 | 2.49e-03 | 2.00 | 6.42e+05 |
| 1601 | 1602 | 9.353576 | 6.21e-04 | 2.00 | 2.56e+06 |

Table 4.2: Numerical values at strike price for European Put option with linear boundary conditions. Parameters from Table 4.1 and uniform discretization grid used

| Nodes | Tstep | Value | Error | Rate | Total Cost |
|---|---|---|---|---|---|
| 51 | 52 | 9.331110 | 2.31e-02 | — | 2.60e+03 |
| 101 | 102 | 9.348249 | 5.95e-03 | 1.96 | 1.02e+04 |
| 201 | 202 | 9.352710 | 1.49e-03 | 2.00 | 4.04e+04 |
| 401 | 402 | 9.353825 | 3.72e-04 | 2.00 | 1.61e+05 |
| 801 | 802 | 9.354104 | 9.30e-05 | 2.00 | 6.42e+05 |
| 1601 | 1602 | 9.354174 | 2.32e-05 | 2.00 | 2.56e+06 |

Table 4.3: Numerical values at strike price for European Put option with linear boundary conditions. Parameters from Table 4.1 and nonuniform discretization grid used

Additionally, we compute a Richardson extrapolation based on values computed at successive discretization levels. Since the order of accuracy of our numerical scheme is $\mathcal{O}(h^2)$, we can write the computed values at the strike price as

$$v_K^{(n)} = V(T,K) + ch^2 + \mathcal{O}(h^p) \tag{4.4}$$

and

$$v_K^{(2n)} = V(T,K) + \frac{1}{4}ch^2 + \mathcal{O}(h^p), \tag{4.5}$$

for some integer $p > 2$ where $v_K^{(n)}$ denotes the value computed at the strike price for the coarse discretization level and $v_K^{(2n)}$ denotes the value computed at the strike price for the finer discretization level, $V(T,K)$ denotes the exact solution, and $h$ and $c$ are generic constants.

Then, by Richardson extrapolation technique,

$$\begin{aligned} 4v_K^{(2n)} - v_K^{(n)} &= 3V(T,K) + 4(\frac{1}{4}c_1h^2) - c_1h^2 + \mathcal{O}(h^p) \\ \frac{4v_K^{(2n)} - v_K^{(n)}}{3} &= V(T,K) + \mathcal{O}(h^p) \end{aligned}$$

Then, the numerical scheme

$$R_K^{(n,2n)} = \frac{1}{3}(4v_K^{(2n)} - v_K^{(n)}) \tag{4.6}$$

has an order of accuracy of $p > 2$. From the results, we can see that the observed order of accuracy is around four.

| $N$ | $2N$ | Richardson | Error | Convergence |
|---|---|---|---|---|
| 51 | 101 | 9.374402 | 2.02e-02 | — |
| 101 | 201 | 9.355046 | 8.49e-04 | 4.57 |
| 201 | 401 | 9.354247 | 4.97e-05 | 4.09 |
| 401 | 801 | 9.354200 | 3.06e-06 | 4.02 |
| 801 | 1601 | 9.354197 | 1.90e-07 | 4.01 |

Table 4.4: Richardson extrapolation from the values obtained in Table 4.2. Compare to Tables 4.2 and 4.3 to see the comparison in error reduction

We also show the results in Table 4.4 in Figure 4.4 as a plot.

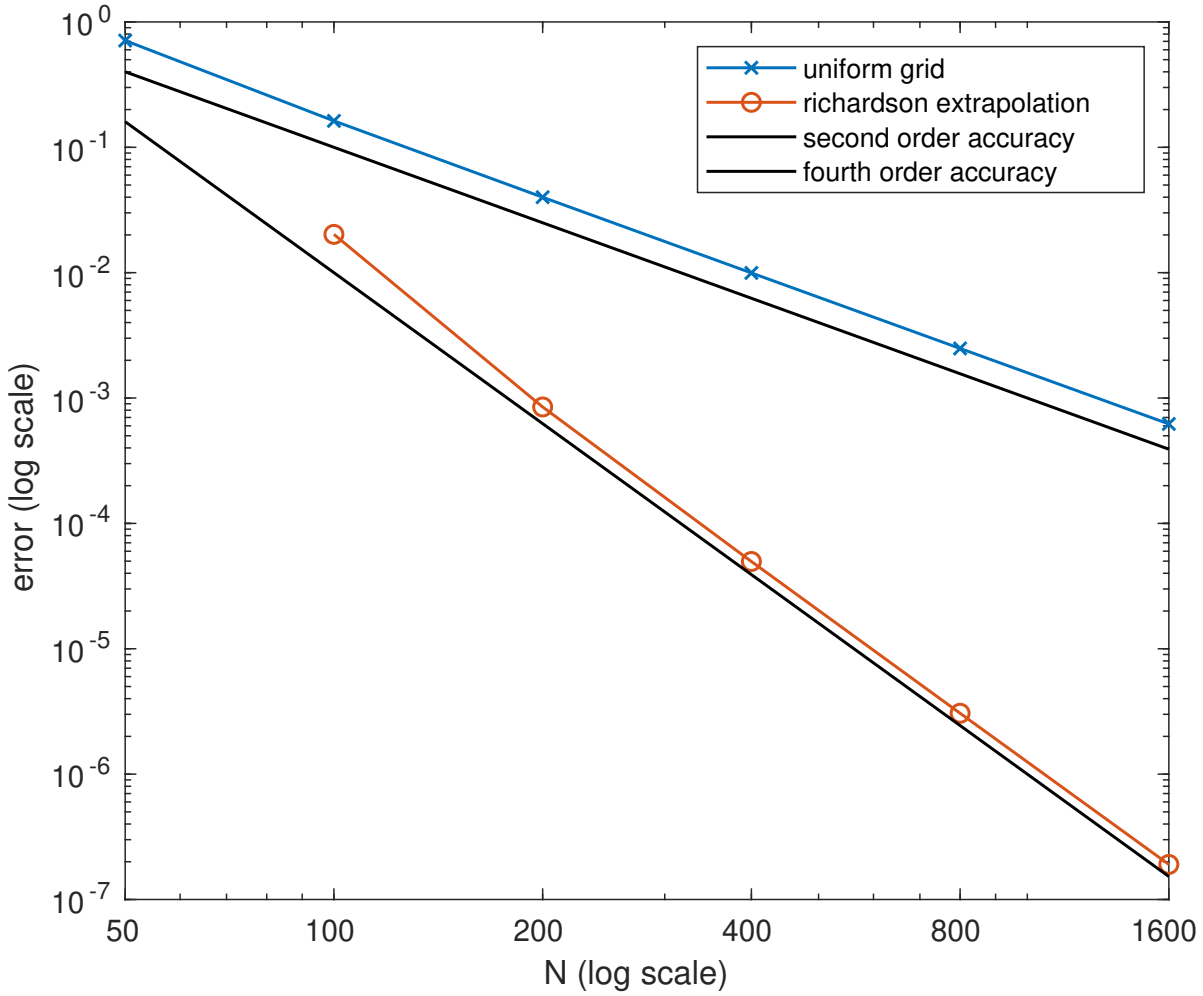


Figure 4.4: Richardson extrapolation for values obtained in Table 4.4. As can be seen the extrapolated value converges in 4th order rather than 2nd order.

Alternatively, the results from the tables are presented in Figure 4.5 which shows that both grids have second order convergence but the nonuniform grid reduces the error by a significant factor.

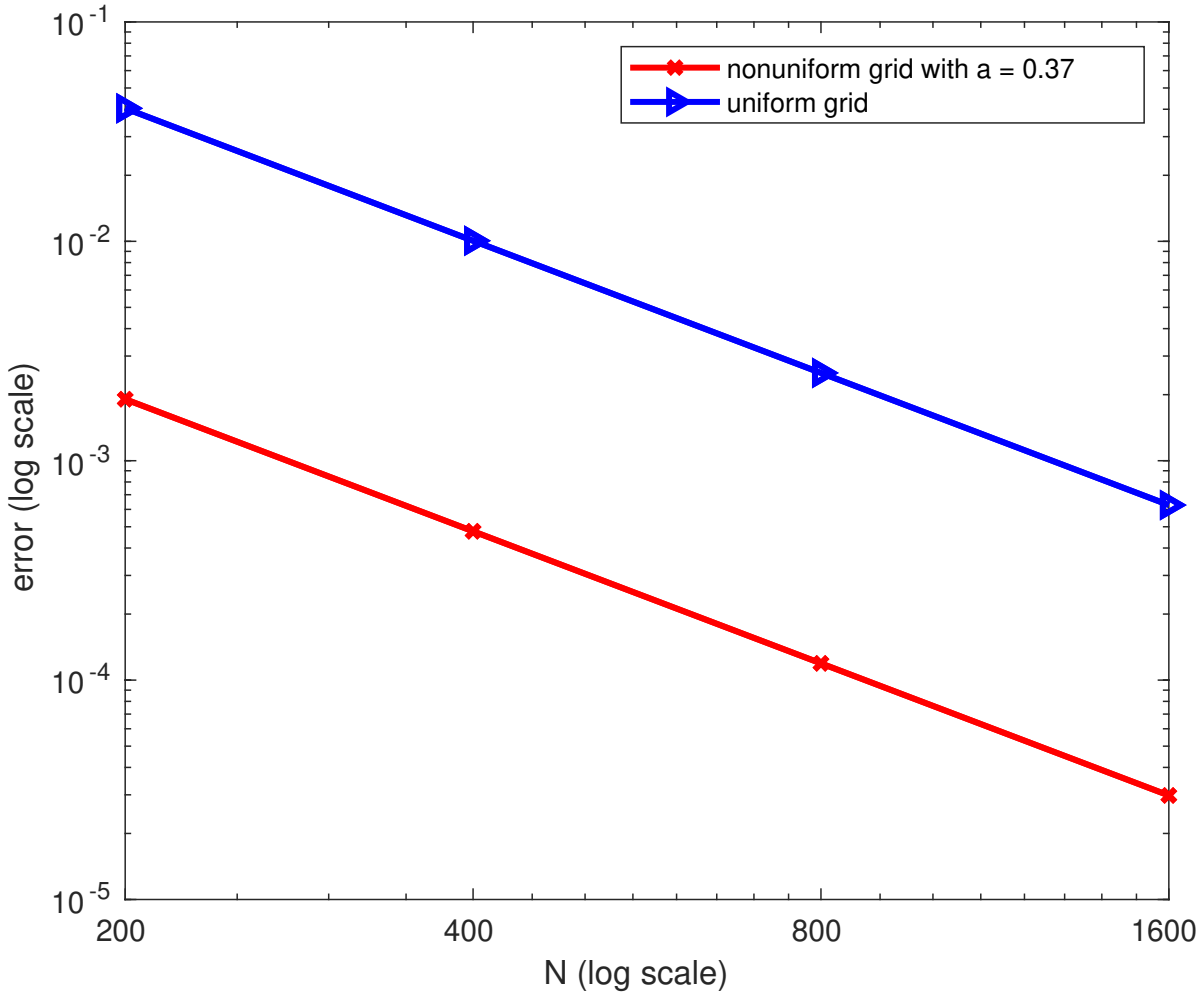


Figure 4.5: Convergence study of European Put option, comparison of uniform and nonuniform grid

Next, we will consider what ratio of number of timesteps to number of nodes is appropriate.

In the following experiment, we keep the computational cost fixed while varying the number of nodes and the number of timesteps. Note that since the computational cost is approximated as

$$\text{cost} \approx \text{number of nodes} \times \text{number of timesteps}, \tag{4.7}$$

the number of timesteps is determined by the number of nodes and the fixed computational cost. A partial table of results is in Table 4.5 and the corresponding figure is Figure 4.6

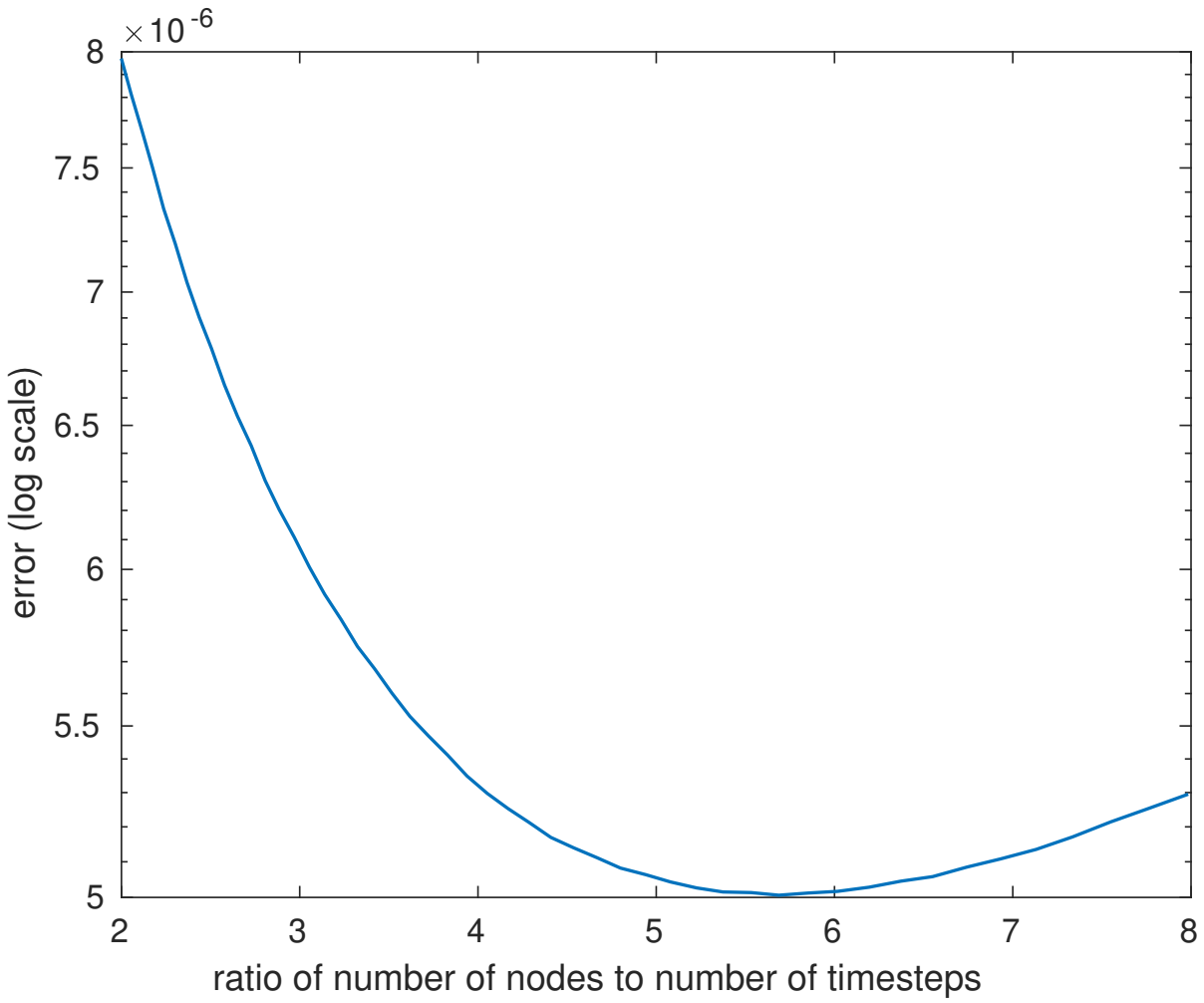


Figure 4.6: Plot of European Options with different ratios of timesteps to nodes. As can be seen the minimum is approximately between 5.5 and 5.9.

| Nodes | Tsteps | Ratio | Cost | Error |
|---|---|---|---|---|
| 4001 | 2002 | 2.00 | 8.01e+06 | 7.97e-06 |
| 5617 | 1426 | 3.94 | 8.01e+06 | 5.35e-06 |
| 5698 | 1406 | 4.05 | 8.01e+06 | 5.30e-06 |
| 6291 | 1273 | 4.94 | 8.01e+06 | 5.06e-06 |
| 6370 | 1256 | 5.08 | 8.01e+06 | 5.04e-06 |
| 6471 | 1238 | 5.23 | 8.01e+06 | 5.03e-06 |
| 6563 | 1221 | 5.37 | 8.01e+06 | 5.02e-06 |
| 6657 | 1203 | 5.53 | 8.01e+06 | 5.01e-06 |
| 6752 | 1187 | 5.69 | 8.01e+06 | 5.01e-06 |
| 6848 | 1170 | 5.85 | 8.01e+06 | 5.01e-06 |
| 6945 | 1154 | 6.02 | 8.01e+06 | 5.02e-06 |
| 7044 | 1137 | 6.19 | 8.01e+06 | 5.03e-06 |
| 7145 | 1121 | 6.37 | 8.01e+06 | 5.05e-06 |
| 8001 | 1002 | 7.98 | 8.02e+06 | 5.30e-06 |

Table 4.5: Parameters from Table 4.1 and nonuniform discretization grid used. Cost is fixed at approximately $8 \times 10^6$.

As can be seen, subject to a fixed cost of approximately $8 \times 10^6$, the error is approximately minimized near a ratio of 5.5-5.9.

Another way of looking at this is examining the plots of the typical convergence plot and a plot of the error vs. cost side by side as in Figure 4.7.

As we can see the lines showing a ratio of 4 and 8 are overlaid, which is consistent with the results in Table 4.5 and Figure 4.6

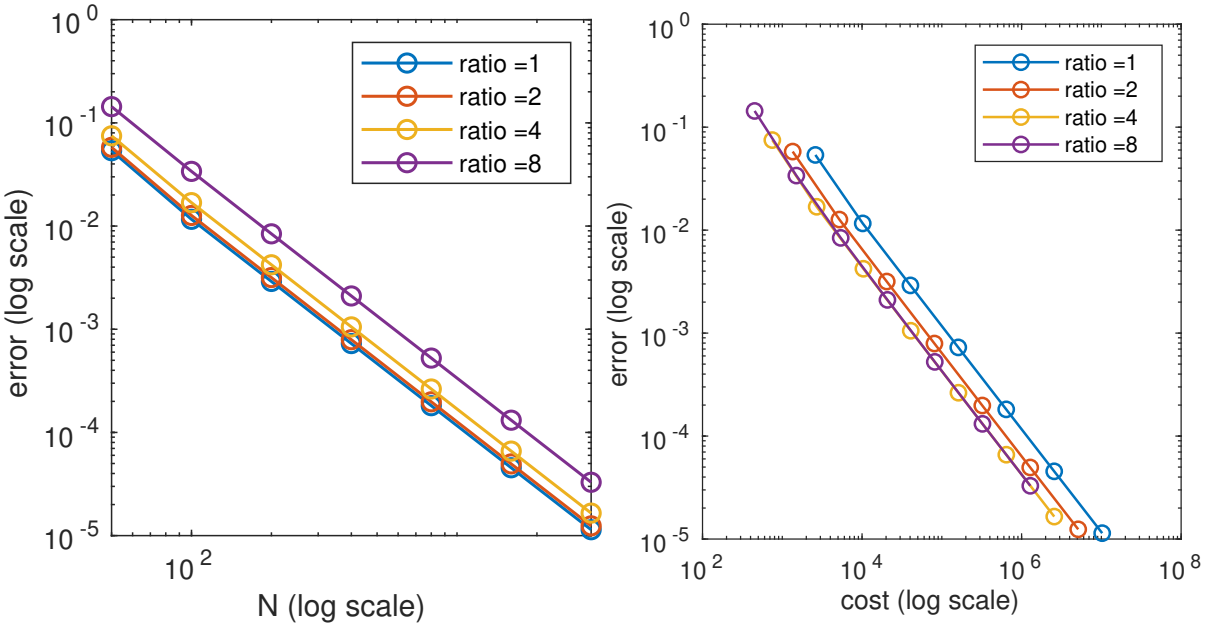


Figure 4.7: Side-by-side plot of error vs. $N$ and error vs. total cost. Note that since total cost is linear in $N^2$, both lines are straight.

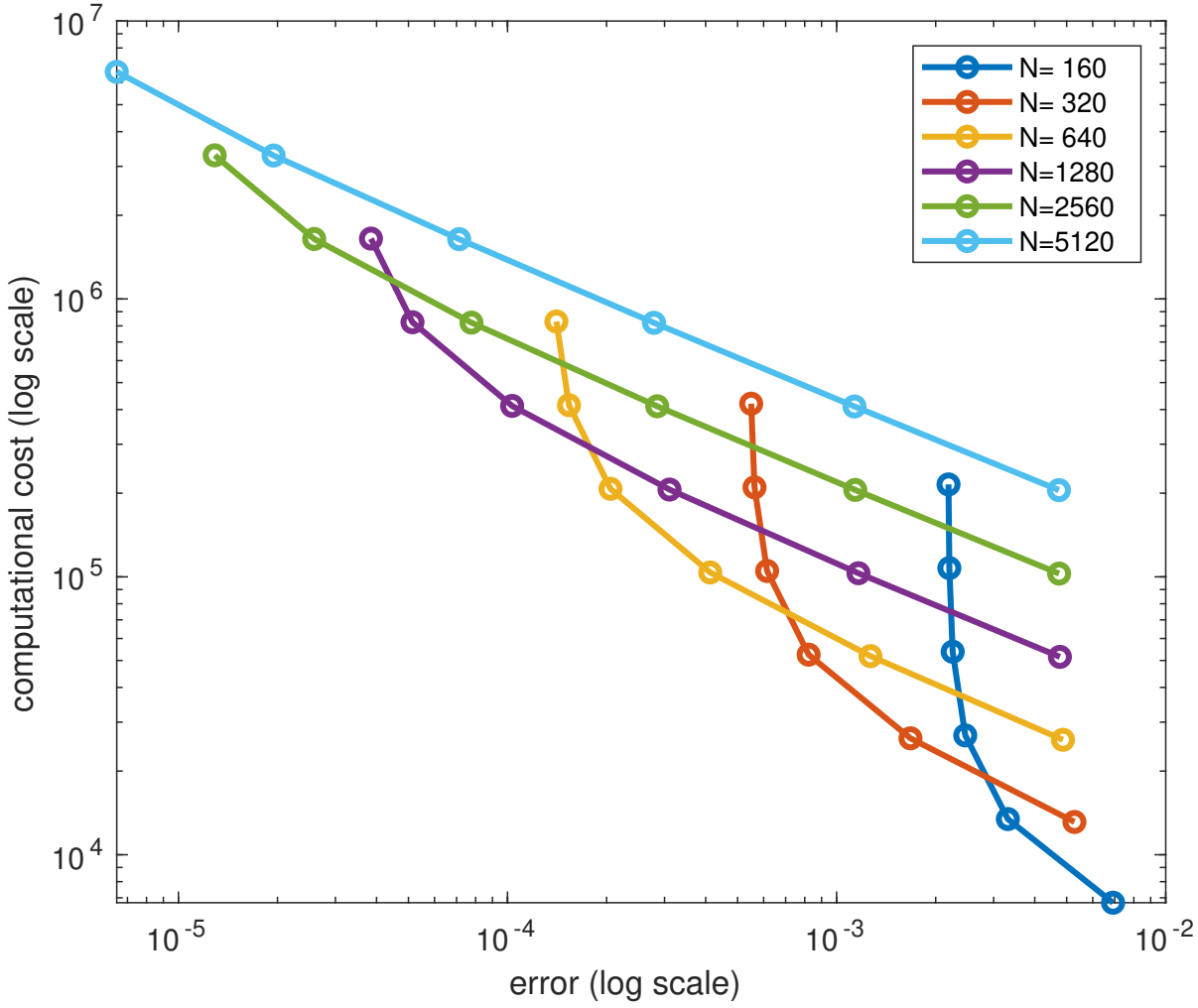


Figure 4.8: Plot of European options with different $N$ and $\Delta\tau$ values. The $\Delta\tau$ values are chosen such that the values of $\Delta\tau$ are $4/N$. In other words, the greatest choice of $\Delta\tau$ has a value of $1/40$ and each subsequent level halves the value.

Figure 4.8 shows the results from different runs, and 36 different cases are considered. We want to answer the questions such as "What is the minimum computational cost required to obtain a solution accurate to $10^{-4}$?". As can be seen from the plot, the purple line ($N = 1280$) is minimized there, and is near the fourth point in the plot. Therefore, when $N = 1280$ and $\Delta\tau = 1/320$, the efficiency of the solver is maximized. A similar ratio is obtained when considering the solution accuracy to be within $10^{-3}$ or $10^{-2}$. Therefore, for European options we check the results of the computation with a $4:1$ ratio of the number of nodes to the number of timesteps, and get Tables 4.6 and 4.7.

| Nodes | Tstep | Value | Error | Rate | Total Cost |
|---|---|---|---|---|---|
| 51 | 15 | 8.635527 | 7.19e-01 | — | 7.50e+02 |
| 101 | 27 | 9.190050 | 1.64e-01 | 2.13 | 2.70e+03 |
| 201 | 52 | 9.313809 | 4.04e-02 | 2.02 | 1.04e+04 |
| 401 | 102 | 9.344138 | 1.01e-02 | 2.01 | 4.08e+04 |
| 801 | 202 | 9.351685 | 2.51e-03 | 2.00 | 1.62e+05 |
| 1601 | 402 | 9.353569 | 6.28e-04 | 2.00 | 6.43e+05 |

Table 4.6: Numerical values at strike price for European Put option with linear boundary conditions. Parameters from Table 4.1 and uniform discretization grid used

| Nodes | Tstep | Value | Error | Rate | Total Cost |
|---|---|---|---|---|---|
| 51 | 15 | 9.324301 | 2.99e-02 | — | 7.50e+02 |
| 101 | 27 | 9.346569 | 7.63e-03 | 1.97 | 2.70e+03 |
| 201 | 52 | 9.352290 | 1.91e-03 | 2.00 | 1.04e+04 |
| 401 | 102 | 9.353720 | 4.77e-04 | 2.00 | 4.08e+04 |
| 801 | 202 | 9.354078 | 1.19e-04 | 2.00 | 1.62e+05 |
| 1601 | 402 | 9.354167 | 2.98e-05 | 2.00 | 6.43e+05 |

Table 4.7: Numerical values at strike price for European Put option with linear boundary conditions. Parameters from Table 4.1 and nonuniform discretization grid used

As can be seen from the tables, when comparing the results in Tables 4.6 and 4.7 to the results in Tables 4.2 and 4.3 for roughly the same computational cost, the error is reduced by a factor of 4 for the uniform grid, and a factor of about 3.5 for the nonuniform grid.

### 4.2.2 American options

We present the results for American options with policy iteration, penalty iteration, and penalty iteration with variable timestepping in Tables 4.8, 4.9, and 4.10 respectively:

| Nodes | Tstep | Iter | Value | Change | Rate | Total Cost | Richardson | avg iter |
|---|---|---|---|---|---|---|---|---|
| 51 | 52 | 60 | 9.842843 | — | — | 3.00e+03 | — | 1.15 |
| 101 | 102 | 123 | 9.862481 | 1.96e-02 | — | 1.23e+04 | 9.869027 | 1.21 |
| 201 | 202 | 247 | 9.867954 | 5.47e-03 | 1.84 | 4.94e+04 | 9.869778 | 1.22 |
| 401 | 402 | 490 | 9.869444 | 1.49e-03 | 1.88 | 1.96e+05 | 9.869940 | 1.22 |
| 801 | 802 | 977 | 9.869868 | 4.24e-04 | 1.81 | 7.82e+05 | 9.870009 | 1.22 |
| 1601 | 1602 | 1938 | 9.869997 | 1.29e-04 | 1.71 | 3.10e+06 | 9.870040 | 1.21 |

Table 4.8: Numerical values at strike price for American Put option with linear boundary conditions, solved with policy iteration (Algorithm 1). Nonuniform discretization grid and constant timesteps are used

| Nodes | Tstep | Iter | Value | Change | Rate | Total Cost | Richardson | avg iter |
|---|---|---|---|---|---|---|---|---|
| 51 | 52 | 63 | 9.842410 | — | — | 3.15e+03 | — | 1.21 |
| 101 | 102 | 125 | 9.862287 | 1.99e-02 | — | 1.25e+04 | 9.868913 | 1.23 |
| 201 | 202 | 249 | 9.867898 | 5.61e-03 | 1.82 | 4.98e+04 | 9.869768 | 1.23 |
| 401 | 402 | 495 | 9.869417 | 1.52e-03 | 1.88 | 1.98e+05 | 9.869923 | 1.23 |
| 801 | 802 | 988 | 9.869858 | 4.41e-04 | 1.78 | 7.90e+05 | 9.870005 | 1.23 |
| 1601 | 1602 | 1975 | 9.869994 | 1.36e-04 | 1.70 | 3.16e+06 | 9.870039 | 1.23 |

Table 4.9: Numerical values at strike price for American Put option with linear boundary conditions, solved with penalty iteration (Algorithm 2). Nonuniform discretization grid and constant timesteps are used

As can be seen the convergence rate is not exactly at 2.0. This can be remedied by use of variable or adaptive timesteps, as given in [16]. Unfortunately we haven't been able to make this timestepping work with the policy iteration algorithm.

The variable timestep selector is defined by

$$\Delta\tau^{j+2} = \left( \min_i \left[ \frac{\text{dnorm}}{\frac{|V(S_i,\tau^j+\Delta\tau^{j+1})-V(S_i,\tau^j)|}{\max(D,|V(S_i,\tau^j+\Delta\tau^{j+1})|,|V(S_i,\tau^j)|}} \right] \right) \Delta\tau^{j+1} \tag{4.8}$$

In our variable timestep selector, $D$ is chosen to be $1$ and we use initial $\Delta\tau = (1/8) \times 10^{-3}$ and $\texttt{dnorm} = 0.3$ for the coarsest discretization level, and divide both quantities by 2 on each subsequent level.

| Nodes | Tstep | Iter | Value | Change | Rate | Total Cost | Richardson | avg iter |
|---|---|---|---|---|---|---|---|---|
| 51 | 41 | 52 | 9.846729 | — | — | 2.60e+03 | — | 1.27 |
| 101 | 76 | 99 | 9.864016 | 1.73e-02 | — | 9.90e+03 | 9.869778 | 1.30 |
| 201 | 144 | 198 | 9.868569 | 4.55e-03 | 1.92 | 3.96e+04 | 9.870086 | 1.38 |
| 401 | 280 | 437 | 9.869693 | 1.12e-03 | 2.02 | 1.75e+05 | 9.870067 | 1.56 |
| 801 | 552 | 932 | 9.869971 | 2.79e-04 | 2.01 | 7.46e+05 | 9.870064 | 1.69 |
| 1601 | 1097 | 1952 | 9.870041 | 6.94e-05 | 2.01 | 3.12e+06 | 9.870064 | 1.78 |

Table 4.10: Numerical values at strike price for American Put option with linear boundary conditions, solved with penalty iteration (Algorithm 2). Nonuniform discretization grid and variable timesteps are used

As can be seen, we now have a higher rate of convergence and very close to the expected result of 2.0. Additionally, the predictions are much more consistent due to the consistency in the rate of convergence and the change, a reflection of the size of the error, is also much smaller (less than half) than that of the uniform timesteps for the same discretization level. Finally, although the number of iterations per timestep has increased, the overall number of iterations has decreased when we use variable timesteps, so there is less of a computational cost.

### 4.2.3 Borrow-Lend

In this section, we look at the results from the three different algorithms for the Borrow-Lend problem. Since the Borrow-Lend problem, Stock Borrowing Fees, and Uncertain Volatility have European exercise rights, and since we did not note any degradation of the order of convergence, we keep the timesteps constant.

In addition, we keep the same ratio of timesteps to nodes as we had for vanilla European options, as we tried different ratios and arrived at the same conclusions as what we got for vanilla European options.

The additional parameters are $r_l = 0.03$ and $r_b = 0.05$.

We present the results using policy iteration, diagonal penalty iteration, and tridiagonal penalty iteration in Tables 4.11, 4.13, and 4.12 respectively. The corresponding results for the long position are in Tables 4.14, 4.16, and 4.15 respectively.

| Nodes | Tstep | Iter | Value | Change | Rate | Total Cost | Richardson | avg iter |
|---|---|---|---|---|---|---|---|---|
| 51 | 15 | 15 | 24.007537 | — | — | 7.50e+02 | — | 1.00 |
| 101 | 27 | 27 | 24.054339 | 4.68e-02 | — | 2.70e+03 | 24.069940 | 1.00 |
| 201 | 52 | 53 | 24.066376 | 1.20e-02 | 1.96 | 1.06e+04 | 24.070388 | 1.02 |
| 401 | 102 | 106 | 24.069384 | 3.01e-03 | 2.00 | 4.24e+04 | 24.070387 | 1.04 |
| 801 | 202 | 209 | 24.070136 | 7.52e-04 | 2.00 | 1.67e+05 | 24.070386 | 1.03 |
| 1601 | 402 | 416 | 24.070324 | 1.88e-04 | 2.00 | 6.66e+05 | 24.070386 | 1.03 |

Table 4.11: Numerical values at strike price for short position of Borrow-Lend pricing problem with linear boundary conditions, solved with policy iteration (Algorithm 1). Nonuniform discretization grid used

| Nodes | Tstep | Iter | Value | Change | Rate | Total Cost | Richardson | avg iter |
|---|---|---|---|---|---|---|---|---|
| 51 | 15 | 16 | 24.007537 | — | — | 8.00e+02 | — | 1.07 |
| 101 | 27 | 28 | 24.054339 | 4.68e-02 | — | 2.80e+03 | 24.069940 | 1.04 |
| 201 | 52 | 54 | 24.066376 | 1.20e-02 | 1.96 | 1.08e+04 | 24.070388 | 1.04 |
| 401 | 102 | 106 | 24.069384 | 3.01e-03 | 2.00 | 4.24e+04 | 24.070387 | 1.04 |
| 801 | 202 | 209 | 24.070136 | 7.52e-04 | 2.00 | 1.67e+05 | 24.070386 | 1.03 |
| 1601 | 402 | 417 | 24.070324 | 1.88e-04 | 2.00 | 6.67e+05 | 24.070386 | 1.04 |

Table 4.12: Numerical values at strike price for short position of Borrow-Lend pricing problem with linear boundary conditions, solved with tridiagonal penalty iteration (Algorithm 3). Nonuniform discretization grid used

| Nodes | Tstep | Iter | Value | Change | Rate | Total Cost | Richardson | avg iter |
|---|---|---|---|---|---|---|---|---|
| 51 | 15 | 52 | 24.007537 | — | — | 2.60e+03 | — | 3.47 |
| 101 | 27 | 86 | 24.054339 | 4.68e-02 | — | 8.60e+03 | 24.069940 | 3.19 |
| 201 | 52 | 158 | 24.066376 | 1.20e-02 | 1.96 | 3.16e+04 | 24.070388 | 3.04 |
| 401 | 102 | 307 | 24.069384 | 3.01e-03 | 2.00 | 1.23e+05 | 24.070387 | 3.01 |
| 801 | 202 | 484 | 24.070136 | 7.52e-04 | 2.00 | 3.87e+05 | 24.070386 | 2.40 |
| 1601 | 402 | 864 | 24.070324 | 1.88e-04 | 2.00 | 1.38e+06 | 24.070386 | 2.15 |

Table 4.13: Numerical values at strike price for short position of Borrow-Lend pricing problem with linear boundary conditions, solved with diagonal penalty iteration (Algorithm 4). Nonuniform discretization grid used

| Nodes | Tstep | Iter | Value | Change | Rate | Total Cost | Richardson | avg iter |
|---|---|---|---|---|---|---|---|---|
| 51 | 15 | 16 | 23.050542 | — | — | 8.00e+02 | — | 1.07 |
| 101 | 27 | 28 | 23.094241 | 4.37e-02 | — | 2.80e+03 | 23.108807 | 1.04 |
| 201 | 52 | 53 | 23.105528 | 1.13e-02 | 1.95 | 1.06e+04 | 23.109290 | 1.02 |
| 401 | 102 | 104 | 23.108351 | 2.82e-03 | 2.00 | 4.16e+04 | 23.109292 | 1.02 |
| 801 | 202 | 205 | 23.109057 | 7.06e-04 | 2.00 | 1.64e+05 | 23.109293 | 1.01 |
| 1601 | 402 | 409 | 23.109234 | 1.76e-04 | 2.00 | 6.54e+05 | 23.109292 | 1.02 |

Table 4.14: Numerical values at strike price for long position of Borrow-Lend pricing problem with linear boundary conditions, solved with policy iteration (Algorithm 1). Nonuniform discretization grid used

| Nodes | Tstep | Iter | Value | Change | Rate | Total Cost | Richardson | avg iter |
|---|---|---|---|---|---|---|---|---|
| 51 | 15 | 15 | 23.050536 | — | — | 7.50e+02 | — | 1.00 |
| 101 | 27 | 28 | 23.094240 | 4.37e-02 | — | 2.80e+03 | 23.108808 | 1.04 |
| 201 | 52 | 52 | 23.105528 | 1.13e-02 | 1.95 | 1.04e+04 | 23.109291 | 1.00 |
| 401 | 102 | 103 | 23.108351 | 2.82e-03 | 2.00 | 4.12e+04 | 23.109292 | 1.01 |
| 801 | 202 | 205 | 23.109057 | 7.06e-04 | 2.00 | 1.64e+05 | 23.109293 | 1.01 |
| 1601 | 402 | 408 | 23.109234 | 1.76e-04 | 2.00 | 6.53e+05 | 23.109292 | 1.01 |

Table 4.15: Numerical values at strike price for long position of Borrow-Lend pricing problem with linear boundary conditions, solved with penalty iteration (Algorithm 3). Nonuniform discretization grid used

| Nodes | Tstep | Iter | Value | Change | Rate | Total Cost | Richardson | avg iter |
|---|---|---|---|---|---|---|---|---|
| 51 | 15 | 51 | 23.050536 | — | — | 2.55e+03 | — | 3.40 |
| 101 | 27 | 85 | 23.094240 | 4.37e-02 | — | 8.50e+03 | 23.108808 | 3.15 |
| 201 | 52 | 157 | 23.105528 | 1.13e-02 | 1.95 | 3.14e+04 | 23.109291 | 3.02 |
| 401 | 102 | 268 | 23.108351 | 2.82e-03 | 2.00 | 1.07e+05 | 23.109292 | 2.63 |
| 801 | 202 | 460 | 23.109057 | 7.06e-04 | 2.00 | 3.68e+05 | 23.109293 | 2.28 |
| 1601 | 402 | 852 | 23.109234 | 1.76e-04 | 2.00 | 1.36e+06 | 23.109292 | 2.12 |

Table 4.16: Numerical values at strike price for long position of Borrow-Lend pricing problem with linear boundary conditions, solved with diagonal penalty iteration (Algorithm 4). Nonuniform discretization grid used

As can be seen, for each of the algorithms the predicted values agree over different runs and also with each other. Additionally, the rate of convergence is 2 throughout. Of notable importance

is that the number of iterations taken by the diagonal penalty iteration is a minimum of two each timestep, and as a result is less efficient compared to the tridiagonal algorithm, hence, for the future problems only tridiagonal penalty iteration is considered.

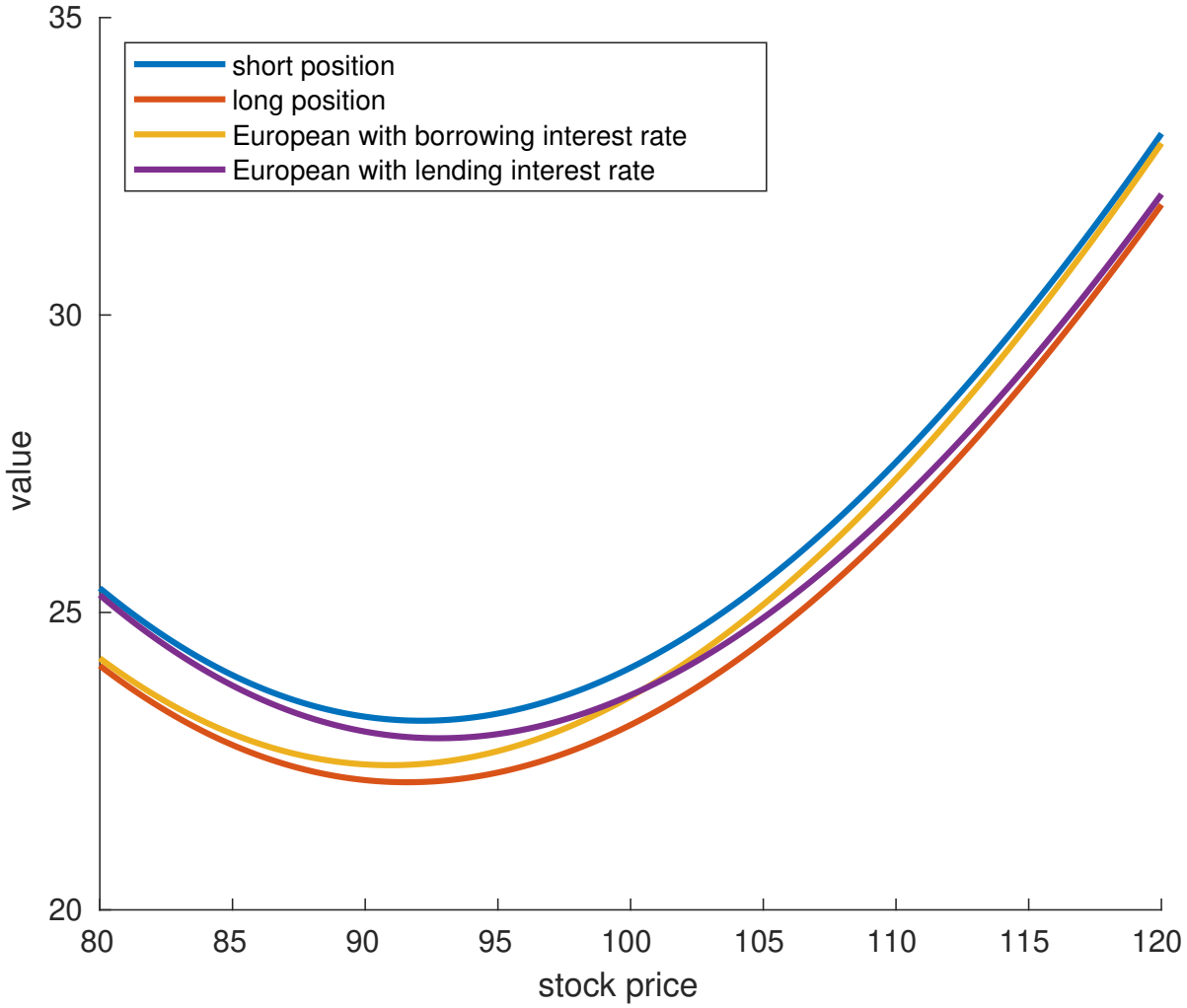


Figure 4.9: Solution curve of Borrow-Lend pricing problem and European options

### 4.2.4 Stock Borrowing Fees

For the Stock Borrowing Fees problem, the additional parameter is $r_f = ---4$. Since no American early exercise right is involved, we again keep the ratio of timesteps to nodes from Borrow-Lend and European options. We present results of the short position from policy and (tridiagonal) penalty iteration algorithms in Tables 4.17 and 4.18, and the corresponding long positions in 4.19 and 4.20.

| Nodes | Tstep | Iter | Value | Change | Rate | Total Cost | Richardson | avg iter |
|---|---|---|---|---|---|---|---|---|
| 51 | 15 | 19 | 24.071318 | — | — | 9.50e+02 | — | 1.27 |
| 101 | 27 | 38 | 24.118360 | 4.70e-02 | — | 3.80e+03 | 24.134041 | 1.41 |
| 201 | 52 | 73 | 24.130491 | 1.21e-02 | 1.96 | 1.46e+04 | 24.134535 | 1.40 |
| 401 | 102 | 144 | 24.133523 | 3.03e-03 | 2.00 | 5.76e+04 | 24.134534 | 1.41 |
| 801 | 202 | 282 | 24.134280 | 7.57e-04 | 2.00 | 2.26e+05 | 24.134533 | 1.40 |
| 1601 | 402 | 562 | 24.134470 | 1.89e-04 | 2.00 | 8.99e+05 | 24.134533 | 1.40 |

Table 4.17: Numerical values at strike price for short position of Stock Borrowing Fee pricing problem with linear boundary conditions, solved with policy iteration (Algorithm 1). Nonuniform discretization grid used

| Nodes | Tstep | Iter | Value | Change | Rate | Total Cost | Richardson | avg iter |
|---|---|---|---|---|---|---|---|---|
| 51 | 15 | 18 | 24.071318 | — | — | 9.00e+02 | — | 1.20 |
| 101 | 27 | 38 | 24.118360 | 4.70e-02 | — | 3.80e+03 | 24.134041 | 1.41 |
| 201 | 52 | 72 | 24.130491 | 1.21e-02 | 1.96 | 1.44e+04 | 24.134535 | 1.38 |
| 401 | 102 | 143 | 24.133523 | 3.03e-03 | 2.00 | 5.72e+04 | 24.134534 | 1.40 |
| 801 | 202 | 282 | 24.134280 | 7.57e-04 | 2.00 | 2.26e+05 | 24.134533 | 1.40 |
| 1601 | 402 | 561 | 24.134470 | 1.89e-04 | 2.00 | 8.98e+05 | 24.134533 | 1.40 |

Table 4.18: Numerical values at strike price for short position of Stock Borrowing Fee pricing problem with linear boundary conditions, solved with penalty iteration (Algorithm 3). Nonuniform discretization grid used

| Nodes | Tstep | Iter | Value | Change | Rate | Total Cost | Richardson | avg iter |
|---|---|---|---|---|---|---|---|---|
| 51 | 15 | 19 | 22.625605 | — | — | 9.50e+02 | — | 1.27 |
| 101 | 27 | 37 | 22.669427 | 4.38e-02 | — | 3.70e+03 | 22.684035 | 1.37 |
| 201 | 52 | 71 | 22.680662 | 1.12e-02 | 1.96 | 1.42e+04 | 22.684407 | 1.37 |
| 401 | 102 | 140 | 22.683470 | 2.81e-03 | 2.00 | 5.60e+04 | 22.684406 | 1.37 |
| 801 | 202 | 278 | 22.684172 | 7.02e-04 | 2.00 | 2.22e+05 | 22.684406 | 1.38 |
| 1601 | 402 | 553 | 22.684347 | 1.75e-04 | 2.00 | 8.85e+05 | 22.684406 | 1.38 |

Table 4.19: Numerical values at strike price for long position of Stock Borrowing Fee pricing problem with linear boundary conditions, solved with policy iteration (Algorithm 1). Nonuniform discretization grid used

| Nodes | Tstep | Iter | Value | Change | Rate | Total Cost | Richardson | avg iter |
|---|---|---|---|---|---|---|---|---|
| 51 | 15 | 19 | 22.625605 | — | — | 9.50e+02 | — | 1.27 |
| 101 | 27 | 36 | 22.669427 | 4.38e-02 | — | 3.60e+03 | 22.684035 | 1.33 |
| 201 | 52 | 71 | 22.680662 | 1.12e-02 | 1.96 | 1.42e+04 | 22.684407 | 1.37 |
| 401 | 102 | 140 | 22.683470 | 2.81e-03 | 2.00 | 5.60e+04 | 22.684406 | 1.37 |
| 801 | 202 | 278 | 22.684172 | 7.02e-04 | 2.00 | 2.22e+05 | 22.684406 | 1.38 |
| 1601 | 402 | 553 | 22.684347 | 1.75e-04 | 2.00 | 8.85e+05 | 22.684406 | 1.38 |

Table 4.20: Numerical values at strike price for long position of Stock Borrowing Fee pricing problem with linear boundary conditions, solved with penalty iteration (Algorithm 3). Nonuniform discretization grid used

### 4.2.5 Stock Borrowing Fees with American options

For American options, we use the variable timestep selector described in [16]. We present results of the short position solved by policy iteration, penalty iteration, and penalty iteration with variable timesteps in Tables 4.21, 4.22, and 4.23. The corresponding results for the long position are presented in Tables 4.24, 4.25, and 4.26 respectively.

Additionally, we also present results from our modified policy iteration algorithm for the short and long positions, with constant and variable timesteps in Tables 4.27, 4.28, 4.29, 4.30

| Nodes | Tstep | Iter | Value | Change | Rate | Total Cost | Richardson | avg iter |
|---|---|---|---|---|---|---|---|---|
| 51 | 52 | 65 | 24.271651 | — | — | 3.25e+03 | — | 1.25 |
| 101 | 102 | 136 | 24.312760 | 4.11e-02 | — | 1.36e+04 | 24.326463 | 1.33 |
| 201 | 202 | 271 | 24.323903 | 1.11e-02 | 1.88 | 5.42e+04 | 24.327618 | 1.34 |
| 401 | 402 | 542 | 24.326884 | 2.98e-03 | 1.90 | 2.17e+05 | 24.327877 | 1.35 |
| 801 | 802 | 1091 | 24.327704 | 8.20e-04 | 1.86 | 8.73e+05 | 24.327977 | 1.36 |
| 1601 | 1602 | 2166 | 24.327939 | 2.35e-04 | 1.81 | 3.47e+06 | 24.328017 | 1.35 |

Table 4.21: Numerical values at strike price for short position of Stock Borrowing Fee pricing problem with American exercise rights with linear boundary conditions, solved with policy iteration (Algorithm 1). Nonuniform discretization grid and uniform timesteps used

| Nodes | Tstep | Iter | Value | Change | Rate | Total Cost | Richardson | avg iter |
|---|---|---|---|---|---|---|---|---|
| 51 | 52 | 68 | 24.271363 | — | — | 3.40e+03 | — | 1.31 |
| 101 | 102 | 137 | 24.312489 | 4.11e-02 | — | 1.37e+04 | 24.326198 | 1.34 |
| 201 | 202 | 273 | 24.323818 | 1.13e-02 | 1.86 | 5.46e+04 | 24.327594 | 1.35 |
| 401 | 402 | 545 | 24.326851 | 3.03e-03 | 1.90 | 2.18e+05 | 24.327861 | 1.36 |
| 801 | 802 | 1091 | 24.327690 | 8.39e-04 | 1.85 | 8.73e+05 | 24.327970 | 1.36 |
| 1601 | 1602 | 2171 | 24.327933 | 2.43e-04 | 1.79 | 3.47e+06 | 24.328014 | 1.36 |

Table 4.22: Numerical values at strike price for short position of Stock Borrowing Fee pricing problem with American exercise rights with linear boundary conditions, solved with penalty iteration (Algorithm 3). Nonuniform discretization grid and uniform timesteps used

| Nodes | Tstep | Iter | Value | Change | Rate | Total Cost | Richardson | avg iter |
|---|---|---|---|---|---|---|---|---|
| 51 | 30 | 44 | 24.294097 | — | — | 2.20e+03 | — | 1.47 |
| 101 | 51 | 81 | 24.320602 | 2.65e-02 | — | 8.10e+03 | 24.329436 | 1.59 |
| 201 | 93 | 154 | 24.326520 | 5.92e-03 | 2.16 | 3.08e+04 | 24.328493 | 1.66 |
| 401 | 175 | 293 | 24.327677 | 1.16e-03 | 2.36 | 1.17e+05 | 24.328063 | 1.67 |
| 801 | 340 | 576 | 24.327961 | 2.84e-04 | 2.03 | 4.61e+05 | 24.328055 | 1.69 |
| 1601 | 672 | 1147 | 24.328024 | 6.33e-05 | 2.16 | 1.84e+06 | 24.328045 | 1.71 |

Table 4.23: Numerical values at strike price for short position of Stock Borrowing Fee pricing problem with American exercise rights with linear boundary conditions, solved with penalty iteration (Algorithm 3). Nonuniform discretization grid and variable timesteps used

The same conclusions can be drawn from the American options: the variable timesteps are superior in every meaningful aspect.

| Nodes | Tstep | Iter | Value | Change | Rate | Total Cost | Richardson | avg iter |
|---|---|---|---|---|---|---|---|---|
| 51 | 52 | 67 | 23.027313 | — | — | 3.35e+03 | — | 1.29 |
| 101 | 102 | 138 | 23.068124 | 4.08e-02 | — | 1.38e+04 | 23.081728 | 1.35 |
| 201 | 202 | 269 | 23.079505 | 1.14e-02 | 1.84 | 5.38e+04 | 23.083298 | 1.33 |
| 401 | 402 | 538 | 23.082630 | 3.13e-03 | 1.86 | 2.15e+05 | 23.083672 | 1.34 |
| 801 | 802 | 1080 | 23.083532 | 9.02e-04 | 1.79 | 8.64e+05 | 23.083832 | 1.35 |
| 1601 | 1602 | 2166 | 23.083804 | 2.72e-04 | 1.73 | 3.47e+06 | 23.083895 | 1.35 |

Table 4.24: Numerical values at strike price for long position of Stock Borrowing Fee pricing problem with American exercise rights with linear boundary conditions, solved with policy iteration (Algorithm 1). Nonuniform discretization grid and uniform timesteps used

Although there is no proof that the double penalty method for the long position converges, we have not had computational difficulties. In Tables 4.24 and 4.25 we note a small deterioration of the ratio of convergence as the discretization becomes finer. This problem was noted in [16] and was resolved by a variable timestep selector. We use a similar one here and stable 2nd order of convergence is restored as shown in Table 4.26.

| Nodes | Tstep | Iter | Value | Change | Rate | Total Cost | Richardson | avg iter |
|---|---|---|---|---|---|---|---|---|
| 51 | 52 | 67 | 23.028663 | — | — | 3.35e+03 | — | 1.29 |
| 101 | 102 | 136 | 23.068281 | 3.96e-02 | — | 1.36e+04 | 23.081487 | 1.33 |
| 201 | 202 | 268 | 23.079231 | 1.10e-02 | 1.86 | 5.36e+04 | 23.082881 | 1.33 |
| 401 | 402 | 533 | 23.082250 | 3.02e-03 | 1.86 | 2.13e+05 | 23.083256 | 1.33 |
| 801 | 802 | 1058 | 23.083114 | 8.64e-04 | 1.80 | 8.46e+05 | 23.083402 | 1.32 |
| 1601 | 1602 | 2126 | 23.083378 | 2.63e-04 | 1.71 | 3.40e+06 | 23.083465 | 1.33 |

Table 4.25: Numerical values at strike price for long position of Stock Borrowing Fee pricing problem with American exercise rights with linear boundary conditions, solved with penalty iteration (Algorithm 3). Nonuniform discretization grid and uniform timesteps used

| Nodes | Tstep | Iter | Value | Change | Rate | Total Cost | Richardson | avg iter |
|---|---|---|---|---|---|---|---|---|
| 51 | 30 | 43 | 23.050148 | — | — | 2.15e+03 | — | 1.43 |
| 101 | 51 | 80 | 23.076855 | 2.67e-02 | — | 8.00e+03 | 23.085758 | 1.57 |
| 201 | 92 | 148 | 23.082332 | 5.48e-03 | 2.29 | 2.96e+04 | 23.084157 | 1.61 |
| 401 | 173 | 286 | 23.083229 | 8.97e-04 | 2.61 | 1.14e+05 | 23.083528 | 1.65 |
| 801 | 336 | 561 | 23.083448 | 2.20e-04 | 2.03 | 4.49e+05 | 23.083521 | 1.67 |
| 1601 | 664 | 1115 | 23.083495 | 4.68e-05 | 2.23 | 1.78e+06 | 23.083511 | 1.68 |

Table 4.26: Numerical values at strike price for long position of Stock Borrowing Fee pricing problem with American exercise rights with linear boundary conditions, solved with penalty iteration (Algorithm 3). Nonuniform discretization grid and variable timesteps used

| Nodes | Tstep | Iter | Value | Change | Rate | Total Cost | Richardson | avg iter |
|---|---|---|---|---|---|---|---|---|
| 51 | 52 | 64 | 24.271363 | 0.00e+00 | 0.00 | 3.20e+03 | 0.000000 | 1.23 |
| 101 | 102 | 128 | 24.312474 | 4.11e-02 | 0.00 | 1.28e+04 | 24.326178 | 1.25 |
| 201 | 202 | 256 | 24.323815 | 1.13e-02 | 1.86 | 5.12e+04 | 24.327595 | 1.27 |
| 401 | 402 | 502 | 24.326848 | 3.03e-03 | 1.90 | 2.01e+05 | 24.327859 | 1.25 |
| 801 | 802 | 1001 | 24.327690 | 8.41e-04 | 1.85 | 8.01e+05 | 24.327970 | 1.25 |
| 1601 | 1602 | 2019 | 24.327933 | 2.43e-04 | 1.79 | 3.23e+06 | 24.328014 | 1.26 |

Table 4.27: Numerical values at strike price for short position of Stock Borrowing Fee pricing problem with American exercise rights with linear boundary conditions, solved with improved policy iteration (Algorithm 5). Nonuniform discretization grid and uniform timesteps used

| Nodes | Tstep | Iter | Value | Change | Rate | Total Cost | Richardson | avg iter |
|---|---|---|---|---|---|---|---|---|
| 51 | 52 | 67 | 24.282741 | 0.00e+00 | 0.00 | 3.35e+03 | 0.000000 | 1.29 |
| 101 | 93 | 124 | 24.316341 | 3.36e-02 | 0.00 | 1.24e+04 | 24.327541 | 1.33 |
| 201 | 176 | 239 | 24.325161 | 8.82e-03 | 1.93 | 4.78e+04 | 24.328101 | 1.36 |
| 401 | 342 | 463 | 24.327331 | 2.17e-03 | 2.02 | 1.85e+05 | 24.328055 | 1.35 |
| 801 | 674 | 921 | 24.327867 | 5.35e-04 | 2.02 | 7.37e+05 | 24.328045 | 1.37 |
| 1601 | 1339 | 1809 | 24.328000 | 1.33e-04 | 2.01 | 2.89e+06 | 24.328044 | 1.35 |

Table 4.28: Numerical values at strike price for short position of Stock Borrowing Fee pricing problem with American exercise rights with linear boundary conditions, solved with improved policy iteration (Algorithm 5). Nonuniform discretization grid and variable timesteps used

| Nodes | Tstep | Iter | Value | Change | Rate | Total Cost | Richardson | avg iter |
|---|---|---|---|---|---|---|---|---|
| 51 | 52 | 66 | 23.029142 | 0.00e+00 | 0.00 | 3.30e+03 | 0.000000 | 1.27 |
| 101 | 102 | 131 | 23.068723 | 3.96e-02 | 0.00 | 1.31e+04 | 23.081917 | 1.28 |
| 201 | 202 | 259 | 23.079676 | 1.10e-02 | 1.85 | 5.18e+04 | 23.083326 | 1.28 |
| 401 | 402 | 523 | 23.082678 | 3.00e-03 | 1.87 | 2.09e+05 | 23.083679 | 1.30 |
| 801 | 802 | 1034 | 23.083542 | 8.64e-04 | 1.80 | 8.27e+05 | 23.083830 | 1.29 |
| 1601 | 1602 | 2073 | 23.083805 | 2.63e-04 | 1.72 | 3.32e+06 | 23.083892 | 1.29 |

Table 4.29: Numerical values at strike price for long position of Stock Borrowing Fee pricing problem with American exercise rights with linear boundary conditions, solved with improved policy iteration (Algorithm 5). Nonuniform discretization grid and uniform timesteps used

| Nodes | Tstep | Iter | Value | Change | Rate | Total Cost | Richardson | avg iter |
|---|---|---|---|---|---|---|---|---|
| 51 | 51 | 66 | 23.041036 | 0.00e+00 | 0.00 | 3.30e+03 | 0.000000 | 1.29 |
| 101 | 92 | 123 | 23.073287 | 3.23e-02 | 0.00 | 1.23e+04 | 23.084038 | 1.34 |
| 201 | 174 | 238 | 23.081331 | 8.04e-03 | 2.00 | 4.76e+04 | 23.084012 | 1.37 |
| 401 | 338 | 471 | 23.083296 | 1.96e-03 | 2.03 | 1.88e+05 | 23.083950 | 1.39 |
| 801 | 666 | 937 | 23.083778 | 4.83e-04 | 2.03 | 7.50e+05 | 23.083939 | 1.41 |
| 1601 | 1323 | 1886 | 23.083897 | 1.18e-04 | 2.03 | 3.02e+06 | 23.083936 | 1.43 |

Table 4.30: Numerical values at strike price for long position of Stock Borrowing Fee pricing problem with American exercise rights with linear boundary conditions, solved with improved policy iteration (Algorithm 5). Nonuniform discretization grid and variable timesteps used

### 4.2.6 Uncertain Volatility Model

The parameters for the Uncertain Volatility problem are in Table 4.31

| Description | Symbol | Value |
|---|---|---|
| Maturity time | $T$ | 1/2 |
| Strike Prices | $K_1 = K - a$ | 95 |
| | $K_2 = K$ | 100 |
| | $K_3 = K + a$ | 105 |
| Space Truncation boundary | $S_{\max}$ | 1000 |
| Lower bound of Volatility | $\sigma_{\min}$ | 0.30 |
| Upper bound of Volatility | $\sigma_{\max}$ | 0.45 |
| Interest rate | $r$ | 0.04 |

Table 4.31: Numerical values of parameters used in Uncertain Volatility problem

Although it would be nice to have a suitable nonuniform grid, unfortunately the grid in [11] cannot ensure that there are gridpoints at the three strike prices, and thus cannot promise smooth convergence. Therefore, we will use a uniform grid, and point out that a suitable nonuniform grid will be of interest in future work.

For the results, we start with $N = 1000$ to ensure that there exists a gridpoint on the three strike prices $K_1, K_2, K_3$, because of the discontinuous derivative of the payoff function at those points. This leads to desirable convergence properties.

Additionally, we have tested that the penalty iterates are monotone for both the short and long options with assertion statements in the program, so if the condition is violated the program will halt with error.

The results for the best case solved with policy iteration and penalty iteration are in Tables 4.32 and 4.33, and the results for the worst case are in Tables 4.34 and 4.35.

| Nodes | Tstep | Iter | Value | Change | Rate | Total Cost | Richardson | avg iter |
|---|---|---|---|---|---|---|---|---|
| 101 | 52 | 59 | 0.885623 | — | — | 5.90e+03 | — | 1.13 |
| 201 | 102 | 116 | 0.809679 | -7.59e-02 | — | 2.32e+04 | 0.784365 | 1.14 |
| 401 | 202 | 232 | 0.803512 | -6.17e-03 | 3.62 | 9.28e+04 | 0.801457 | 1.15 |
| 801 | 402 | 461 | 0.802303 | -1.21e-03 | 2.35 | 3.69e+05 | 0.801900 | 1.15 |
| 1601 | 802 | 927 | 0.801841 | -4.62e-04 | 1.39 | 1.48e+06 | 0.801687 | 1.16 |
| 3201 | 1602 | 1865 | 0.801736 | -1.05e-04 | 2.14 | 5.97e+06 | 0.801701 | 1.16 |

Table 4.32: Numerical values at strike price for best case of Uncertain Volatility problem with linear boundary conditions, solved with policy iteration (Algorithm 1). Uniform discretization grid used

| Nodes | Tstep | Iter | Value | Change | Rate | Total Cost | Richardson | avg iter |
|---|---|---|---|---|---|---|---|---|
| 101 | 52 | 62 | 0.113637 | — | — | 6.20e+03 | — | 1.19 |
| 201 | 102 | 120 | 0.120442 | 6.80e-03 | — | 2.40e+04 | 0.122711 | 1.18 |
| 401 | 202 | 240 | 0.125163 | 4.72e-03 | 0.53 | 9.60e+04 | 0.126737 | 1.19 |
| 801 | 402 | 478 | 0.125636 | 4.73e-04 | 3.32 | 3.82e+05 | 0.125794 | 1.19 |
| 1601 | 802 | 958 | 0.125804 | 1.68e-04 | 1.49 | 1.53e+06 | 0.125860 | 1.19 |
| 3201 | 1602 | 1920 | 0.125840 | 3.54e-05 | 2.25 | 6.14e+06 | 0.125851 | 1.20 |

Table 4.33: Numerical values at strike price for best case of Uncertain Volatility problem with linear boundary conditions, solved with penalty iteration (Algorithm 3). Uniform discretization grid used

| Nodes | Tstep | Iter | Value | Change | Rate | Total Cost | Richardson | avg iter |
|---|---|---|---|---|---|---|---|---|
| 101 | 52 | 59 | 0.885623 | — | — | 5.90e+03 | — | 1.13 |
| 201 | 102 | 116 | 0.809679 | -7.59e-02 | — | 2.32e+04 | 0.784365 | 1.14 |
| 401 | 202 | 232 | 0.803512 | -6.17e-03 | 3.62 | 9.28e+04 | 0.801457 | 1.15 |
| 801 | 402 | 461 | 0.802303 | -1.21e-03 | 2.35 | 3.69e+05 | 0.801900 | 1.15 |
| 1601 | 802 | 927 | 0.801841 | -4.62e-04 | 1.39 | 1.48e+06 | 0.801687 | 1.16 |
| 3201 | 1602 | 1865 | 0.801736 | -1.05e-04 | 2.14 | 5.97e+06 | 0.801701 | 1.16 |

Table 4.34: Numerical values at strike price for worst case of Uncertain Volatility problem with linear boundary conditions, solved with policy iteration (Algorithm 1). Uniform discretization grid used

| Nodes | Tstep | Iter | Value | Change | Rate | Total Cost | Richardson | avg iter |
|---|---|---|---|---|---|---|---|---|
| 101 | 52 | 63 | 0.113637 | — | — | 6.30e+03 | — | 1.21 |
| 201 | 102 | 121 | 0.120442 | 6.80e-03 | — | 2.42e+04 | 0.122711 | 1.19 |
| 401 | 202 | 240 | 0.125163 | 4.72e-03 | 0.53 | 9.60e+04 | 0.126737 | 1.19 |
| 801 | 402 | 478 | 0.125636 | 4.73e-04 | 3.32 | 3.82e+05 | 0.125794 | 1.19 |
| 1601 | 802 | 958 | 0.125804 | 1.68e-04 | 1.49 | 1.53e+06 | 0.125860 | 1.19 |
| 3201 | 1602 | 1949 | 0.125840 | 3.54e-05 | 2.25 | 6.24e+06 | 0.125851 | 1.22 |

Table 4.35: Numerical values at strike price for worst case of Uncertain Volatility problem with linear boundary conditions, solved with penalty iteration (Algorithm 3). Uniform discretization grid used

The solution looks like Figure 4.10:

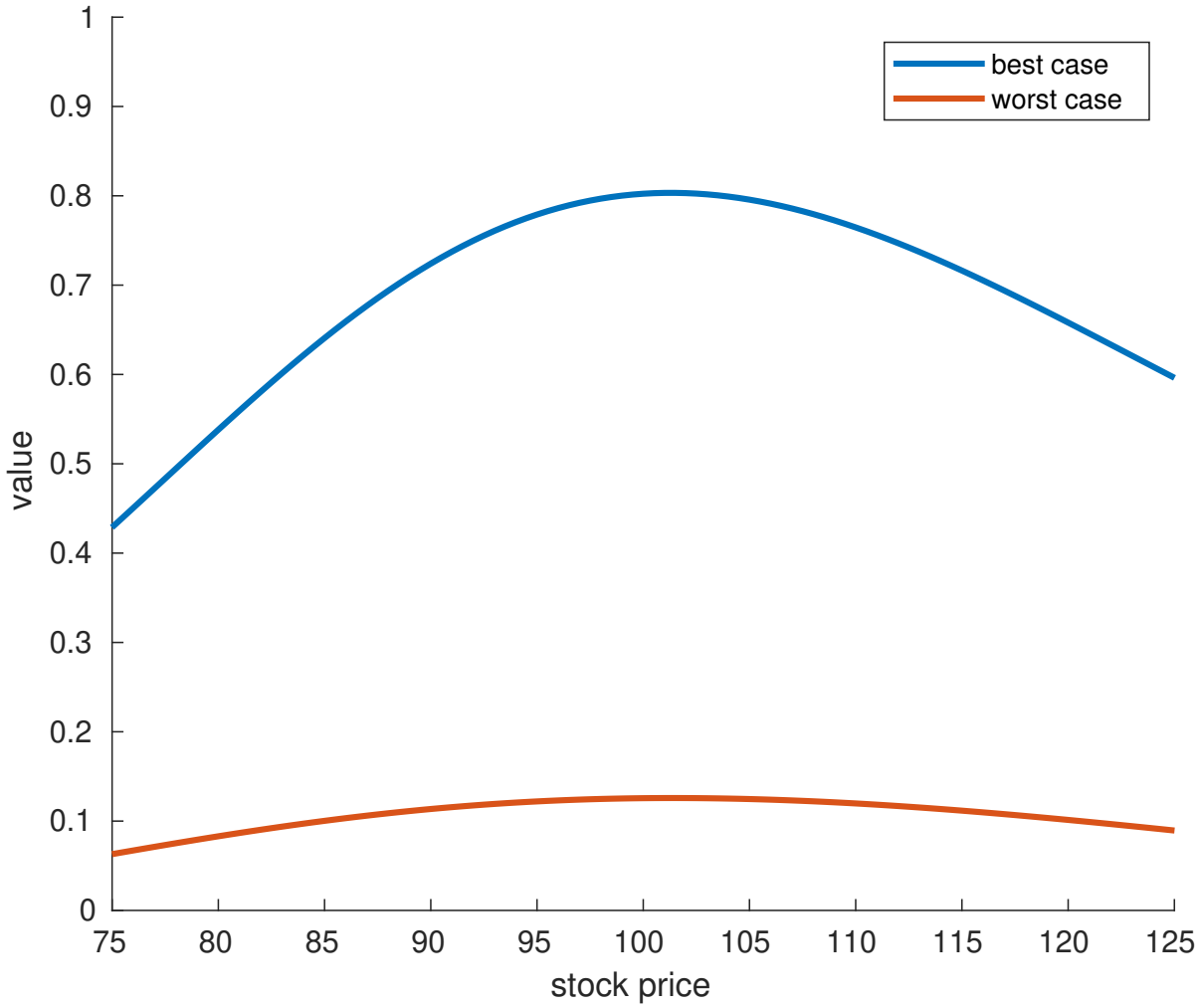


Figure 4.10: Solution of Uncertain Volatility Model with parameters from Table 4.31, calculated with $S_{\max} = 800$ and $N = 1600$

Interestingly, when we do not use Rannacher Smoothing, we are able to get results similar to that found in Figure 5 of [17] in Figure 4.11. Note that, like in [17], we have a "nonsense" solution: the negative value of the solution for the worst case.

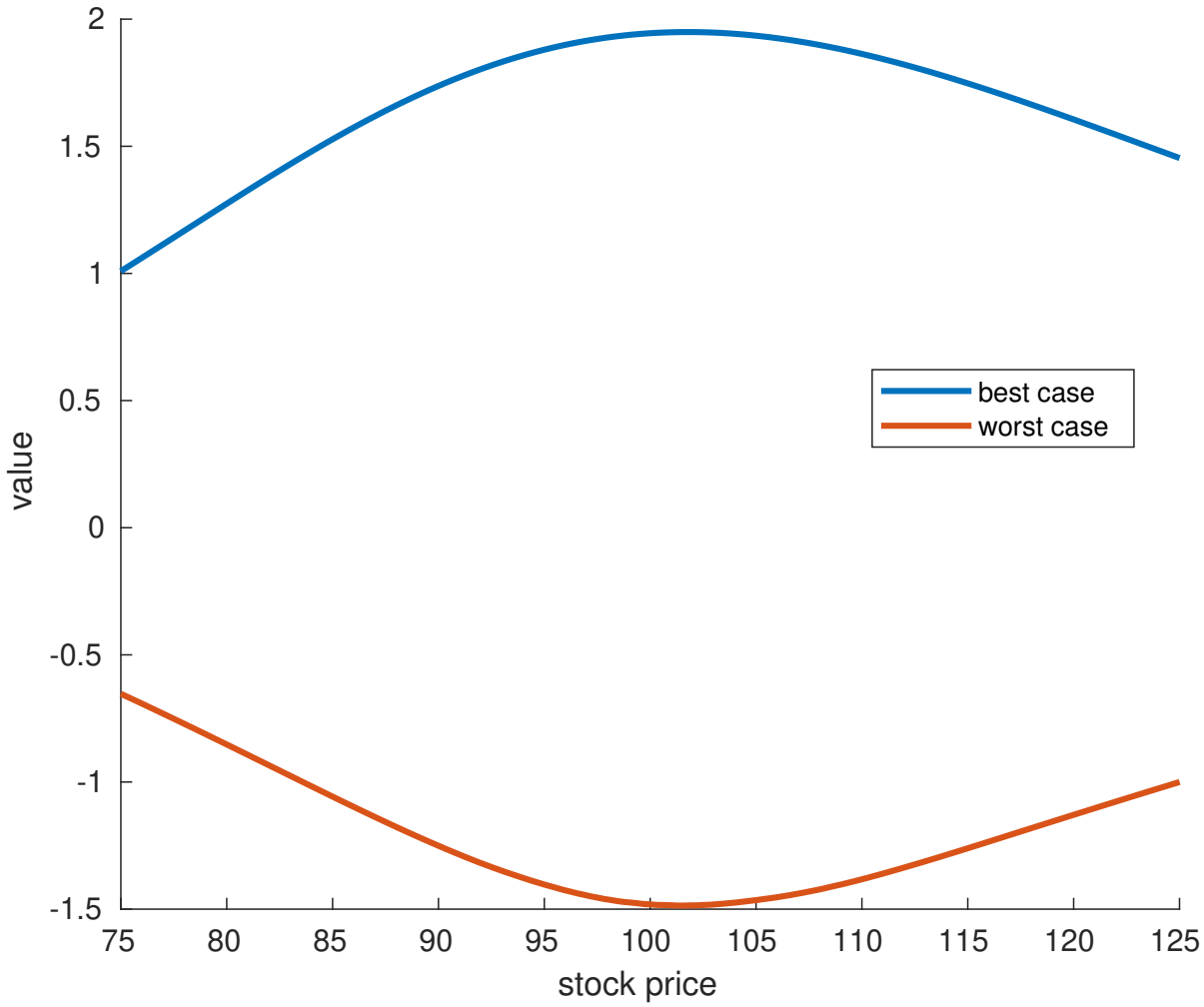


Figure 4.11: Solution of Uncertain Volatility Model, computed with no Rannacher smoothing, calculated with $S_{\max} = 800$ and $N = 1600$

### 4.2.7 Transaction cost models – Put payoff

As described earlier, the results from the transaction cost models differ based on the payoff and based on the type of exercise right given.

Since the put payoff is convex, the model linearizes and is reduced to Equation (2.37) for European options, or Equation (2.38) for American options.

| Nodes | Tstep | Iter | Value | Error | Rate | Total Cost | avg iter |
|---|---|---|---|---|---|---|---|
| 51 | 47 | 61 | 14.420762 | 3.11e-02 | — | 3.05e+03 | 1.30 |
| 101 | 89 | 118 | 14.444983 | 6.92e-03 | 2.17 | 1.18e+04 | 1.33 |
| 201 | 173 | 269 | 14.450173 | 1.73e-03 | 2.00 | 5.38e+04 | 1.55 |
| 401 | 342 | 562 | 14.451472 | 4.33e-04 | 2.00 | 2.25e+05 | 1.64 |
| 801 | 681 | 1254 | 14.451798 | 1.08e-04 | 2.00 | 1.00e+06 | 1.84 |
| 1601 | 1361 | 2652 | 14.451879 | 2.71e-05 | 2.00 | 4.24e+06 | 1.95 |

Table 4.36: European Transaction Model with Put payoff, with values computed at the strike price $K$. Parameters: $\sigma = 1.0$, $r = 0.1$, $T = 0.25$, $\kappa = 0.18$, $K = 100$. Note that the values clearly converge to the exact solution of $14.451906$, matching our expectations.

| Nodes | Tstep | Iter | Value | Change | Rate | Total Cost | Richardson | avg iter |
|---|---|---|---|---|---|---|---|---|
| 51 | 15 | 21 | 14.616135 | 0.00e+00 | 0.00 | 1.05e+03 | 0.000000 | 1.40 |
| 101 | 27 | 40 | 14.660480 | 4.43e-02 | 0.00 | 4.00e+03 | 14.675262 | 1.48 |
| 201 | 52 | 85 | 14.672650 | 1.22e-02 | 1.87 | 1.70e+04 | 14.676707 | 1.63 |
| 401 | 102 | 173 | 14.676668 | 4.02e-03 | 1.60 | 6.92e+04 | 14.678008 | 1.70 |
| 801 | 202 | 359 | 14.678055 | 1.39e-03 | 1.53 | 2.87e+05 | 14.678517 | 1.78 |
| 1601 | 402 | 716 | 14.678536 | 4.81e-04 | 1.53 | 1.15e+06 | 14.678697 | 1.78 |

Table 4.37: American Transaction Model with Put payoff, with values computed at the strike price $K$. Parameters: $\sigma = 1.0$, $r = 0.1$, $T = 0.25$, $\kappa = 0.18$, $K = 100$. Note that the values are very close to those in Table 11.1 of [16], matching our expectations. Constant timesteps used.

| Nodes | Tstep | Iter | Value | Change | Rate | Total Cost | avg iter |
|---|---|---|---|---|---|---|---|
| 51 | 47 | 101 | 14.644326 | — | — | 5.05e+03 | 2.15 |
| 101 | 89 | 200 | 14.671158 | 2.68e-02 | — | 2.00e+04 | 2.25 |
| 201 | 173 | 394 | 14.676948 | 5.79e-03 | 2.21 | 7.88e+04 | 2.28 |
| 401 | 343 | 757 | 14.678397 | 1.45e-03 | 2.00 | 3.03e+05 | 2.21 |
| 801 | 683 | 1389 | 14.678758 | 3.61e-04 | 2.01 | 1.11e+06 | 2.03 |
| 1601 | 1364 | 2744 | 14.678848 | 9.00e-05 | 2.00 | 4.39e+06 | 2.01 |

Table 4.38: American Transaction Model with Put payoff, with values computed at the strike price $K$. Parameters: $\sigma = 1.0$, $r = 0.1$, $T = 0.25$, $\kappa = 0.18$, $K = 100$. Note that the values are very close to those in Table 11.1 of [16], matching our expectations. Also, variable timesteps are used, following the description in [16]

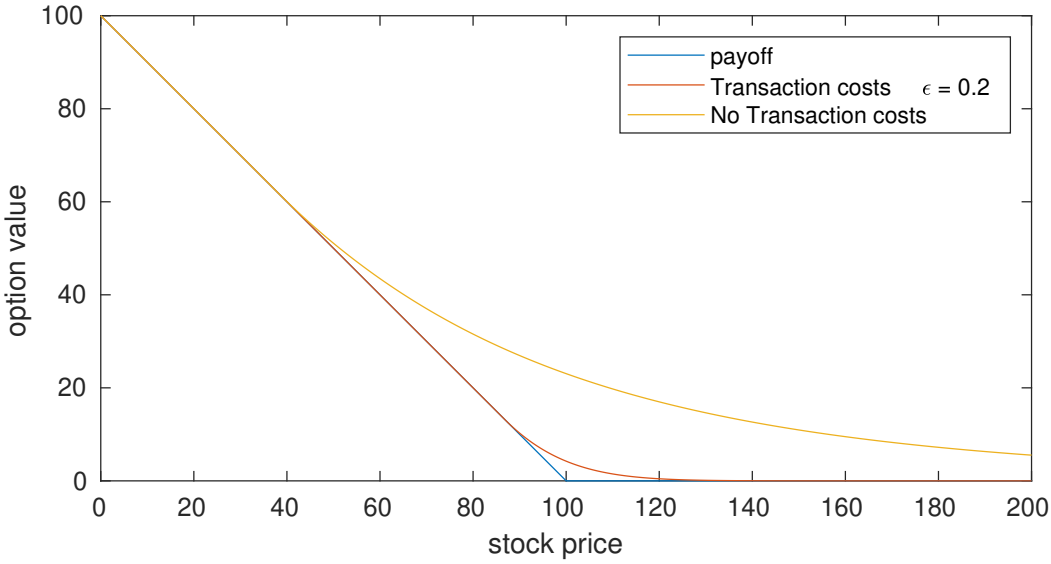


Figure 4.12: American put option with and without transaction cost.

### 4.2.8 Transaction cost models – Butterfly payoff

A butterfly spread payoff shows the nonlinearity in Equations (2.35) and (2.36).

| Nodes | Tstep | Iter | Value | Change | Rate | Total Cost | avg iter |
|---|---|---|---|---|---|---|---|
| 51 | 52 | 66 | 0.110787 | — | — | 3.30e+03 | 1.27 |
| 101 | 102 | 129 | 0.117863 | 7.08e-03 | — | 1.29e+04 | 1.26 |
| 201 | 202 | 254 | 0.124574 | 6.71e-03 | 0.08 | 5.08e+04 | 1.26 |
| 401 | 402 | 507 | 0.125135 | 5.61e-04 | 3.58 | 2.03e+05 | 1.26 |
| 801 | 802 | 1005 | 0.125270 | 1.36e-04 | 2.05 | 8.04e+05 | 1.25 |
| 1601 | 1602 | 2003 | 0.125301 | 3.11e-05 | 2.12 | 3.20e+06 | 1.25 |

Table 4.39: Penalty method for Transaction Cost Model with Butterfly Spread payoff, value computed at the strike price $K$. Parameters: $\sigma = 0.65$, $r = 0.05$, $T = 1$, $\kappa = 0.1$, $K = 100$.

| Nodes | Tstep | Iter | Value | Change | Rate | Total Cost | Richardson | avg iter |
|---|---|---|---|---|---|---|---|---|
| 51 | 52 | 53 | 7.919782 | 0.00e+00 | 0.00 | 2.65e+03 | 0.000000 | 1.02 |
| 101 | 102 | 105 | 7.930563 | 1.08e-02 | 0.00 | 1.05e+04 | 7.934157 | 1.03 |
| 201 | 202 | 208 | 7.933152 | 2.59e-03 | 2.06 | 4.16e+04 | 7.934015 | 1.03 |
| 401 | 402 | 417 | 7.933792 | 6.39e-04 | 2.02 | 1.67e+05 | 7.934005 | 1.04 |
| 801 | 802 | 832 | 7.933949 | 1.58e-04 | 2.02 | 6.66e+05 | 7.934002 | 1.04 |
| 1601 | 1602 | 1666 | 7.933986 | 3.72e-05 | 2.08 | 2.67e+06 | 7.933999 | 1.04 |

Table 4.40: Penalty method for American Transaction Cost Model with Butterfly Spread payoff, value computed at $0.9K$. Parameters: $\sigma = 0.65$, $r = 0.05$, $T = 1$, $\kappa = 0.1$, $K = 100$.

| Nodes | Tstep | Iter | Value | Change | Rate | Total Cost | Richardson | avg iter |
|---|---|---|---|---|---|---|---|---|
| 51 | 52 | 53 | 8.548380 | 0.00e+00 | 0.00 | 2.65e+03 | 0.000000 | 1.02 |
| 101 | 102 | 105 | 8.556495 | 8.11e-03 | 0.00 | 1.05e+04 | 8.559199 | 1.03 |
| 201 | 202 | 208 | 8.558486 | 1.99e-03 | 2.03 | 4.16e+04 | 8.559149 | 1.03 |
| 401 | 402 | 417 | 8.558984 | 4.99e-04 | 2.00 | 1.67e+05 | 8.559150 | 1.04 |
| 801 | 802 | 832 | 8.559112 | 1.28e-04 | 1.96 | 6.66e+05 | 8.559155 | 1.04 |
| 1601 | 1602 | 1666 | 8.559146 | 3.43e-05 | 1.90 | 2.67e+06 | 8.559158 | 1.04 |

Table 4.41: Penalty method for American Transaction Cost Model with Butterfly Spread payoff, value computed at $1.1K$. Parameters: $\sigma = 0.65$, $r = 0.05$, $T = 1$, $\kappa = 0.1$, $K = 100$.

We do not compute the convergence rate for American transaction cost model at the center strike price, because the constraint from the American early exercise right causes the price to stay at approximately the same value. Instead, we compute the convergence rate at the two other strike prices, shown in Tables 4.40 and 4.41.

Although different behaviour at the central strike price is shown in [37], we have reason to believe in the correctness of our algorithm. The convergence rate is very clearly 2.0 as seen in Tables 4.40, 4.41; in addition, [37] uses a Padé scheme [23] which imposes certain smoothness requirements on the solution that do not hold for American options. For one very well-known example, the solution of the American Put option has a discontinuous second derivative [16], which makes precise computation of a solution using a Padé scheme impossible. In addition, our results are supported by analysis in [21], indicating that the solution at $S = K$ stays constant at $a$ for Butterfly spread payoffs.

It is also possible to implement variable timesteps [16], to fix the very small decrease of quality in the convergence rate at $1.1K$. Results are included in Tables 4.42 and 4.43

| Nodes | Tstep | Iter | Value | Change | Rate | Total Cost | Richardson | avg iter |
|---|---|---|---|---|---|---|---|---|
| 51 | 44 | 45 | 7.920952 | 0.00e+00 | 0.00 | 2.25e+03 | 0.000000 | 1.02 |
| 101 | 88 | 93 | 7.930826 | 9.87e-03 | 0.00 | 9.30e+03 | 7.934118 | 1.06 |
| 201 | 177 | 188 | 7.933215 | 2.39e-03 | 2.05 | 3.76e+04 | 7.934012 | 1.06 |
| 401 | 353 | 390 | 7.933802 | 5.87e-04 | 2.03 | 1.56e+05 | 7.933998 | 1.10 |
| 801 | 703 | 818 | 7.933946 | 1.44e-04 | 2.02 | 6.54e+05 | 7.933995 | 1.16 |
| 1601 | 1402 | 1680 | 7.933980 | 3.41e-05 | 2.08 | 2.69e+06 | 7.933992 | 1.20 |

Table 4.42: Penalty method for American Transaction Cost Model with Butterfly Spread payoff, value computed at $0.9K$. Parameters: $\sigma = 0.65$, $r = 0.05$, $T = 1$, $\kappa = 0.1$, $K = 100$. Variable timestepping used.

| Nodes | Tstep | Iter | Value | Change | Rate | Total Cost | Richardson | avg iter |
|---|---|---|---|---|---|---|---|---|
| 51 | 44 | 45 | 8.549498 | 0.00e+00 | 0.00 | 2.25e+03 | 0.000000 | 1.02 |
| 101 | 88 | 93 | 8.556754 | 7.26e-03 | 0.00 | 9.30e+03 | 8.559173 | 1.06 |
| 201 | 177 | 188 | 8.558555 | 1.80e-03 | 2.01 | 3.76e+04 | 8.559155 | 1.06 |
| 401 | 353 | 390 | 8.559003 | 4.48e-04 | 2.01 | 1.56e+05 | 8.559152 | 1.10 |
| 801 | 703 | 818 | 8.559115 | 1.13e-04 | 1.99 | 6.54e+05 | 8.559153 | 1.16 |
| 1601 | 1402 | 1680 | 8.559144 | 2.84e-05 | 1.99 | 2.69e+06 | 8.559153 | 1.20 |

Table 4.43: Penalty method for American Transaction Cost Model with Butterfly Spread payoff, value computed at $1.1K$. Parameters: $\sigma = 0.65$, $r = 0.05$, $T = 1$, $\kappa = 0.1$, $K = 100$. Variable timestepping used.

The above was for penalty iteration algorithms. We present the results for Algorithm 5 in Tables 4.44, 4.45, 4.46, 4.47 for constant and variable timesteps with the solution computed at $0.9K$ and $1.1K$.

| Nodes | Tstep | Iter | Value | Change | Rate | Total Cost | Richardson | avg iter |
|---|---|---|---|---|---|---|---|---|
| 101 | 102 | 103 | 7.919891 | 0.00e+00 | 0.00 | 1.03e+04 | 0.000000 | 1.01 |
| 201 | 202 | 205 | 7.930590 | 1.07e-02 | 0.00 | 4.10e+04 | 7.934157 | 1.01 |
| 401 | 402 | 408 | 7.933160 | 2.57e-03 | 2.06 | 1.63e+05 | 7.934016 | 1.01 |
| 801 | 802 | 816 | 7.933794 | 6.34e-04 | 2.02 | 6.53e+05 | 7.934005 | 1.02 |
| 1601 | 1602 | 1629 | 7.933950 | 1.56e-04 | 2.02 | 2.61e+06 | 7.934003 | 1.02 |

Table 4.44: Policy Iteration (Algorithm 5) for American Transaction Cost Model with Butterfly Spread payoff, value computed at $0.9K$. Parameters: $\sigma = 0.65$, $r = 0.05$, $T = 1$, $\kappa = 0.1$, $K = 100$. Constant timesteps used

| Nodes | Tstep | Iter | Value | Change | Rate | Total Cost | Richardson | avg iter |
|---|---|---|---|---|---|---|---|---|
| 101 | 102 | 103 | 8.548485 | 0.00e+00 | 0.00 | 1.03e+04 | 0.000000 | 1.01 |
| 201 | 202 | 205 | 8.556521 | 8.04e-03 | 0.00 | 4.10e+04 | 8.559200 | 1.01 |
| 401 | 402 | 408 | 8.558495 | 1.97e-03 | 2.03 | 1.63e+05 | 8.559153 | 1.01 |
| 801 | 802 | 816 | 8.558990 | 4.95e-04 | 2.00 | 6.53e+05 | 8.559155 | 1.02 |
| 1601 | 1602 | 1629 | 8.559117 | 1.26e-04 | 1.97 | 2.61e+06 | 8.559159 | 1.02 |

Table 4.45: Policy Iteration (Algorithm 5) for American Transaction Cost Model with Butterfly Spread payoff, value computed at $1.1K$. Parameters: $\sigma = 0.65$, $r = 0.05$, $T = 1$, $\kappa = 0.1$, $K = 100$. Constant timesteps used

| Nodes | Tstep | Iter | Value | Change | Rate | Total Cost | Richardson | avg iter |
|---|---|---|---|---|---|---|---|---|
| 101 | 42 | 43 | 7.920866 | 0.00e+00 | 0.00 | 4.30e+03 | 0.000000 | 1.02 |
| 201 | 87 | 91 | 7.930837 | 9.97e-03 | 0.00 | 1.82e+04 | 7.934160 | 1.05 |
| 401 | 176 | 187 | 7.933215 | 2.38e-03 | 2.07 | 7.48e+04 | 7.934008 | 1.06 |
| 801 | 353 | 373 | 7.933802 | 5.87e-04 | 2.02 | 2.98e+05 | 7.933998 | 1.06 |
| 1601 | 704 | 743 | 7.933946 | 1.44e-04 | 2.02 | 1.19e+06 | 7.933995 | 1.06 |

Table 4.46: Policy Iteration (Algorithm 5) for American Transaction Cost Model with Butterfly Spread payoff, value computed at $0.9K$. Parameters: $\sigma = 0.65$, $r = 0.05$, $T = 1$, $\kappa = 0.1$, $K = 100$. Variable timesteps used

| Nodes | Tstep | Iter | Value | Change | Rate | Total Cost | Richardson | avg iter |
|---|---|---|---|---|---|---|---|---|
| 101 | 42 | 43 | 8.549417 | 0.00e+00 | 0.00 | 4.30e+03 | 0.000000 | 1.02 |
| 201 | 87 | 91 | 8.556764 | 7.35e-03 | 0.00 | 1.82e+04 | 8.559213 | 1.05 |
| 401 | 176 | 187 | 8.558554 | 1.79e-03 | 2.04 | 7.48e+04 | 8.559151 | 1.06 |
| 801 | 353 | 373 | 8.559003 | 4.49e-04 | 2.00 | 2.98e+05 | 8.559152 | 1.06 |
| 1601 | 704 | 743 | 8.559115 | 1.13e-04 | 1.99 | 1.19e+06 | 8.559153 | 1.06 |

Table 4.47: Policy Iteration (Algorithm 5) for American Transaction Cost Model with Butterfly Spread payoff, value computed at $1.1K$. Parameters: $\sigma = 0.65$, $r = 0.05$, $T = 1$, $\kappa = 0.1$, $K = 100$. Variable timesteps used

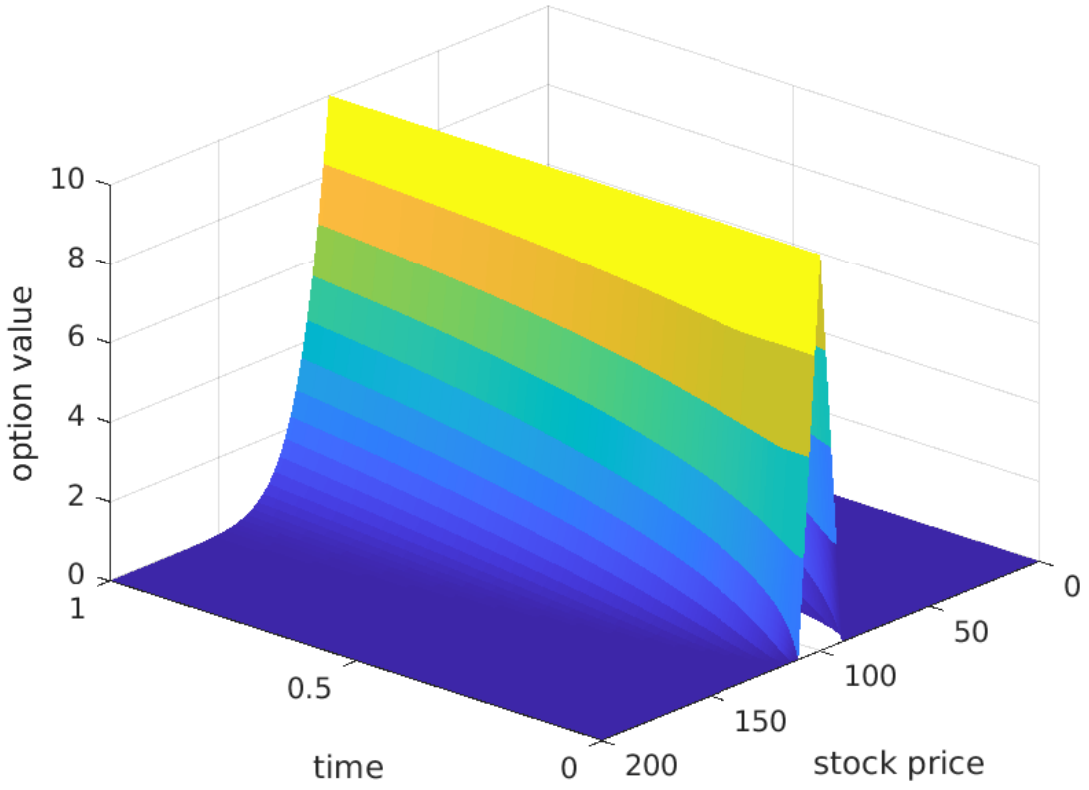


Figure 4.13: Butterfly spread payoff, with transaction cost $\kappa = 0.2$. Colormap: Blue indicates low value (0) and yellow indicates high value (10). Note the cusp at the strike price ($K = 100$), a feature of the American early exercise right.

## 4.3 Solution Agreement

It is important for us to show that both the penalty and policy algorithms agree on the result, and that both types of boundary conditions on the far side of the boundary lead to numerically similar solutions. First, we will show the results for different types of boundary condition, and then we show the results of penalty and policy algorithms.

### 4.3.1 Dirichlet vs Linear Boundary Conditions

Note that the equations here are the same except for the last node. We test with two different types of problems: one where the value at the last node is expected to be almost zero, and one

where it is expected to be clearly positive.

For the case where the value at the last node is zero, we test with European Put option, and for the case where the value at the last node is positive, we test with the Borrow-Lend problem solved with tridiagonal penalty method. We report the maximum difference, which is expected to be very small, over the entire time and space domain.

For European Put option:

```
testing agreement between Dirichlet and linear boundary conditions
European Put: 7.81022e-07
```

For Borrow-Lend Problem:

```
testing agreement between Dirichlet and linear boundary conditions
Borrow-Lend Short: 7.81022e-07
```

### 4.3.2 Policy vs Penalty Methods

Here, due to the differing treatment of the American penalty term, we do not expect that the results for the penalty and policy methods would work the same when American style exercise rights are used, especially since the variable timestepping algorithm used in [16, 9] does not work with the policy iteration algorithm. Nevertheless, we can still test the agreement for those with European-style exercise rights, and we again compute the maximum difference over the entire time and space grid.

Borrow-Lend Problem:

```
Borrow-Lend comparison
Difference between policy and tridiag penalty: 2.715756e-07
Difference between policy and diag penalty: 2.715756e-07
Difference between tridiag penalty and diag penalty: 3.988090e-08
```

Stock Borrowing Fee Problem:

```
Stock Borrowing Fee comparison
max difference between policy and tridiag penalty: 1.692797e-10
```

Uncertain Volatility Problem:

```
Uncertain Volatility comparison
```

```
max difference between policy and tridiag penalty: 1.459388e-13
```

Therefore, as we can see, we have numerical agreement between the penalty and policy algorithms.

## 4.4 Smoothness of Greeks

In addition to the numerical convergence, we also test for the smoothness of the Greeks, which are important for hedging purposes and other practical applications. We have already seen cases where seemingly reasonable solution curves can lead to large oscillations in the plot of the Greeks (Figure 3.1).

A summary of the plots can be seen in Figure 4.14. Of particular interest is the Gamma ($V_{SS}$) for American put options: Without the variable timesteps there are oscillations in the solution (see Figure 4.15). .

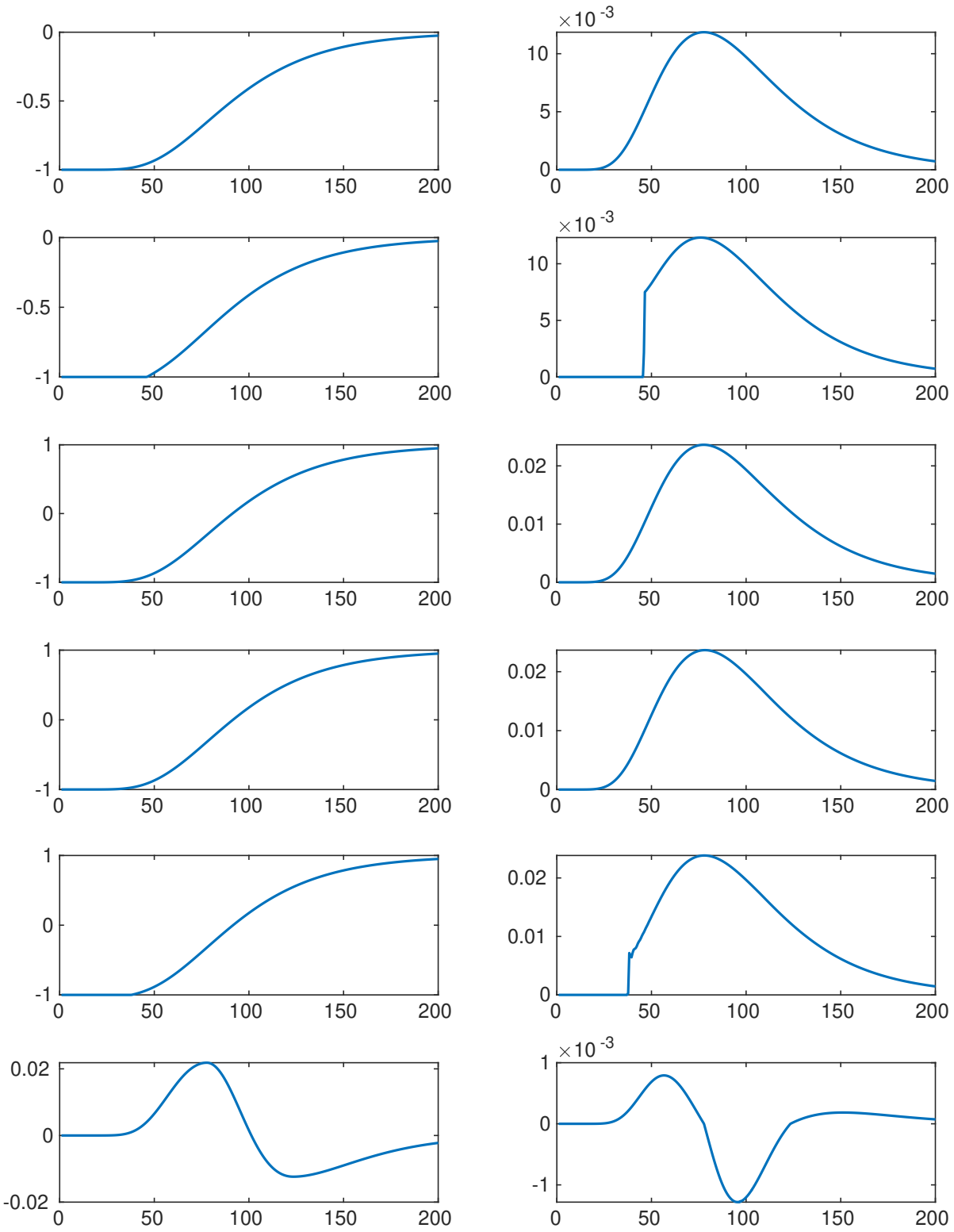


Figure 4.14: Left: Delta; right: Gamma. From top to bottom: European, American, Borrow-Lend, Stock Borrowing Fee, Stock Borrowing Fee (American), Uncertain Volatility problems. As can be seen the Greeks are smooth with the exception of options with American exercise rights, which is a natural feature due to the nonlinear constraint

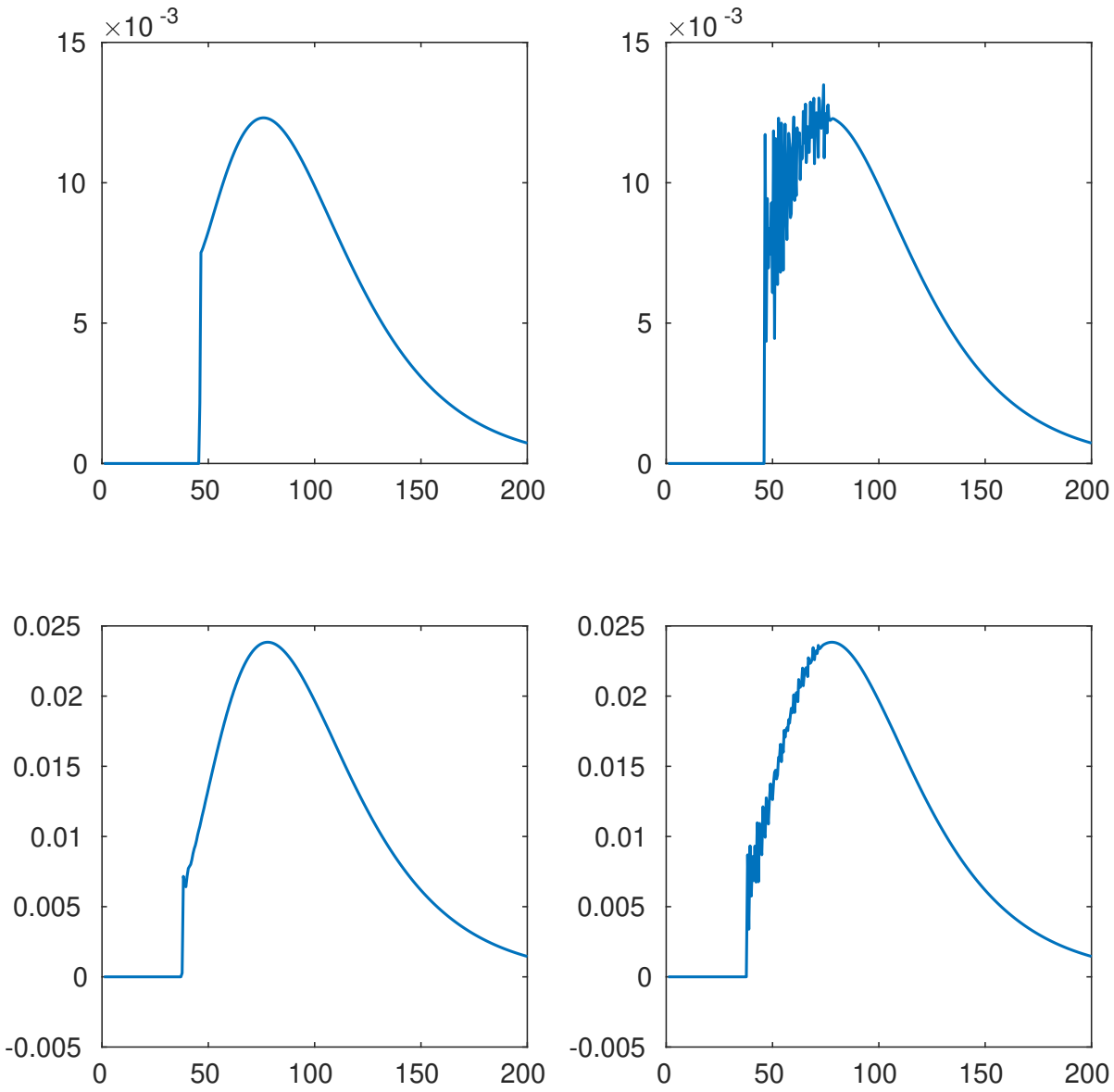


Figure 4.15: Left: Variable timesteps. Right: Constant timesteps. Top: American Put option. Bottom: Stock Borrowing Fee problem with American early exercise right

We also compare with the Greeks computed from policy iteration:

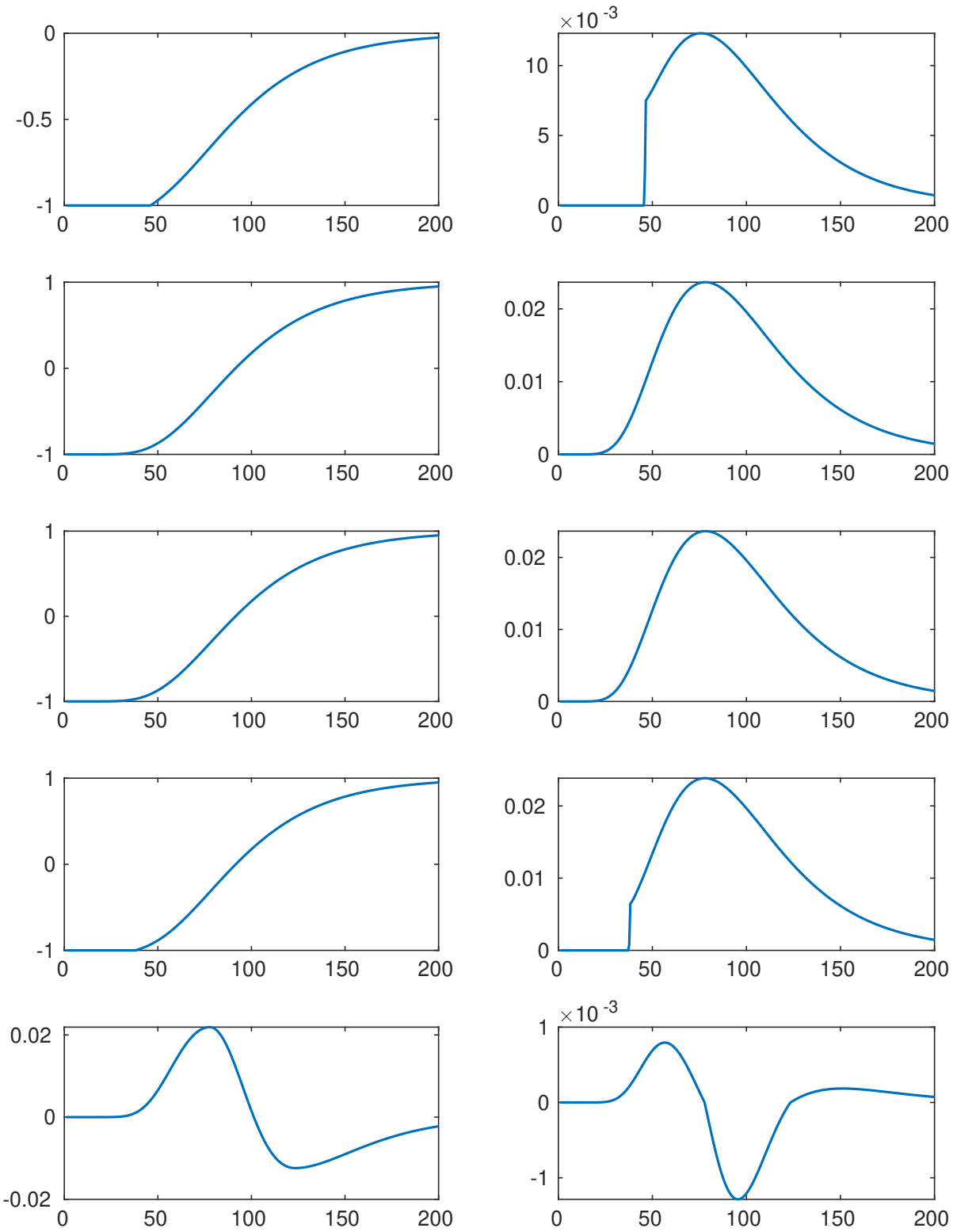


Figure 4.16: Left: Delta; right: Gamma. From top to bottom: American, Borrow-Lend, Stock Borrowing Fee, Stock Borrowing Fee (American), Uncertain Volatility solved with policy iteration

## 4.5 Convergence of Greeks

Note that the successive differences between different levels of discretization can go down to small values such as $10^{-7}$ or even lower. We do not consider the rate of convergence beyond that point as long as the error does not increase dramatically.

### 4.5.1 European Put option

It is also important that the Greeks converge. Unfortunately, if we only use Rannacher Smoothing over one ordinary timestep (corresponding to two Rannacher steps; recall that each Rannacher step is 1/2 that of a normal timestep), as was suggested in the original paper [30], then the Gamma only shows first-order convergence in Table 4.48.

| Nodes | Error (Delta) | Convergence (Delta) | Error (Gamma) | Convergence (Gamma) |
|---|---|---|---|---|
| 51 | 4.18e-04 | — | -2.08e-04 | — |
| 101 | 1.05e-04 | 2.00 | -1.54e-04 | 0.43 |
| 201 | 2.60e-05 | 2.01 | 7.88e-05 | 0.96 |
| 401 | 6.50e-06 | 2.00 | 3.93e-05 | 1.01 |
| 801 | 1.63e-06 | 2.00 | 1.96e-05 | 1.00 |
| 1601 | 4.07e-07 | 2.00 | 9.79e-06 | 1.00 |

Table 4.48: Numerical values of Greeks at strike price for European Put option, computed with Rannacher Smoothing over one timestep

To remedy this, we can have Rannacher smoothing over two or even three timesteps, as seen in Tables 4.49 and 4.50.

| Nodes | Error (Delta) | Convergence (Delta) | Error (Gamma) | Convergence (Gamma) |
|---|---|---|---|---|
| 51 | 4.68e-04 | — | -1.05e-05 | — |
| 101 | 1.17e-04 | 2.00 | -1.32e-06 | 2.98 |
| 201 | 2.93e-05 | 2.00 | -4.20e-07 | 1.65 |
| 401 | 7.34e-06 | 2.00 | -1.01e-07 | 2.05 |
| 801 | 1.83e-06 | 2.00 | -2.48e-08 | 2.03 |
| 1601 | 4.59e-07 | 2.00 | -6.15e-09 | 2.01 |

Table 4.49: Numerical values of Greeks at strike price for European Put option, computed with Rannacher Smoothing over two timesteps

| Nodes | Error (Delta) | Convergence (Delta) | Error (Gamma) | Convergence (Gamma) |
|---|---|---|---|---|
| 51 | 5.22e-04 | — | -2.86e-05 | — |
| 101 | 1.31e-04 | 2.00 | -5.65e-06 | 2.34 |
| 201 | 3.27e-05 | 2.00 | -1.41e-06 | 2.01 |
| 401 | 8.16e-06 | 2.00 | -3.51e-07 | 2.00 |
| 801 | 2.04e-06 | 2.00 | -8.78e-08 | 2.00 |
| 1601 | 5.10e-07 | 2.00 | -2.20e-08 | 2.00 |

Table 4.50: Numerical values of Greeks at strike price for European Put option, computed with Rannacher Smoothing over three timesteps

This is in agreement with the results in [10], which states that taking four Rannacher timesteps instead of two ordinary timesteps is best for convergence.

For the remaining problems, thanks to numerical agreement between the penalty and policy methods for problems with European-style exercise rights, we do not need to show both results. However, since the problems with American-style exercise rights do have differences, we need to show both the penalty and policy methods for the American Put and American Stock Borrowing Fee problems.

### 4.5.2 American Put option

We present the results for the American Put option solved with Policy Iteration, Penalty Iteration, and Penalty Iteration with variable timestepping in Tables 4.51, 4.52, and 4.53 respectively. As can be seen, the penalty iteration with variable timesteps is the only algorithm where the difference is of the same sign over different discretization levels.

| Nodes | Diff (Delta) | Convergence (Delta) | Diff (Gamma) | Convergence (Gamma) |
|---|---|---|---|---|
| 101 | -3.19e-04 | — | -1.39e-05 | — |
| 201 | -8.05e-05 | 1.99 | -9.89e-07 | 3.81 |
| 401 | -2.29e-05 | 1.82 | -1.07e-08 | 6.53 |
| 801 | -6.38e-06 | 1.84 | 1.04e-07 | -3.28 |
| 1601 | -1.82e-06 | 1.81 | 6.60e-08 | 0.66 |

Table 4.51: Numerical values of Greeks at strike pricefor American Put option, computed with policy iteration and constant timesteps

| Nodes | Diff (Delta) | Convergence (Delta) | Diff (Gamma) | Convergence (Gamma) |
|---|---|---|---|---|
| 101 | -3.16e-04 | — | -1.37e-05 | — |
| 201 | -8.02e-05 | 1.98 | -7.43e-07 | 4.20 |
| 401 | -2.28e-05 | 1.82 | 3.60e-08 | 4.37 |
| 801 | -6.42e-06 | 1.83 | 1.33e-07 | -1.88 |
| 1601 | -1.85e-06 | 1.79 | 7.68e-08 | 0.79 |

Table 4.52: Numerical values of Greeks at strike price for American Put option, computed with penalty iteration and constant timesteps

| Nodes | Diff (Delta) | Convergence (Delta) | Diff (Gamma) | Convergence (Gamma) |
|---|---|---|---|---|
| 101 | -3.18e-04 | — | -2.03e-05 | — |
| 201 | -7.23e-05 | 2.14 | -4.20e-06 | 2.28 |
| 401 | -2.01e-05 | 1.85 | -9.86e-07 | 2.09 |
| 801 | -5.23e-06 | 1.94 | -2.22e-07 | 2.15 |
| 1601 | -1.33e-06 | 1.98 | -5.49e-08 | 2.02 |

Table 4.53: Numerical values of Greeks at strike price for American Put option, computed with policy iteration and variable timesteps

### 4.5.3 Borrow Lend

| Nodes | Diff (Delta) | Convergence (Delta) | Diff (Gamma) | Convergence (Gamma) |
|---|---|---|---|---|
| 101 | -6.43e-04 | — | 1.49e-05 | — |
| 201 | -1.65e-04 | 1.97 | 3.79e-07 | 5.30 |
| 401 | -4.10e-05 | 2.01 | 2.73e-07 | 0.48 |
| 801 | -1.02e-05 | 2.01 | 6.49e-08 | 2.07 |
| 1601 | -2.53e-06 | 2.01 | 2.59e-08 | 1.32 |

Table 4.54: Numerical values of Greeks at strike price for Borrow-Lend problem, short position

| Nodes | Diff (Delta) | Convergence (Delta) | Diff (Gamma) | Convergence (Gamma) |
|---|---|---|---|---|
| 101 | -6.56e-04 | — | 1.87e-05 | — |
| 201 | -1.66e-04 | 1.98 | 1.99e-06 | 3.24 |
| 401 | -4.11e-05 | 2.02 | 7.36e-07 | 1.43 |
| 801 | -1.03e-05 | 1.99 | 1.76e-07 | 2.07 |
| 1601 | -2.59e-06 | 2.00 | 3.80e-08 | 2.21 |

Table 4.55: Numerical values of Greeks at strike price for Borrow-Lend problem, long position

### 4.5.4 Stock Borrowing Fees

| Nodes | Diff (Delta) | Convergence (Delta) | Diff (Gamma) | Convergence (Gamma) |
|---|---|---|---|---|
| 101 | -6.35e-04 | — | 1.41e-05 | — |
| 201 | -1.62e-04 | 1.97 | 4.15e-07 | 5.09 |
| 401 | -4.03e-05 | 2.01 | 3.37e-07 | 0.30 |
| 801 | -9.97e-06 | 2.02 | 9.40e-08 | 1.84 |
| 1601 | -2.53e-06 | 1.98 | 7.74e-09 | 3.60 |

Table 4.56: Numerical values of Greeks at strike price for Stock Borrowing Fee problem, short position

| Nodes | Diff (Delta) | Convergence (Delta) | Diff (Gamma) | Convergence (Gamma) |
|---|---|---|---|---|
| 101 | -6.21e-04 | — | 1.71e-05 | — |
| 201 | -1.59e-04 | 1.97 | 1.90e-06 | 3.17 |
| 401 | -3.96e-05 | 2.00 | 5.71e-07 | 1.73 |
| 801 | -9.92e-06 | 2.00 | 1.44e-07 | 1.99 |
| 1601 | -2.48e-06 | 2.00 | 3.33e-08 | 2.11 |

Table 4.57: Numerical values of Greeks at strike price for Stock Borrowing Fee problem, long position

### 4.5.5 American Stock Borrowing Fees

| Nodes | Diff (Delta) | Convergence (Delta) | Diff (Gamma) | Convergence (Gamma) |
|---|---|---|---|---|
| 201 | -9.94e-05 | — | -5.81e-06 | — |
| 401 | -2.41e-05 | 2.04 | -1.10e-06 | 2.40 |
| 801 | -6.25e-06 | 1.95 | -2.18e-07 | 2.33 |
| 1601 | -1.59e-06 | 1.97 | -1.37e-08 | 4.00 |
| 3201 | -4.17e-07 | 1.94 | 1.36e-08 | 0.01 |

Table 4.58: Numerical values of Greeks at strike price for Stock Borrowing Fees problem with American exercise rights (short position), computed with policy iteration and constant timesteps

| Nodes | Diff (Delta) | Convergence (Delta) | Diff (Gamma) | Convergence (Gamma) |
|---|---|---|---|---|
| 201 | -9.61e-05 | — | -5.64e-06 | — |
| 401 | -2.34e-05 | 2.04 | -1.04e-06 | 2.43 |
| 801 | -6.06e-06 | 1.95 | -1.96e-07 | 2.41 |
| 1601 | -1.54e-06 | 1.97 | -3.13e-09 | 5.97 |
| 3201 | -4.04e-07 | 1.93 | 1.79e-08 | -2.52 |

Table 4.59: Numerical values of Greeks at strike price for Stock Borrowing Fees problem with American exercise rights (short position), computed with penalty iteration and constant timesteps

| Nodes | Diff (Delta) | Convergence (Delta) | Diff (Gamma) | Convergence (Gamma) |
|---|---|---|---|---|
| 201 | -3.66e-05 | — | -3.42e-05 | — |
| 401 | -3.67e-06 | 3.32 | -1.06e-05 | 1.69 |
| 801 | -3.77e-07 | 3.28 | -2.83e-06 | 1.91 |
| 1601 | -2.33e-09 | 7.34 | -7.50e-07 | 1.91 |
| 3201 | 1.92e-08 | -3.04 | -1.93e-07 | 1.96 |

Table 4.60: Numerical values of Greeks at strike price for Stock Borrowing Fees problem with American exercise rights (short position), computed with penalty iteration and variable timesteps

| Nodes | Diff (Delta) | Convergence (Delta) | Diff (Gamma) | Convergence (Gamma) |
|---|---|---|---|---|
| 201 | -7.35e-05 | — | -4.44e-06 | — |
| 401 | -1.70e-05 | 2.11 | -1.01e-06 | 2.14 |
| 801 | -3.74e-06 | 2.19 | -1.70e-07 | 2.57 |
| 1601 | -1.01e-06 | 1.88 | 1.70e-08 | 3.33 |
| 3201 | -3.00e-07 | 1.76 | 2.67e-08 | -0.65 |

Table 4.61: Numerical values of Greeks at strike price for Stock Borrowing Fees problem with American exercise rights (long position), computed with policy iteration and constant timesteps

| Nodes | Diff (Delta) | Convergence (Delta) | Diff (Gamma) | Convergence (Gamma) |
|---|---|---|---|---|
| 201 | -1.02e-04 | — | -3.03e-06 | — |
| 401 | -2.63e-05 | 1.95 | -4.88e-07 | 2.63 |
| 801 | -7.04e-06 | 1.90 | 2.13e-08 | 4.52 |
| 1601 | -1.97e-06 | 1.84 | 7.29e-08 | -1.77 |
| 3201 | -5.80e-07 | 1.76 | 4.39e-08 | 0.73 |

Table 4.62: Numerical values of Greeks at strike price for Stock Borrowing Fees problem with American exercise rights (long position), computed with penalty iteration and constant timesteps

| Nodes | Diff (Delta) | Convergence (Delta) | Diff (Gamma) | Convergence (Gamma) |
|---|---|---|---|---|
| 201 | -3.00e-05 | — | -3.20e-05 | — |
| 401 | -2.00e-06 | 3.91 | -1.10e-05 | 1.54 |
| 801 | 7.40e-07 | 1.43 | -2.94e-06 | 1.90 |
| 1601 | 2.95e-07 | 1.33 | -7.90e-07 | 1.90 |
| 3201 | 9.14e-08 | 1.69 | -2.04e-07 | 1.95 |

Table 4.63: Numerical values of Greeks at strike price for Stock Borrowing Fees problem with American exercise rights (long position), computed with penalty iteration and variable timesteps

### 4.5.6 Uncertain Volatility

| Nodes | Diff (Delta) | Convergence (Delta) | Diff (Gamma) | Convergence (Gamma) |
|---|---|---|---|---|
| 201 | 3.50e-04 | — | -1.39e-04 | — |
| 401 | 4.06e-05 | 3.11 | -1.41e-05 | 3.30 |
| 801 | 1.07e-05 | 1.92 | -3.01e-06 | 2.23 |
| 1601 | 3.82e-06 | 1.49 | -9.98e-07 | 1.59 |
| 3201 | 7.69e-07 | 2.31 | -2.32e-07 | 2.11 |

Table 4.64: Numerical values of Greeks at strike price for Uncertain Volatility problem (best case)

| Nodes | Diff (Delta) | Convergence (Delta) | Diff (Gamma) | Convergence (Gamma) |
|---|---|---|---|---|
| 201 | -6.39e-06 | — | 8.83e-06 | — |
| 401 | -5.63e-06 | 0.18 | 6.57e-06 | 0.43 |
| 801 | -1.98e-07 | 4.83 | 6.00e-07 | 3.45 |
| 1601 | -8.33e-08 | 1.25 | 2.22e-07 | 1.43 |
| 3201 | -5.11e-09 | 4.03 | 4.58e-08 | 2.28 |

Table 4.65: Numerical values of Greeks at strike price for Uncertain Volatility problem (worst case)

### 4.5.7 Transaction cost models

| Nodes | Diff (Delta) | Convergence (Delta) | Diff (Gamma) | Convergence (Gamma) |
|---|---|---|---|---|
| 100 | 0.00e+00 | 0.00 | 0.00e+00 | 0.00 |
| 200 | -8.25e-05 | 0.00 | -1.49e-06 | 0.00 |
| 400 | -2.06e-05 | 2.00 | -3.73e-07 | 2.00 |
| 800 | -5.16e-06 | 2.00 | -9.33e-08 | 2.00 |
| 1600 | -1.29e-06 | 2.00 | -2.33e-08 | 2.00 |
| 3200 | -3.22e-07 | 2.00 | -6.00e-09 | 1.96 |

Table 4.66: Numerical values of Greeks at strike price $K = 100$ for European Put with $\sigma = 1.0$, $r = 0.1$, $T = 0.25$, $\kappa = 0.18$

| Nodes | Diff (Delta) | Convergence (Delta) | Diff (Gamma) | Convergence (Gamma) |
|---|---|---|---|---|
| 100 | 0.00e+00 | 0.00 | 0.00e+00 | 0.00 |
| 200 | -4.68e-05 | 0.00 | -4.44e-06 | 0.00 |
| 400 | -1.26e-05 | 1.89 | -1.72e-06 | 1.37 |
| 800 | -3.14e-06 | 2.01 | -5.65e-07 | 1.60 |
| 1600 | -8.15e-07 | 1.95 | -8.86e-09 | 5.99 |
| 3200 | -2.05e-07 | 1.99 | -5.13e-08 | -2.54 |

Table 4.67: Numerical values of Greeks at strike price $K = 100$ for American Put using variable timesteps with $\sigma = 1.0$, $r = 0.1$, $T = 0.25$, $\kappa = 0.18$

| Nodes | Diff (Delta) | Convergence (Delta) | Diff (Gamma) | Convergence (Gamma) |
|---|---|---|---|---|
| 100 | 0.00e+00 | 0.00 | 0.00e+00 | 0.00 |
| 200 | -2.59e-05 | 0.00 | 2.26e-06 | 0.00 |
| 400 | -2.12e-05 | 0.29 | 1.82e-06 | 0.31 |
| 800 | -2.57e-06 | 3.05 | 2.45e-07 | 2.90 |
| 1600 | -8.85e-07 | 1.54 | 8.12e-08 | 1.59 |
| 3200 | -1.90e-07 | 2.22 | 1.77e-08 | 2.20 |

Table 4.68: Numerical values of Greeks at strike price $K = 100$ for European Transaction cost model with butterfly payoff. Parameters: $\sigma = 0.65$, $r = 0.05$, $T = 1$, $\kappa = 0.1$

| Nodes | Diff (Delta) | Convergence (Delta) | Diff (Gamma) | Convergence (Gamma) |
|---|---|---|---|---|
| 100 | 0.00e+00 | 0.00 | 0.00e+00 | 0.00 |
| 200 | 1.36e-04 | 0.00 | 2.40e-05 | 0.00 |
| 400 | 3.51e-06 | 5.28 | 3.18e-06 | 2.91 |
| 800 | 6.50e-06 | -0.89 | 1.70e-06 | 0.91 |
| 1600 | -4.75e-07 | 3.77 | 8.42e-09 | 7.65 |
| 3200 | 6.05e-07 | -0.35 | 1.65e-07 | -4.30 |

Table 4.69: Numerical values of Greeks at strike price $K2 = 90$ for American Transaction cost model using constant timesteps with butterfly payoff. Parameters: $\sigma = 0.65$, $r = 0.05$, $T = 1$, $\kappa = 0.1$

| Nodes | Diff (Delta) | Convergence (Delta) | Diff (Gamma) | Convergence (Gamma) |
|---|---|---|---|---|
| 100 | 0.00e+00 | 0.00 | 0.00e+00 | 0.00 |
| 200 | -4.24e-04 | 0.00 | 1.16e-05 | 0.00 |
| 400 | -1.05e-04 | 2.01 | 2.82e-06 | 2.04 |
| 800 | -2.67e-05 | 1.98 | 7.06e-07 | 2.00 |
| 1600 | -6.92e-06 | 1.95 | 1.80e-07 | 1.97 |
| 3200 | -1.86e-06 | 1.90 | 4.71e-08 | 1.94 |

Table 4.70: Numerical values of Greeks at strike price $K3 = 110$ for American Transaction cost model using constant timesteps with butterfly payoff. Parameters: $\sigma = 0.65$, $r = 0.05$, $T = 1$, $\kappa = 0.1$

| Nodes | Diff (Delta) | Convergence (Delta) | Diff (Gamma) | Convergence (Gamma) |
|---|---|---|---|---|
| 100 | 0.00e+00 | 0.00 | 0.00e+00 | 0.00 |
| 200 | 7.73e-05 | 0.00 | 2.15e-05 | 0.00 |
| 400 | -1.37e-05 | 2.49 | 2.59e-06 | 3.05 |
| 800 | 2.36e-06 | 2.54 | 1.59e-06 | 0.70 |
| 1600 | -1.57e-06 | 0.59 | -3.22e-08 | 5.63 |
| 3200 | 3.43e-07 | 2.19 | 1.58e-07 | -2.29 |

Table 4.71: Numerical values of Greeks at strike price $K2 = 90$ for American Transaction cost model using variable timesteps with butterfly payoff. Parameters: $\sigma = 0.65$, $r = 0.05$, $T = 1$, $\kappa = 0.1$

| Nodes | Diff (Delta) | Convergence (Delta) | Diff (Gamma) | Convergence (Gamma) |
|---|---|---|---|---|
| 100 | 0.00e+00 | 0.00 | 0.00e+00 | 0.00 |
| 200 | -3.66e-04 | 0.00 | 9.63e-06 | 0.00 |
| 400 | -8.93e-05 | 2.04 | 2.31e-06 | 2.06 |
| 800 | -2.26e-05 | 1.98 | 6.00e-07 | 1.94 |
| 1600 | -5.69e-06 | 1.99 | 1.43e-07 | 2.07 |
| 3200 | -1.45e-06 | 1.98 | 3.72e-08 | 1.94 |

Table 4.72: Numerical values of Greeks at strike price $K3 = 110$ for American Transaction cost model using variable timesteps with butterfly payoff. Parameters: $\sigma = 0.65$, $r = 0.05$, $T = 1$, $\kappa = 0.1$

## 4.6 Montonicity tests of Iterates

As stated in the proofs earlier, the iterates are monotone for our penalty methods with the exception of the diagonal penalty method for Borrow-Lend and the double penalty method for Stock Borrowing Fees with American exercise rights and a long position. In our implementations, we use assertion statements and if the condition fails, then the program halts with an error. Since this is never triggered, then we confirmed that the iterates are monotone.

Due to the fact that numerical computation inherently comes with errors, we do not test for monotonicity but for $\epsilon$-monotonicity (see [15]); that is, monotonicity up to a specified tolerance. In our numerical tests we pick $\epsilon = 10^{-8}$, and as mentioned earlier, this condition is never triggered.

Next, we will show some plots which help us understand the nonmonotonicity of the penalty iteration for the Stock Borrowing Fee problem with American exercise rights. We plot the difference between the solution curves for the short and long positions between the first and second iterations, where monotonicity is expected for the short position and not for the long position. Near the strike price, as can be seen in Figure 4.17, the effects of the American penalty matrix $P_A$ can be seen, and show that the iterates are both increasing in this region.

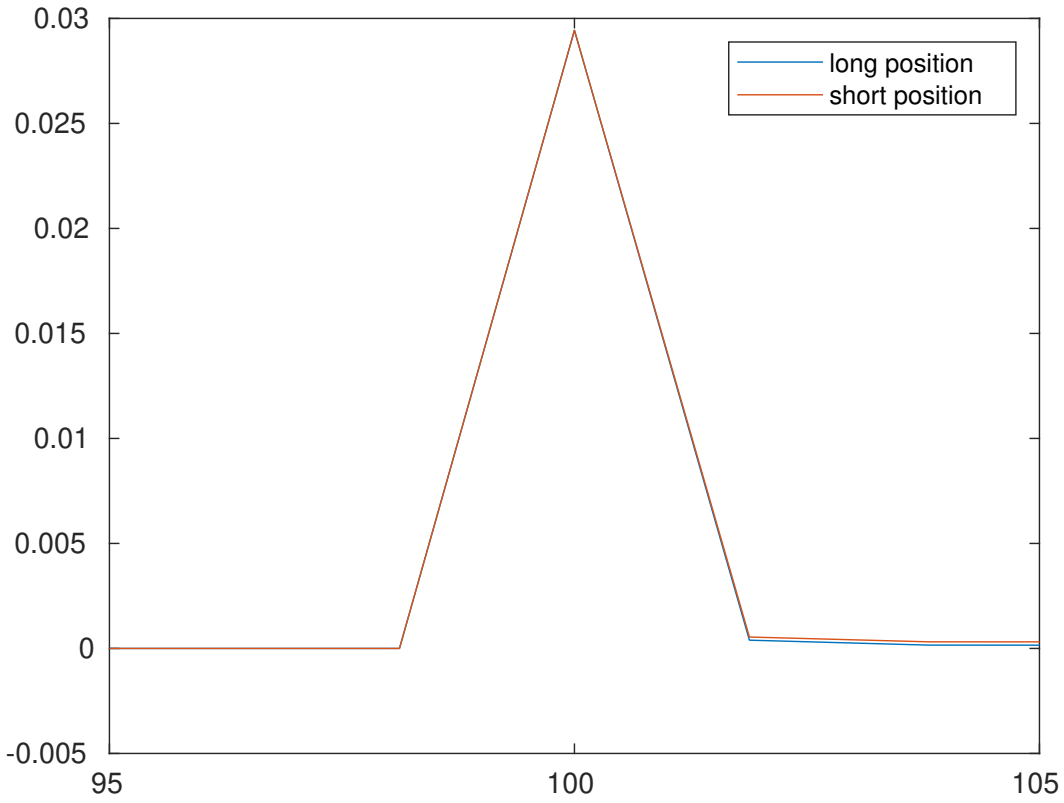


Figure 4.17: Plot of difference of iterates $v^{j,2} - v^{j,1}$ for $j = 2$ to demonstrate monotonicity test for Stock Borrowing Fees with American exercise rights near the strike price

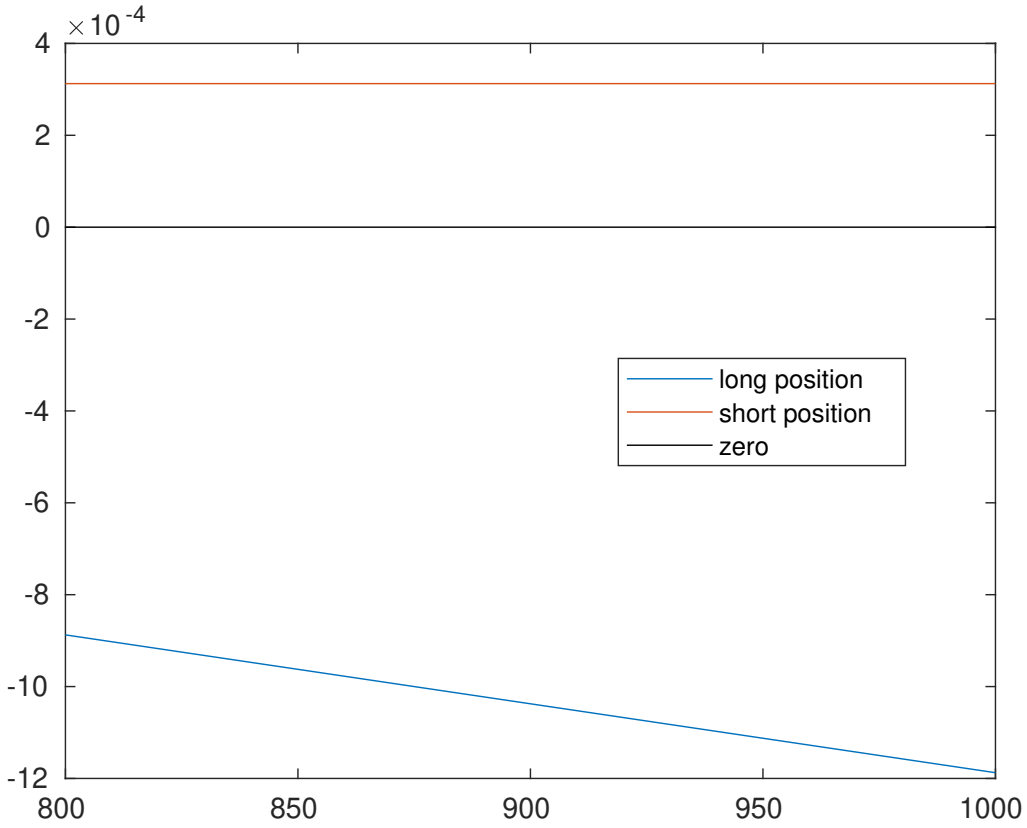


Figure 4.18: Plot of difference of iterates $v^{j,2} - v^{j,1}$ for $j = 2$ to demonstrate monotonicity test for Stock Borrowing Fees with American exercise rights near the endpoint

Near the endpoint in Figure 4.18, the effects of the penalty matrix involving derivatives $P$ can be seen, and hence, the iterates are increasing for the short position and are decreasing for the long position, due to the fact that the matrix from the short position is calculating a maximum and the matrix from the long position is calculating a minimum. Hence, monotonicity doesn't hold for the long position with American options for the entire interval; however, it may hold for certain subdomains of the interval, which explains the good convergence that we nevertheless obtain from our penalty method.

## 4.7 Summary

To summarize the results in this chapter, we list the main points that we found though our numerical experiments:

- For the time and space domain specified and constant timesteps used, a ratio of $4:1$ for the number of nodes to the number of timesteps balances the error in time and space when we have European exercise rights.
- When we have American exercise rights, we need variable timestepping to maintain second-order accuracy of solutions and smoothness of Greeks.
- Both tridiagonal penalty and policy formulations require a comparable number of iterations, and on average between one and two iterations per timestep.
- The diagonal penalty iteration (Algorithm 4) takes far more iterations than the tridiagonal penalty iteration (Algorithm 3).
- We verified numerical agreement between Dirichlet and linear boundary conditions, and also between policy and penalty iteration algorithms where applicable.
- We verified that the iterates are monotone, where applicable.
- The versatility of Algorithm 3 is shown by applying it uniformly to a large set of problems.

# Chapter 5

# Conclusions and Future Work

Many problems in financial option pricing can be written as control problems expressed by HJB or related equations and as nonlinear PDE problems, often with penalty terms. In this thesis, we have considered a number of these problems. For some of them, we developed new penalty iteration algorithms for their penalized PDE formulations. For all the problems considered, we carried a comparison between policy iteration algorithms applied to the HJB formulations and penalty iteration algorithms applied to the penalized PDE formulations.

In contrast to the original paper [14] on the policy iteration algorithms which take a minimum of two iterations before testing for convergence, we have modified the algorithm such that the minimum number of iterations we calculate before testing convergence is one. This is more of a fair comparison, because as we have shown in our results, the average number of iterations per timestep required for convergence is between one and two for the policy iteration algorithms even for the most extreme of the nonlinear problems.

Additionally, we have shown numerical convergence for the second-order Crank-Nicolson timestepping methods for the HJBI equation arising from the long position of Stock Borrowing Fees problem combined with an American early exercise feature and for the Uncertain Volatility model, which were not numerically shown in [14] and [17] respectively.

For the new penalty-like algorithms that we have introduced, we use tridiagonal penalty-like matrices for the discretization of nonlinearities involving partial derivatives for all of the problems that we have examined. As we have seen, they have better convergence properties compared to the diagonal penalty-like matrices. The tridiagonal penalty-like matrices that we use can be combined with penalty matrices such as those used in [16] and for some of the penalty-like algorithms we have proved their monotonic convergence. The monotonically convergent

algorithms are the family of tridiagonal penalty methods for the Borrow Lend, Stock Borrowing Fees, and Uncertain Volatility/Transaction cost models. Moreover, for the Stock Borrowing Fee problem with American exercise rights, we have shown monotonic convergence for the short position. Although the convergence is not monotone for the long position, the penalty iteration algorithm has similar behaviour in terms of number of iterations taken and convergence criteria to that of the policy iteration, and the numerical values are in agreement with the policy iteration.

As mentioned earlier, the motivation for examining the Uncertain Volatility problem is that it is a highly nonlinear problem, with a reasonably large number of iterations taken per timestep. This allows us to more thoroughly test the monotonicity of the iterates, which gives us extra validation in the correctness of our proofs.

When comparing the policy and the penalty iteration methods, we found the number of iterations and accuracy comparable. An important advantage of the penalty iteration methods is that they avoids enumeration of all cases of combinations of control variables to pick set that produces the minimum (or maximum) of the considered quantity, thus leading to less computational cost per iteration.

In addition to extending the algorithms introduced here to higher dimensions, there is plenty of other interesting future work that can be done. As noted earlier, we did not know of a nonuniform grid that ensures the presence of points on all three cusps of the Butterfly Spread payoff function, which is why we used a uniform grid. It would be nice in the future to work on a grid that has the properties of the grid used in [11] while also ensuring the existence of points on the three cusps.

Another interesting area to work on is the fact that despite the negative results in [17] on the use of Crank-Nicolson discretization for the Uncertain Volatility problem, we have not had numerical difficulties in the implementation, and not only do our policy and penalty methods agree with each other on the numerical solution, but our results also agree with the correct results obtained with fully implicit timestepping in [17], as long as Rannacher smoothing is applied. We also proved convergence for the penalty iteration algorithm for this problem. It would be worth revisiting this problem from the HJB perspective and try to prove convergence with Crank-Nicolson-Rannacher timestepping.

For longer term research, we are interested in extending the penalty methods to more general problems in finance. For example, the problem considered in [32] is a quite general HJB variational inequality, which involves a minimization (or maximization) and within the minimized

quantity there is a nonlinear term involving the unknown function and its first spatial derivative, as well as other terms. The authors of [32] handle the minimization by policy iteration and the nonlinear term by diagonal penalty matrices. It would be interesting to consider tridiagonal penalty matrices as well as writing the problem without minimization quantities, but with extra penalty terms, and use a penalty iteration for its solution. This technique will avoid the explicit minimization over all possible control cases, and potentially lead to a more efficient solution technique.

# Chapter 6

# Bibliography


[1] M. AVELLANEDA, A. LEVY, AND A. PARAS, *Pricing and hedging derivative securities in markets with uncertain volatilities*, Applied Mathematical Finance, 2 (1995), pp. 73–88.

[2] R. BELLMAN, *On the theory of dynamic programming*, Proceedings of the National Academy of Sciences of the United States of America, 38 (1952), pp. 716–719.

[3] R. BELLMAN ET AL., *The theory of dynamic programming*, Bulletin of the American Mathematical Society, 60 (1954), pp. 503–515.

[4] Y. BERGMAN, *Option pricing with differential interest rates*, The Review of Financial Studies, 8 (1995), pp. 475–500.

[5] A. BERMAN AND R. PLEMMONS, *Nonnegative matrices in the mathematical sciences*, SIAM, 1994.

[6] F. BLACK AND M. SCHOLES, *The pricing of options and corporate liabilities*, Journal of political economy, 81 (1973), pp. 637–654.

[7] Y. CHEN AND C. CHRISTARA, *Penalty methods for bilateral XVA pricing in European and American contingent claims by a PDE model*, Journal of Computational Finance, (2021). to appear.

[8] Y. CHEN AND J. W. WAN, *Deep neural network framework based on backward stochastic differential equations for pricing and hedging American options in high dimensions*, Quantitative Finance, (2020), pp. 1–23.

[9] C. CHRISTARA AND D.-M. DANG, *Adaptive and high-order methods for valuing American options*, Journal of Computational Finance, 14 (2011), pp. 73–113.

[10] C. CHRISTARA AND N. C.-H. LEUNG, *Analysis of quantization error in financial pricing via finite difference methods*, SIAM Journal on Numerical Analysis, 56 (2018), pp. 1731–1757.

[11] N. CLARKE AND K. PARROTT, *Multigrid for American option pricing with stochastic volatility*, Applied Mathematical Finance, 6 (1999), pp. 177–195.

[12] J. CRANK AND P. NICOLSON, *A practical method for numerical evaluation of solutions of partial differential equations of the heat-conduction type*, in Mathematical Proceedings of the Cambridge Philosophical Society, vol. 43, Cambridge University Press, 1947, pp. 50–67.

[13] D. DUFFIE, N. GARLEANU, AND L. H. PEDERSEN, *Securities lending, shorting, and pricing*, Journal of Financial Economics, 66 (2002), pp. 307–339.

[14] P. FORSYTH AND G. LABAHN, *Numerical methods for controlled Hamilton-Jacobi-Bellman PDEs in finance*, Journal of Computational Finance, 11 (2007), pp. 1–43.

[15] ——, *$\varepsilon$-monotone Fourier methods for optimal stochastic control in finance*, Journal of Computational Finance, 22 (2019), pp. 25–71.

[16] P. FORSYTH AND K. VETZAL, *Quadratic convergence for valuing American options using a penalty method*, SIAM J. Sci. Comput., 23 (2002), pp. 2095–2122.

[17] ——, *Numerical methods for nonlinear PDEs in finance*, in Handbook of Computational Finance, Springer, 2012, pp. 503–528.

[18] M. GILES AND R. CARTER, *Convergence analysis of Crank-Nicolson and Rannacher time-marching*, Journal of Computational Finance, 9 (2006), pp. 89–112.

[19] J. HAN, A. JENTZEN, AND W. E, *Solving high-dimensional partial differential equations using deep learning*, Proceedings of the National Academy of Sciences, 115 (2018), pp. 8505–8510.

[20] M. HEATH, *Scientific computing: an introductory survey*, SIAM, 2018.

[21] S. D. HOWISON, C. REISINGER, AND J. H. WITTE, *The effect of nonsmooth payoffs on the penalty approximation of American options*, SIAM Journal on Financial Mathematics, 4 (2013), pp. 539–574.

[22] Y. M. KABANOV AND M. M. SAFARIAN, *On Leland's strategy of option pricing with transactions costs*, Finance and Stochastics, 1 (1997), pp. 239–250.

[23] A. KHALIQ, D. VOSS, AND M. YOUSUF, *Pricing exotic options with L-stable Padé schemes*, Journal of Banking & Finance, 31 (2007), pp. 3438–3461.

[24] H. E. LELAND, *Option pricing and replication with transactions costs*, The Journal of Finance, 40 (1985), pp. 1283–1301.

[25] H. E. LELAND ET AL., *Comments on "Hedging errors with Leland's option model in the presence of transactions costs"*, Finance Research Letters, 4 (2007), pp. 200–202.

[26] O. MANGASARIAN, *Characterizations of real matrices of monotone kind*, SIAM Review, 10 (1968), pp. 439–441.

[27] C. PARKINSON, D. ARNOLD, A. BERTOZZI, Y. T. CHOW, AND S. OSHER, *Optimal human navigation in steep terrain: A Hamilton-Jacobi-Bellman approach*, arXiv preprint arXiv:1805.04973, (2018).

[28] D. POOLEY, K. VETZAL, AND P. FORSYTH, *Convergence remedies for non-smooth payoffs in option pricing*, Journal of Computational Finance, 6 (2003), pp. 25–40.

[29] D. M. POOLEY, P. A. FORSYTH, AND K. R. VETZAL, *Numerical convergence properties of option pricing PDEs with uncertain volatility*, IMA Journal of Numerical Analysis, 23 (2003), pp. 241–267.

[30] R. RANNACHER, *Finite element solution of diffusion problems with irregular data*, Numerische Mathematik, 43 (1984), pp. 309–327.

[31] C. REISINGER AND P. A. FORSYTH, *Piecewise constant policy approximations to Hamilton–Jacobi–Bellman equations*, Applied Numerical Mathematics, 103 (2016), pp. 27–47.

[32] C. REISINGER AND Y. ZHANG, *A penalty scheme and policy iteration for nonlocal HJB variational inequalities with monotone drivers*, arXiv preprint arXiv:1805.06255, (2018).

[33] M. S. SANTOS AND J. RUST, *Convergence properties of policy iteration*, SIAM Journal on Control and Optimization, 42 (2004), pp. 2094–2115.

[34] R. TAKEI, R. TSAI, H. SHEN, AND Y. LANDA, *A practical path-planning algorithm for a simple car: a Hamilton-Jacobi approach*, in Proceedings of the 2010 American control conference, IEEE, 2010, pp. 6175–6180.

[35] P. WILMOTT, T. HOGGARD, AND A. E. WHALLEY, *Hedging option portfolios in the presence of transaction costs*, Advances in Futures and Options Research, 7 (1994), pp. 21–35.

[36] P. WILMOTT, S. HOWISON, AND J. DEWYNNE, *The mathematics of financial derivatives: a student introduction*, Cambridge university press, 1995.

[37] M. YOUSUF, A. KHALIQ, AND B. KLEEFELD, *The numerical approximation of nonlinear Black–Scholes model for exotic path-dependent American options with transaction cost*, International Journal of Computer Mathematics, 89 (2012), pp. 1239–1254.

[38] Y. ZHAO AND W. T. ZIEMBA, *Hedging errors with Leland's option model in the presence of transaction costs*, Finance Research Letters, 4 (2007), pp. 49–58.