On Convergence of Numerical Methods for Pricing Convertible Bonds

by

Dongyi Li

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

Abstract

On Convergence of Numerical Methods for Pricing Convertible Bonds

Dongyi Li

Master of Science

Graduate Department of Computer Science

University of Toronto

2007

In this thesis two frameworks are considered to value convertible bonds with credit risk: the TF model (Tsiveriotis and Fernandes) and the AFV model (Ayache, Forsyth and Vetzal). Both models are associated with a pair of coupled partial differential equations (PDEs) with free boundary constraints. In practice, the projected overrelaxation method and the penalty method are widely used to solve such free-boundary value problems, and the Crank-Nicolson time-stepping scheme is adopted as the underlying discretization method to achieve quadratic precision. However, due to the complexity of the PDEs in these two models and discontinuities in practice present in the boundary conditions, only linear convergence is observed in many cases. The objective of this thesis is to investigate the difficulties related to convergence and stability when solving these coupled PDEs with the Crank-Nicolson scheme, and to suggest some modifications of the basic schemes to improve stability and restore quadratic convergence.

Acknowledgements

This work was supported by the Computer Science Department of University of Toronto and the Natural Sciences and Engineering Research Council of Canada (NSERC). I would like to thank them and many people who have in one way or another made this thesis possible. First of all, I wish to express my sincere gratitude to my supervisor Professor Kenneth R. Jackson for all his help, support, valuable discussions, and thorough comments. I am also extremely grateful to Professor Christina C. Christara for her helpful advice and careful comments on my work. I would like to thank all members of the Numerical Analysis Group of the Computer Science Department for their help and friendly discussions. At last, I would like to extend a special thanks to my parents whose support enabled me to complete this work.

Contents

1 Introduction **1**

2 The Pricing Models **6**

2.1 Plain-Vanilla European Options 7

2.2 Plain-Vanilla American Options 9

2.3 Convertible Bonds 12

2.3.1 TF Model 13

2.3.2 AFV Model 18

2.3.3 Connection Between the TF Model and the AFV Model 22

2.3.4 Issues Concerning Cash Flows 23

3 The Finite Difference Methods **26**

3.1 General Black-Scholes PDE 27

3.2 Variable Transformation for PDEs 28

3.3 Discretization 28

3.3.1 Space Discretization 29

3.3.2 Time Discretization 32

3.4 Special Issues 34

3.4.1 Far Field 34

3.4.2 Boundary Conditions 35

3.4.3 A Few Additional Considerations 37

3.5 The Constrained Linear System for Each Pricing Model 41

4 Iterative Methods for Solving Free Boundary Problems 47

4.1 SOR / PSOR Method 48

4.1.1 Introduction to the SOR/PSOR Method 48

4.1.2 Implementation for European Options 50

4.1.3 Implementation for American Options 53

4.1.4 Implementation for Convertible Bonds 54

4.2 Penalty Method 63

4.2.1 Discrete Penalty Method for American Options 66

4.2.2 Continuous Penalty Method for American Options 69

4.2.3 Implementation of the Discrete Penalty Method for the TF Model 70

4.2.4 Implementation of the Discrete Penalty Method for the AFV Model 76

5 Observations and Improvements 82

5.1 Observations 88

5.1.1 M-matrix 88

5.1.2 Numerical Results for Linear Boundary Conditions 94

5.1.3 Far-Field Effects 97

5.1.4 Effects of Nonsmooth Payoff Functions 102

5.2 Improvements 110

5.2.1 Rannacher Smoothing and Controlling $\lambda = \frac{\Delta\tau}{h}$ or $\Gamma = \frac{\Delta\tau}{h^2}$ 110

5.2.2 Grid Positioning 116

5.2.3 Removing Oscillations in Penalty Methods 123

5.2.4 Adaptive Spatial Grids 124

5.2.5 Adaptive Time-Stepping 130

5.3 Numerical Results for CBs 132

5.3.1 TF Model 132

5.3.2 AFV Model 135

6 Conclusions and Future Work **137**

Bibliography **141**

List of Tables

5.1 Model parameters for pricing the European/American options 85

5.2 TF model parameters for pricing the convertible bond 86

5.3 AFV model parameters for pricing the convertible bond 87

5.4 Numerical results for a European call option at $S = 100$ using the SOR method $(1 < \omega < 2)$ to solve the associated linear equations. The numerical method uses the linear boundary condition (5.10), $tol = 1.0\text{e}^{-7}$, and $S_{max} = 400$. The analytical solution is 16.92091465. 90

5.5 Numerical results for an American put option at $S = 100$ using the PSOR method $(1 < \omega < 2)$ to solve the nonlinear equations associated with the free-boundary condition. The numerical method uses linear boundary conditions (5.10), $tol = 1.0\text{e}^{-7}$ and $S_{max} = 400$. Compare our results with the solution 14.67883348 in [Li05] and the solution 14.67882 in [FV02]. . 91

5.6 Comparison of the numerical results for a European call option at $S = 100$ based on the linear boundary conditions (5.10) with those based on Dirichlet boundary conditions. In both cases, we used $tol = 1.0\text{e}^{-7}$ and $S_{max} = 400$. The analytical solution at $S = 100$ is given. 95

5.7 Comparison of the numerical results for an American put option at $S = 100$ based on the linear boundary conditions (5.10) with those based on Dirichlet boundary conditions. In both cases, we used $S_{max} = 400$ and $tol = 1.0\mathrm{e}^{-7}$. The solution is 14.67883348 from [Li05] and 14.67882 from [FV02]. 96

5.8 The numerical results for an American put option at $S = 100$ using the discrete penalty method with different far-field values S_{max}. In all cases, $tol = 1.0\mathrm{e}^{-7}$. 99

5.9 The numerical results for the price of a simple CB (without coupon payments and without any call or put features) under the TF model at $S = 100$ using different far-field values S_{max}. The discrete penalty method with $tol = 1.0\mathrm{e}^{-7}$ was used in the computation. 100

5.10 The numerical results for the price of a simple CB (without coupon payments and without any call or put features) under the AFV model at $S = 100$ using different far-field values S_{max}. The discrete penalty method with $tol = 1.0\mathrm{e}^{-7}$ was used in the computation. 101

5.11 Comparison of the numerical results for a European call option at $S = 1$ with Rannacher smoothing and without. We ensure that the kink point of the payoff function is at a grid point. $S_{max} = 4$, $\lambda = \frac{\Delta\tau}{h} = 10$, and $tol = 1.0\mathrm{e}^{-7}$. 103

5.12 Comparison of the numerical results for the Crank-Nicolson method with Rannacher smoothing and without for a simple CB (without coupon payments and without any call/put features) at $S = 100$ under the TF model. We put the critical points of the initial functions at a grid point, and keep $\lambda = 0.25$ constant. Here, $tol = 1.0\mathrm{e}^{-7}$. 106

5.13 Comparison of the numerical results for the Crank-Nicolson with Rannacher smoothing and without for a simple CB (without coupon payments and without any call/put features) at $S = 100$ under the AFV model. We put the discontinuity points of the initial functions at a grid point, and keep $\lambda = 0.25$ constant. Here, $tol = 1.0\mathrm{e}^{-7}$ 108

5.14 Comparison of the numerical results using the Crank-Nicolson method for a European call option at $S = 1$ with and without Rannacher smoothing and with different values of the ratio $\lambda = \frac{\Delta\tau}{h}$. We put the kink point of the payoff function at a grid point. $S_{max} = 4$ and $tol = 1.0\mathrm{e}^{-7}$. 113

5.15 The numerical results for the Crank-Nicolson method without Rannacher smoothing for a European call option at $S = 1$. We put the kink point of the payoff function at a grid point and keep $\Gamma = \frac{\Delta\tau}{h^2}$ constant. $S_{max} = 4$ and $tol = 1.0\mathrm{e}^{-7}$. 116

5.16 Comparison of the numerical results for the Crank-Nicolson with and without Rannacher smoothing for a simple CB (without coupon payments and without any call/put features) at $S = 100$ under the TF model. We put the critical point of the initial functions at a grid point, and keep $\lambda = 0.01$ constant. Here, $tol = 1.0\mathrm{e}^{-7}$. 118

5.17 Numerical results for the Crank-Nicolson method without Rannacher smoothing for a simple CB (without coupon payments and call/put features) at $S = 100$ under the AFV model. We put the critical point of the initial functions at a grid point, and keep $\lambda = 0.01$ constant. Here, $tol = 1.0\mathrm{e}^{-7}$. 120

5.18 Numerical results for the Crank-Nicolson method without Rannacher smoothing, but using grid positioning for B for a simple CB (without coupon payments and without any call/put features) under the TF model at $S = 100$. Here, $\lambda = 0.01$ and $tol = 1.0\mathrm{e}^{-7}$. 123

5.19 Numerical results at $S = 100$ using the Crank-Nicolson method with the discrete penalty method for an American put option. Here, $tol = 1.0\mathrm{e}^{-7}$. 124

5.20 Numerical results at $S = 100$ using the Crank-Nicolson method with the continuous penalty method for an American put option. Here, $tol = 1.0\mathrm{e}^{-7}$ and $\chi = 2rE$. 125

5.21 Numerical results for the Crank-Nicolson method using the discrete penalty method for an American put option at $S = 100$ using uniform grid spacing and $tol = 1.0\mathrm{e}^{-8}$. 128

5.22 Numerical results for the Crank-Nicolson method using the discrete penalty method for an American put option at $S = 100$ using adaptive spatial grids and $tol = 1.0\mathrm{e}^{-8}$. 130

5.23 Numerical results for the Crank-Nicolson method using a constant time-step and the discrete penalty method for an American put option at $S = 100$. Here, $tol = 1.0\mathrm{e}^{-7}$. 131

5.24 Numerical results for the Crank-Nicolson method using an adaptive time-step scheme and the discrete penalty method for an American put option at $S = 100$. Here, $tol = 1.0\mathrm{e}^{-7}$. 132

5.25 Numerical results for a simple CB (without coupon payments and without any call/put features) at $S = 100$ using the TF model. We used $\lambda = 0.01$ and $tol = 1.0\mathrm{e}^{-7}$. 133

5.26 Numerical results for a semi-full CB (with coupon payments but without any call and put features) at $S = 100$ using the TF model. We used $\lambda = 0.01$ and $tol = 1.0\mathrm{e}^{-7}$. 134

5.27 Numerical results for a full CB (with coupon payments and with call and put features) at $S = 100$ using the TF model. We used $\lambda = 0.01$ and $tol = 1.0\mathrm{e}^{-7}$. 134

5.28 Numerical results for a simple CB (without coupon payments and without any call/put features) at $S = 100$ using the AFV model. We used $\lambda = 0.01$ and $tol = 1.0\mathrm{e}^{-7}$. 135

5.29 Numerical results for a semi-full CB (with coupon payments but without any call/put features) at $S = 100$ using the AFV model. We used $\lambda = 0.01$ and $tol = 1.0\mathrm{e}^{-7}$. 136

5.30 Numerical results for a full CB (with coupon payments and with call/put features) at $S = 100$ using the AFV model. We used $\lambda = 0.01$ and $tol = 1.0\mathrm{e}^{-7}$. 136

List of Figures

2.1 American put option: free boundary 11

2.2 Coupon payments 24

2.3 Dirty call price including accrued interest as a function of time 25

3.1 A non-uniform grid 29

5.1 The value $V(S,0)$ and the associated Delta and Gamma functions for an American put option at time $t = 0$ using PSOR with $n = 1280$ and $m = 1280$. 92

5.2 The value $V(S,0)$ and the associated Delta and Gamma functions for a European Call option with a nonsmooth hockey-stick payoff function. The parameters are $r = 0.05$, $\sigma = 0.25$, $E = 1$, and T=2. (A) shows the numerical results without Rannacher smoothing; (B) is with Rannacher smoothing. 104

5.3 Comparison of $U(S,0)$ and the associated Delta and Gamma functions for a simple CB (without coupon payments and without any call/put features) at $S = 100$ under the TF model computed by the Crank-Nicolson method with Rannacher smoothing and without. We put the critical points of the initial functions at a grid point, and keep $\lambda = 0.25$ constant. (A) shows the numerical results without Rannacher smoothing; (B) is with Rannacher smoothing. 107

5.4 Comparison of $U(S,0)$ and the associated Delta and Gamma functions for a simple CB (without coupon payments and without any call/put features) at $S = 100$ under the AFV model computed by the Crank-Nicolson method with Rannacher Smoothing and without. We put the discontinuity points of the initial functions at a grid point, and keep $\lambda = 0.25$ constant. (A) shows the numerical solutions without Rannacher smoothing; (B) is with Rannacher smoothing. 109

5.5 The value $V(S,0)$ and the associated Delta and Gamma functions for a European call option with a piecewise linear payoff function. The parameters are $r = 0.05$, $\sigma = 0.25$, $E = 1$, and T=2. The Crank-Nicolson method without Rannacher smoothing is used. We put the kink point of the payoff function at a grid point and keep $\lambda = \frac{\Delta\tau}{h}$ constant. For plots (A), $\lambda = 3$. For plots (B), $\lambda = 2.5$. 114

5.6 The value $V(S,0)$ and the associated Delta and Gamma functions for a European call option with a piecewise linear payoff function. Parameters are $r = 0.05$, $\sigma = 0.25$, $E = 1$, and T=2. The Crank-Nicolson method without Rannacher smoothing is used and we put the kink point of the payoff function at a grid. Here, $\Gamma = 3.125$ and $\lambda = 0.039$. 115

5.7 Comparison of $U(S,0)$ and the associated Delta and Gamma functions of a simple CB (without coupon payments and without any call/put features) at $S = 100$ under the TF model computed by the Crank-Nicolson method with and without Rannacher smoothing. We put the critical point of the initial functions at a grid point, and keep $\lambda = 0.25$ constant. Plots (A) show the numerical results with Rannacher smoothing; Plots (B) are without Rannacher smoothing. 119

5.8 The value $U(S,0)$ and the associated Delta and Gamma functions for a simple CB (without coupon payments and without any call/put features) under the AFV model. We used the Crank-Nicolson method without Rannacher smoothing and we put the critical point of the initial function at a grid point. $F = 100$, $n = 1280$, and $m = 1280$. 121

5.9 The value $U(S,0)$ and the associated Delta and Gamma functions for a CB under the TF model for which the associated U has a piecewise linear initial function and B has a step initial function. The Crank-Nicolson method with Rannacher smoothing is used. $\lambda = \frac{\Delta\tau}{h} = 0.01$. 122

5.10 Adaptive spatial grids. 126

5.11 The spatial grid near the free boundary is refined as τ increases. 126

5.12 Comparison of the free boundary curves using uniform spatial grids and adaptive spatial grids with refinement around the free boundary. 129

List of Algorithms

4.1 SOR method for European call options 52

4.2 Crank-Nicolson method for European call options 53

4.3 PSOR method for American put options 55

4.4 Explicit application of the free boundary conditions to U_i^{j+1} and B_i^{j+1} in the TF model. 58

4.5 Crank-Nicolson method for convertible bonds using the TF model 59

4.6 PSOR method for convertible bonds using the TF model 60

4.7 Explicit application of the free boundary conditions to C_i^{j+1} and B_i^{j+1} in the AFV model. 63

4.8 Crank-Nicolson timestepping for convertible bonds using the AFV model . 64

4.9 PSOR Iteration for convertible bonds using the AFV model 65

4.10 Discrete penalty iteration for American put options 69

4.11 Discrete penalty iteration to solve for U^{j+1} in the TF model 76

4.12 Discrete penalty iteration for the TF model 77

4.13 Discrete penalty iteration for the AFV model 81

Chapter 1

Introduction

Two important investment markets are the fixed-income market and the equity market. The former includes instruments such as government bonds and corporate bonds, while the latter includes stocks. A bond is a debt instrument issued for a period of time with the purpose of raising capital for the issuing institution by borrowing. Generally, a bond holder receives the principal at the bond maturity and interest regularly during the bond life. The interest usually is in terms of coupons paid annually or semiannually. Unlike bonds, a stock is an instrument which gives the holder ownership in a company and entitles the holder to a share in the company's assets and profits. These two instruments are subject to different risks and rewards. Generally speaking, bonds are associated with interest and credit risk while stocks are subject to price movements.

A derivative is a financial instrument whose value depends on the values of one or more underlying securities, such as a commodity, bond, equity or currency. An example of a derivative is an option. A call (put) option, for example, gives the holder the right to buy (sell) the underlying by a certain date for a certain price. Two common subclasses of options are European options and American options. American options can be exercised any time up to and including the maturity date while European options can be exercised at maturity only. A bond also can be considered to be a derivative. For example, we can view a government bond as a derivative of the interest rate; a corporate bond's value depends not only on the prevailing interest rate, but also on other factors such as credit risk. Thus, a corporate bond can be regarded as a derivative of the company's assets, as discussed in [BS77, BS80, Ing77], when taking the company's credit quality into account.

In recent decades, the market for convertible bonds has grown rapidly. A convertible bond (CB) is a bond issued by a company for which the holders have the option to exchange the bond for the company's stock at any time of their choosing until the maturity of the bond. It is essentially a bond embedded with an American-style call option on the underlying stock. Before a CB is converted to a stock it behaves like an ordinary bond. Upon conversion, it behaves like a stock. Thus, a convertible bond is a hybrid instrument combining features of fixed-income instruments and equity instruments, bearing some of the risks of both classes. Two features often added to CBs are callability and puttability. The issuer of a callable bond has the right to purchase back the bond at a predetermined price during specified periods. The holder of a puttable bond has the right to sell the bond back to the issuer at a predetermined price at specified times. They are essentially options on bonds.

Pricing CBs is a difficult problem not only because of their hybrid nature described above, but also because corporate bonds expose the holder to credit risk. Ingersoll [Ing77] and Brennan and Schwartz [BS77, BS80] propose a structured approach for valuing risky CBs. They use the total value of the issuing company as the underlying variable. However, because this is not a traded asset, it is difficult to estimate. Alternatively, McConnell and Schwartz [MS86] suggest a pricing model based on the traded stock price as the underlying variable, but their model of credit risk is based simply on a risky discount rate. The models created by Cheung and Nelken [CN94] and Ho and Pfeffer [HP96] handle credit risk in a similar way. However, applying the same risky discount rate to both the bond and equity elements is highly questionable.

More recently, Tsiveriotis and Fernandes [TF98] introduced an effective model, which we refer to as the TF model, for pricing CBs with credit risk. The TF model is also based on the stock price, but has the significant advantage that it models the hybrid nature of a CB in a reasonable fashion. More specifically, it splits a CB into two components: a cash-only part, which is subject to credit risks and requires that a risky discount rate be applied to this component, and an equity part, which is not subject to credit risks and requires that a risk-free discount rate be applied to this component. This splitting leads to two coupled Black-Scholes-like partial differential equations (PDEs). Ayache, Forsyth and Vetzal [AFV02, AFV03] extend the TF model by incorporating a more sophisticated model for credit risks. We refer to their model as the AFV model. It explicitly describes the behavior of the equity and fixed-income components of a CB in the event of a default by the issuing company. The model assumes that default events follow a Poisson process, and that the underlying stock price erodes with time prior to a default and declines dramatically at a critical time, such as the announcement of a default. This contrasts with the TF model in which the stock price remains constant at a default event. Consequently, splitting a CB into equity and fixed-income components is slightly different in the TF and AFV models. This is discussed in more detail in Chapter 2.

Both the TF and AFV models rely on the Black-Scholes analysis [BS73, Mer73]. Not surprisingly, they produce a similar pair of coupled one-factor Black-Scholes-like PDEs. The American-style options embedded in CBs lead to a free boundary problem, which can be transformed into a differential linear complementarity problem (LCP). Closed-form analytic solutions are not known for this problem. Therefore, effective numerical methods are required. In this thesis, we consider finite difference methods to solve the associated LCPs.

Many numerical methods are available to discretize the PDEs associated with the TF and AFV models. Explicit methods are easy to implement, but suffer from stability issues; the implicit method based on the backward Euler time-stepping scheme is unconditionally stable, but exhibits only linear convergence. The Crank-Nicolson method combines features of explicit and implicit methods, and is unconditionally stable with quadratic convergence. Thus, it is widely adopted for time discretization. However, the presence of discontinuities in final conditions and boundary conditions, together with free boundaries, makes quadratic convergence difficult to attain in practice. Many of the numerical results listed in [AFV03], [Li05], and [Mo06] exhibit approximately linear convergence.

Two popular iterative methods to solve LCPs are the projected overrelaxation method (PSOR) and the penalty method. The PSOR method is a classical method which is easy to implement, but we have found it to be inefficient for solving complicated pricing problems associated with CBs. In part, this inefficiency is related to the difficulty of selecting an optimal relaxation factor. The penalty method is suggested in [FV02] for solving the LCPs associated with pricing American-style derivatives. Implemented with a Newton-like iteration, it is often fast and effective, but in some cases can suffer from oscillations that greatly reduces its effectiveness.

Li [Li05] and Mo [Mo06] have applied the Crank-Nicolson method to convertible bond problems using the TF and AFV models, and used the PSOR and penalty methods as the iterative solvers. Both demonstrate convergence and stability difficulties in their experiments, but did not study these difficulties in depth. This thesis explores many issues associated with pricing convertible bonds using the TF and AFV models. In particular, we investigate the convergence and stability difficulties in solving these PDEs. We restrict our attention to discontinuous boundary/final conditions, free boundaries, far field selection, and convection problems. We suggest some remedies to enhance stability, restore quadratical convergence, and achieve high accuracy effectively. Illustrative examples are provided for European and American stock options and for CBs. Furthermore, we compare our numerical results with those in other papers, such as [Li05] and [Mo06]. In this thesis, a divide-and-conquer technique is used to demonstrate the difficulties step by step, starting with simple plain-vanilla European puts and American puts then progressing to more complicated convertible bonds with additional features such as coupon payments, callability and puttability.

The thesis is structured as follows. Chapter 2 introduces options and convertible bonds, along with their mathematical pricing models and the associated PDEs, final conditions and boundary conditions. Emphasis is put on the pricing models for convertible bonds. In particular, the TF model and the AFV model are examined in detail. Chapter 3 discusses numerical methods for the problems described in Chapter 2. The general form of the PDE is given first and then a simple transformation is suggested to simplify the problem. In section 3.3, we discuss the space discretization and time discretization of a general parabolic partial differential equation, and then derive the constrained linear systems associated with the TF and AFV models. Some special issues are reported in section 3.4. In Chapter 4, we describe the algorithms for solving the constrained linear systems obtained in Chapter 3. Two methods are considered: the PSOR method and the penalty method. In Chapter 5, we present the numerical results, analyze the convergence and stability problems associated with the TF and AFV models, and suggest some modifications to the numerical methods to improve stability and the convergence rate. Finally, we draw some conclusions and discuss future work in Chapter 6.

Chapter 2

The Pricing Models

The primary assumption underlying the Black-Scholes analysis for options and other derivatives concerns the stochastic process governing the price of the underlying asset. More specifically, the price of the underlying asset, S, is assumed to follow a geometric Brownian Motion process,

$$dS = \mu S dt + \sigma S dz, \tag{2.1}$$

where μ denotes the expected return of the underlying asset, σ denotes the volatility, and z is a Brownian motion. In the risk-neutral world, μ is equal to the risk-free interest rate r. More detailed background on Brownian motion and risk-neutral valuation can be found in [Shr04].

Based on the assumptions that there are no arbitrage opportunities, there are no dividends paid on the underlying asset S, and an investor can trade continuously, Black and Scholes [BS73] and Merton [Mer73] show that the price of any derivative driven by the movement of the asset S must satisfy the partial differential equation (PDE)

$$\frac{\partial V}{\partial t} + \frac{1}{2}\sigma^2 S^2 \frac{\partial^2 V}{\partial S^2} + rS\frac{\partial V}{\partial S} - rV = 0. \tag{2.2}$$

We refer to this equation as the Black-Scholes PDE. Here V is the price of the derivative and is a function of t and S. For simplicity, we assume r and σ are constants in this thesis, but more generally they may be functions of t and/or S.

If the underlying asset S pays continuous dividends during the life of a derivative, then the Black-Scholes equation can be extended to

$$\frac{\partial V}{\partial t} + \frac{1}{2}\sigma^2 S^2 \frac{\partial^2 V}{\partial S^2} + (r-q)S\frac{\partial V}{\partial S} - rV = 0, \tag{2.3}$$

where q is the continuous dividend yield.

The Black-Scholes equation (2.2) is a parabolic PDE. It may have many solutions. To obtain a unique solution for a particular pricing model, the Black-Scholes equation must be associated with additional constraints, which usually take the form of final conditions, boundary conditions, and/or free boundary conditions.

In the following sections, we introduce European options, American options, and convertible bonds, as well as the pricing models for valuing them.

2.1 Plain-Vanilla European Options

An option is a financial instrument which gives the holder the right, but not the obligation, to exercise the option by a certain date for a certain price. Two main categories of options are European options and American options. A European option can be exercised at its maturity only, while an American option can be exercised any time up to and including its maturity. A plain-vanilla option is a simple option for which the underlying asset doesn't pay any dividends. In this section, we describe the pricing model for plain-vanilla European options; the model for American options is discussed in the next section.

Two types of European options are European call options, which give the holder the right to buy the underlying asset S for the strike price, E, at the maturity time, T, and European put options, which give the holder the right to sell the underlying asset at the maturity time, T, for the strike price, E. Both are derivatives driven by the movements of the underlying asset S. Thus, their prices must satisfy the Black-Schole equation (2.2).

To distinguish a call option from a put option, we need to specify their final conditions and boundary conditions in the pricing models.

Because the pricing model of a European call and a European put are similar (both are governed by the same PDE), we consider the European call only here. The pricing model for a European put is similar and can be found in [WHD95]. The following model is the pricing model for a plain-vanilla European call on a non-dividend-paying stock.

If $S > E$ at maturity T, the option holder should exercise the call option, thereby buying the asset for E. If they immediately sell the asset for S, they will make a profit of $(S - E)$. If $S \leq E$, the holder should not exercise the option: the option expires worthlessly. Thus, the value of the option at maturity is $\max(S - E, 0)$. Therefore, the final condition for the European call option is $V(S,T) = \max(S - E, 0)$. This is often called the payoff function.

The boundary conditions can be derived from financial arguments. If S is ever zero, we can see from (2.1) that dS is also zero and therefore S never changes. Therefore, at maturity T, S is zero and the call option expires worthlessly. Thus, if $S = 0$ at any time t, the value of the option equals zero, i.e., $V(0,t) = 0$. As S tends to infinity, the option will be exercised almost surely. Therefore, the value of the option is the stock price, S, minus the discounted strike price, $E\mathrm{e}^{-r(T-t)}$. Because the discounted strike price is so small relative to the stock price, the value of the option is well approximated by stock price, S. That is, $V(S,t) \approx S$ as $S \to \infty$.

Summarizing the discussion above, the price of an European call option, $V(S,t)$, can be determined by solving the Black-Scholes PDE

$$\frac{\partial V}{\partial t} + \frac{1}{2}\sigma^2 S^2 \frac{\partial^2 V}{\partial S^2} + rS\frac{\partial V}{\partial S} - rV = 0 \tag{2.4}$$

subject to the boundary conditions

$$V(0,t) = 0, \tag{2.5}$$

$$V(S,t) = S - E\mathrm{e}^{-r(T-t)} \approx S, \qquad \text{as } S \to \infty, \tag{2.6}$$

and the final condition

$$V(S,T) = \max(S - E, 0). \tag{2.7}$$

In [WHD95], the analytic solution is given by

$$V(S,t) = SN(d_1) - E\mathrm{e}^{-r(T-t)}N(d_2), \tag{2.8}$$

where

$$d_1 = \frac{\ln(S/E) + (r + \frac{1}{2}\sigma^2)(T-t)}{\sigma\sqrt{T-t}} \tag{2.9}$$

and

$$d_2 = d_1 - \sigma\sqrt{T-t}. \tag{2.10}$$

Here, the function $N(x)$ is the cumulative probability distribution function for the standard normal distribution with mean 0 and variance 1. Equations (2.8)-(2.10) are often referred to as the Black-Scholes formula for a plain-vanilla European call option. A similar formula exists for a plain-vanilla European put option. See [WHD95] for example.

2.2 Plain-Vanilla American Options

Unlike European options, American options can be exercised at any time up to and including the maturity date T. This added flexibility over European options results in an American option having a value that is greater than or equal to that of a similar European option. Often the value of an American option is close to that of the corresponding European option because the American option optimal exercise date is close to or equal to the expiry date. Hull [Hul06] shows that it is never optimal to exercise early an American call on a non-dividend-paying stock. Thus, its value is the same as that of a similar European call option. However, it is sometimes optimal to exercise early a plain-vanilla American put option. In this section, we consider a pricing model for a plain-vanilla American put option.

The early exercise feature complicates the valuation of American put options. As a result, no closed-formed analytical solution is known for these options, because the point at which early exercise is optimal is unknown in advance. Thus, we need to compute the optimal exercise point simultaneously with the option price V. This results in a free boundary problem for V. An effective way to solve such a free boundary problem is to reformulate it as a linear complementarity problem (LCP) and then solve it numerically.

Consider a plain-vanilla American put option on a non-dividend-paying stock. In [WHD95], the authors show that the early exercise feature leads to the price of an American put option satisfying the Black-Scholes inequality

$$\frac{\partial V}{\partial t} + \frac{1}{2}\sigma^2 S^2 \frac{\partial^2 V}{\partial S^2} + rS\frac{\partial V}{\partial S} - rV \leq 0, \tag{2.11}$$

together with the additional constraint that the option price is always greater than or equal to the payoff $G(S) = \max(E - S, 0)$ at any time during the life of the option:

$$V(S,t) \geq G(S). \tag{2.12}$$

We call (2.12) the free boundary condition. At each time that we compute the option price $V(S,t)$, we also need to find the optimal exercise point $S_f(t)$. By implication from (2.12), $S_f(t)$ is the point at which (2.12) changes from an equality to an inequality. As the notation indicates, the free boundary varies with time, t. Figure 2.1 sketches the curve S_f as a function of t.

Note that S_f divides the solution domain into two regions. Define the operator

$$\mathcal{L} = \frac{\partial}{\partial t} + \frac{1}{2}\sigma^2 S^2 \frac{\partial^2}{\partial S^2} + rS\frac{\partial}{\partial S} - r. \tag{2.13}$$

If the option price $V > G$, the option should be held and $\mathcal{L}V = 0$ must be satisfied. If $\mathcal{L}V < 0$, the option should be exercised early and $V = G$ in this case. This can be formulated as the LCP

$$\begin{pmatrix} \mathcal{L}V = 0 \\ V \geq G \end{pmatrix} \vee \begin{pmatrix} \mathcal{L}V \leq 0 \\ V - G = 0 \end{pmatrix}, \tag{2.14}$$

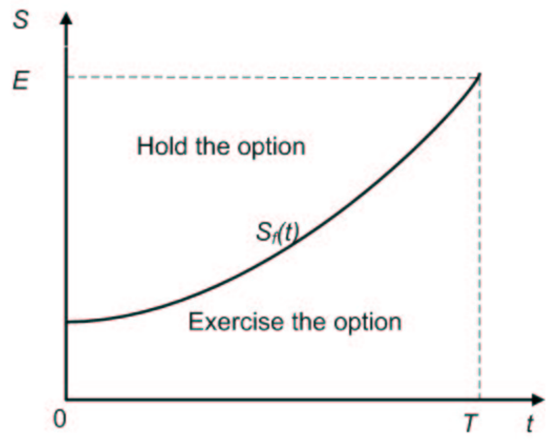


Figure 2.1: American put option: free boundary

where the notation $(x = 0) \vee (y = 0)$ means that either $(x = 0)$ or $(y = 0)$ at each point in the solution domain.

Using an argument similar to that for a European put option, we can conclude that the final condition for an American put is

$$V(S, T) = \max(E - S, 0). \tag{2.15}$$

The boundary conditions are as follows. If S tends to infinity at some time t, the option shouldn't be exercised at time t. However, it is very unlikely that the option will ever be in the money. So we should take its value to be zero. If S equals zero, the option should be exercised and its value is determined by the payoff at that time. Thus, we have the Dirichlet boundary conditions

$$V(0, t) = G(0) = E, \tag{2.16}$$

$$V(S, t) = 0, \quad S \to \infty. \tag{2.17}$$

We give other boundary conditions in Chapter 3 that are linear boundary conditions derived from the PDE itself to replace the Dirichlet boundary conditions (2.16) and (2.17).

2.3 Convertible Bonds

A convertible bond (CB) is a corporate bond that gives the holder the right, but not the obligation, to exchange or convert the bond for common shares of the issuer at a fixed ratio during a particular period. It possesses features of both bonds and equity options. Before conversion, it is a normal bond and may receive coupon payments. Upon conversion, it becomes a stock and may receive dividends. Investors usually compare the value of the CB if not converted to the value if converted to decide when it is optimal to exercise. Therefore, a CB is regarded as a combination of a normal bond and an American call option on the underlying stock. Like a normal bond, its value depends on the level of the prevailing interest rate. Also, like a stock option, its value is driven by movements of the stock price. From this perspective, a CB can be viewed as a derivative of the underlying stock and interest rate.

CBs may have two other important features: a call feature and a put feature. The call feature gives the issuer the right to purchase back the CB at a predetermined price during specified periods. Once it has been called, the holder can either sell it back to the issuer or convert it to the underlying stock, depending on which choice is more beneficial to the holder. Thus, the call feature can be used by the issuer to force conversion earlier than the holder would otherwise choose. This feature benefits the issuer and decreases the value of the CB. On the other hand, the put feature permits the holder to sell back the CB to the issuer at a predetermined price at specified times, protecting the holder from CB price decreases. Therefore, a CB with a put feature is more valuable than a CB without one.

As corporate bonds, CBs are subject to higher credit risks than government bonds. Generally, the credit risk is not negligible. Thus, the credit quality of the issuer should be taken into account in a pricing model. However, because a CB is a hybrid of a bond and an equity option, with each component bearing different risks, we need to distinguish which part of the CB is associated with the credit risk and which part is not. Also we need to apply different discount factors to those different parts. A risky discount factor should be applied to the risky corporate bond part of the CB and a risk-free discount factor should be applied to the equity part of the CB. In the following, we introduce two pricing models for valuing CBs with credit risk: the TF model and the AFV model.

2.3.1 TF Model

As discussed earlier, a CB can be viewed as a derivative of the underlying stock and the interest rate. To price such a derivative, a two-factor Black-Scholes-like model could be used. However, we make the simplifying assumption that the stochastic movements of the interest rate have much less influence on the value of the CB than the variation in the underlying stock price. Under this assumption, a CB is treated as a derivative of the underlying stock only. Consequently, the pricing models are simplified, and we use a single-factor Black-Scholes-like model to value a CB. Extending the single-factor model to a two-factor model is straightforward but beyond the scope of our study.

Tsiveriotis and Fernandes [TF98] constructed the TF model based on a single-factor Black-Scholes-like analysis. The model produces a Black-Scholes-like PDE for the whole value of a CB. However, they realized that a part of the CB only is subject to credit risk. As noted above, different components of the CB bear different risks. In [TF98, page 95], the authors stat that "the equity upside has zero default since the issuer can always deliver its own stock [while] coupon and principal payments and any put provisions ... depend on the issuer's timely access to the required cash amounts, and thus introduce credit risk."

In order to apply credit risk to the appropriate parts, the TF model separates a CB into two components: one component which bears the credit risk and one component which does not. However, how to split the two components is not obvious. Tsiveriotis and Fernandes define the component bearing credit risk as the part for which the value accrues from the issuer's liability, either promised or contingent cash flows; they define the component not bearing credit risk as the part for which the value can always be delivered by the issuer [AFV02]. The former component is considered as a hypothetical security, labeled the "cash-only part of the CB" (COCB). Notice that not all cash flows are subject to credit risk. For example, when a call is triggered under upside situations, the cash flow can always be delivered and thus belongs to the risk-free component. In this model, we think of the COCB as a derivative of the underlying stock that can be valued with a Black-Scholes-like model. This leads to an additional Black-Schole-like equation. Thus, we have a pair of coupled Black-Scholes-like PDEs:

$$\text{CB:}\quad \frac{\partial U}{\partial t}+\frac{1}{2}\sigma^2 S^2\frac{\partial^2 U}{\partial S^2}+r_g S\frac{\partial U}{\partial S}-r(U-B)-(r+r_c)B+h(t)=0, \tag{2.18}$$

and

$$\text{COCB:}\quad \frac{\partial B}{\partial t}+\frac{1}{2}\sigma^2 S^2\frac{\partial^2 B}{\partial S^2}+r_g S\frac{\partial B}{\partial S}-(r+r_c)B+h(t)=0. \tag{2.19}$$

Here, U denotes the value of the CB, B the value of the COCB, S the price of underlying stock, r the risk-free interest rate, r_g the growth rate of the stock, r_c the observed credit spread, and $h(t)$ describes various predetermined external flows (in cash or equity) of the derivative such as coupons. These two equations, (2.18) and (2.19), should be solved simultaneously, since, although (2.19) appears to be independent of (2.18), the former is coupled to the latter through its boundary conditions, which are discussed below.

Notice that r_g associated with in the first derivative terms of $\frac{\partial U}{\partial S}$ and $\frac{\partial B}{\partial S}$ and r associated with the terms of U and B of equations (2.18) and (2.19) are different. Recall that the r associated with all these terms is the same in the Black-Scholes equation (2.2). We assume that $r_g = r$ in the risk-neutral world, but r_c is typically not zero, since it is the credit spread for a risky bond, such as a corporate bond, compared to a risk-free bond, such as a government bond. Usually r_c is implied by the normal bonds of the same issuer with similar maturity as the CB, and can be obtained from historical data. To avoid complications, we assume that $h(t)$ represents discrete cash flows only, given by

$$h(t)=\sum K_j\delta(t-t_j), \tag{2.20}$$

where δ is the Dirac delta function and K_j is the cash flow at time t_j. We consider $h(t)$ in more detail in section 2.3.4. Now, let us discuss the final condition and boundary conditions associated with equations (2.18) and (2.19).

We consider the general case described in [TF98]. Let F be the face value of the bond and T be the maturity date. Assume for now that the stock doesn't pay any dividends. The CB pays a fixed coupon amount K at times t_i, and it can be converted at any time to shares of the underlying stock S at a conversion ratio of κ shares per bond, and it pays $F+K$ at expiration if not converted. Furthermore, the CB is callable by the issuer at a price B_c at any time after T_c and puttable by the holder for a cash amount $B_p < B_c$ at any time after T_p. Notice that B_c and B_p are functions of time unless there are no coupon payments made. We elaborate on this issue in section 2.3.4. Now, we have the following conditions.

1. Final conditions at maturity T:

$$\begin{aligned} U(S,T) &= \begin{cases} F+K & \text{if } F+K \geq \kappa S, \\ \kappa S & \text{otherwise,} \end{cases} \\ B(S,T) &= \begin{cases} F+K & \text{if } F+K \geq \kappa S, \\ 0 & \text{otherwise.} \end{cases} \end{aligned} \tag{2.21}$$

   The final conditions are obvious. If $F+K \geq \kappa S$, the holder of the CB should not convert, but instead receive the principal and last coupon at maturity. Thus, both $U(S,T)$ and $B(S,T)$ equal $F+K$. Otherwise the holder should convert the CB to κ shares of the stock. In this case, $U(S,T) = \kappa S$ and $B(S,T) = 0$.

2. Upside constraints due to conversion for $t \in [0,T]$:

$$\begin{cases} U(S,t) \geq \kappa S, \\ B(S,t) = 0 \quad \text{if } \widetilde{U} \leq \kappa S, \end{cases} \tag{2.22}$$

where $\widetilde{U}$ is the value of U if U were not converted to stock. The holder of a CB should convert the CB to stock if the value of the CB falls below κS. Therefore, $U(S,t) \geq \kappa S$ holds for all t. Moreover, if the holder does convert, then the COCB is zero.

3. Upside constraints due to callability by the CB issuer for $t \in [T_c, T]$:

$$\begin{cases} U(S,t) \leq \max(B_c(t), \kappa S), \\ B(S,t) = 0 \quad \text{if } \widetilde{U} \geq B_c(t). \end{cases} \tag{2.23}$$

If $\widetilde{U} \geq B_c(t)$, then the issuer will call the bond. The holder then has to sell it back to the issuer or convert it to shares, depending on which is more beneficial to the holder. Thus, $U(S,t) \leq \max(B_c(t), \kappa S)$ should always be satisfied for $t \in [T_c, T]$. If the issuer calls the bond back, cash flows are assumed to be available to be delivered (since the issuer is favored in the market) and thus there is no credit risk. Therefore, the COCB equals zero.

4. Downside constraints due to puttability by the CB holder for $t \in [T_p, T]$:

$$\begin{cases} U(S,t) \geq B_p(t), \\ B(S,t) = B_p(t) \quad \text{if } \widetilde{U} \leq B_p(t). \end{cases} \tag{2.24}$$

If the value $\widetilde{U}$ of the CB falls below $B_p(t)$, the holder should sell it back to the issuer for $B_p(t)$. Thus, the value of the CB is at least $B_p(t)$. In this downside situation, the issuer is unfavored in market and may have problems to deliver the cash flow. Therefore this cash flow is subject to credit risk and thus belongs to the bond component. Consequently, the COCB equals $B_p(t)$ too.

From the discussion above, we can see that the conditions (2.22), (2.23) and (2.24) are the free boundary conditions. They determine when it is optimal to make an early conversion, call or put.

When $S = 0$ and $S \to \infty$, it is not easy to derive the Dirichlet boundary conditions from financial arguments, but we can derive them from the PDEs themselves. We postpone this discussion until Chapter 3.

Now, we reformulate the problem (2.18)-(2.24) as a LCP. Considering the value U of the CB in two regions: $B_c(t) > \kappa S$ and $B_c(t) \le \kappa S$. If $B_c(t) > \kappa S$, we can reformulate (2.18) and (2.22)-(2.24) as a LCP. If $B_c(t) \le \kappa S$, the issuer will call the CB back and the holder would convert immediately in this situation. Thus, $U = \kappa S$. In summary, U in the TF model satisfies the following relations:

- $B_c(t) > \kappa S$

$$\begin{pmatrix} \mathcal{L}U - r_c B = 0 \\ U \ge \max(B_p(t), \kappa S) \\ U \le B_c(t) \end{pmatrix} \vee \begin{pmatrix} \mathcal{L}U - r_c B \le 0 \\ U = \max(B_p(t), \kappa S) \\ U \le B_c(t) \end{pmatrix} \vee \begin{pmatrix} \mathcal{L}U - r_c B \ge 0 \\ U \ge \max(B_p(t), \kappa S) \\ U = B_c(t) \end{pmatrix}, \tag{2.25}$$

- $B_c(t) \le \kappa S$

$$U = \kappa S. \tag{2.26}$$

The function U also satisfies the final condition (2.29) below. The value of B is determined by the PDE (2.19) and the associated free boundary conditions (2.22)-(2.24), which depend on U. We do not give a LCP formula for B here since in our algorithms, the penalty method is applied to U only, while the artificial component B is calculated solely for the purpose of substituting B into (2.18). We discuss the algorithms in detail in Chapter 4.

From (2.25) and (2.26), the problem also can be summarized as

$$\frac{\partial U}{\partial t} + \frac{1}{2}\sigma^2 S^2 \frac{\partial^2 U}{\partial S^2} + rS\frac{\partial U}{\partial S} - rU - r_c B = 0, \tag{2.27}$$

subject to the constraints

$$\begin{aligned} U &\ge \max(B_p(t), \kappa S), \\ U &\le \max(B_c(t), \kappa S), \end{aligned} \tag{2.28}$$

and the final condition (2.21)

$$U(S,T) = \max(F + K, \kappa S). \tag{2.29}$$

B in (2.27) satisfies

$$\frac{\partial B}{\partial t} + \frac{1}{2}\sigma^2 S^2 \frac{\partial^2 B}{\partial S^2} + rS\frac{\partial B}{\partial S} - (r + r_c)B = 0 \tag{2.30}$$

and the relevant constraints in (2.21), (2.22), (2.23) and (2.24).

2.3.2 AFV Model

Analogous to the TF model, the AFV model [AFV03] also relies on a single-factor Black-Scholes-like analysis. However, it splits the CB in a slightly different way than the TF model, and it incorporates a more sophisticated credit risk model. Instead of adopting a simple credit spread r_c only, as in the TF model, the AFV model introduces a probability, p, of default in the period from t to $t + dt$ conditional on no defaults prior to t (i.e., the hazard rate); a recovery rate, R, of the bond after defaults; and a stock price jump rate, η, when defaults take place. These parameters specify the behavior of the bond and stock in the event of a default. Moreover, the model allows the holder to choose to keep the CB or convert it to shares upon default. Consequently, separation of a CB into the equity and fixed-income components cannot follow the same criteria as in the TF model. The AFV model splits a CB into a bond part B and an equity part C. Detailed descriptions of the two components can be found in [AFV02] and [AFV03]. Similar to the COCB in the TF model, B and C are regarded as hypothetical derivatives of the underlying stock, and can be valued using a Black-Scholes-like model.

Ayache, Forsyth and Vetzal prove that the AFV model follows the self-financing no-arbitrage theory, and they claim that the model is more consistent than the TF model. Their arguments and examples are presented in [AFV03]. Since comparison of the two models from the financial point-of-view is not the focus of this thesis, we restrict our attention to the PDEs and constraints associated with the AFV model.

The model proposes that

$$\frac{\partial U}{\partial t}+\frac{1}{2}\sigma^2S^2\frac{\partial^2 U}{\partial S^2}+(r+p\eta)S\frac{\partial U}{\partial S}-(r+p)U+p\max(\kappa S(1-\eta),RB)=0, \tag{2.31}$$
$$\frac{\partial B}{\partial t}+\frac{1}{2}\sigma^2S^2\frac{\partial^2 B}{\partial S^2}+(r+p\eta)S\frac{\partial B}{\partial S}-(r+p)B+pRB=0, \tag{2.32}$$
$$\frac{\partial C}{\partial t}+\frac{1}{2}\sigma^2S^2\frac{\partial^2 C}{\partial S^2}+(r+p\eta)S\frac{\partial C}{\partial S}-(r+p)C+p\max(\kappa S(1-\eta)-RB,0)=0, \tag{2.33}$$

where U denotes the value of the CB, B the value of its bond component, and C the value of its equity component. From those equations, it is easy to verify that $U = B + C$. In (2.31)-(2.33), r is the risk-free interest rate, as usual, and p is the hazard rate associated with defaults that follow a Poisson process. That is, pdt is the probability of default during the period $[t, t + dt]$. Moreover, R is the recovery rate of the bond. That is, RB is the value of the bond after default. Furthermore, η quantifies how far the stock price declines after a default. More specifically, $S(1 - \eta)$ is the stock price after a default. These parameters can be constants or functions of t and/or S. For simplicity, we assume that they are constants.

If a default occurs, the holder of the bond can choose to either convert the CB to stock at a post-default value of $\kappa S(1-\eta)$ or accept the recovered value of the bond, RB. Thus, RB appears in the PDE (2.32) of the bond component representing the case that the holder accepts the recovered value of the bond, while $\kappa S(1-\eta) - RB$ appears in the PDE (2.33) for the equity component of the CB in this case.

For convenience, we introduce the operator

$$\mathcal{M}=\frac{\partial}{\partial t}+\frac{1}{2}\sigma^2S^2\frac{\partial^2}{\partial S^2}+(r+p\eta)S\frac{\partial}{\partial S}-(r+p). \tag{2.34}$$

Consider the same general case of the CB described in the TF model. Similarly, we divide U into two regions: $B_c(t) > \kappa S$ and $B_c(t) \leq \kappa S$.

- If $B_c(t) > \kappa S$,

$$\begin{pmatrix} \mathcal{M}U + p\max(\kappa S(1-\eta), RB) = 0 \\ U \geq \max(B_p(t), \kappa S) \\ U \leq B_c(t) \end{pmatrix} \vee \begin{pmatrix} \mathcal{M}U + p\max(\kappa S(1-\eta), RB) \leq 0 \\ U = \max(B_p(t), \kappa S) \\ U \leq B_c(t) \end{pmatrix} \vee \begin{pmatrix} \mathcal{M}U + p\max(\kappa S(1-\eta), RB) \geq 0 \\ U \geq \max(B_p(t)), \kappa S) \\ U = B_c(t) \end{pmatrix}. \tag{2.35}$$

- If $B_c(t) \leq \kappa S$,

$$U = \kappa S. \tag{2.36}$$

The final condition is

$$U(S,T) = \max(\kappa S, F + K), \tag{2.37}$$

for all $S \geq 0$ at time T. Relations (2.35) and (2.36) can be expressed in the following way. The value U of the CB is given by the solution to

$$\mathcal{M}U + p\max(\kappa S(1-\eta), RB) = 0, \tag{2.38}$$

subject to the free boundary conditions

$$\begin{aligned} U &\geq \max(B_p(t), \kappa S), \\ U &\leq \max(B_c(t), \kappa S), \end{aligned} \tag{2.39}$$

as well as the final condition (2.37).

The derivation of the free boundary constraints (2.39) and the final condition (2.37) are analogous to the similar constraints in the TF model. The boundary conditions for $S = 0$ and $S \to \infty$ are discussed in Chapter 3. As usual, $B_c(t)$ and $B_p(t)$ stand for the call price and put price, respectively, discussed in the following section. B in (2.38) is obtained by solving (2.32) with its associated constraints. We discuss it in more detail shortly.

Solving the problem (2.35)– (2.37) for U requires solving for B too. Since $U = B + C$, we can obtain the value of U by solving for B and C instead and then summing them. Thus, the pricing problem can be reformulated as

$$\mathcal{M}C + p\max(\kappa S(1-\eta) - RB, 0) = 0, \tag{2.40}$$

subject to the free boundary conditions

$$B + C \leq \max(B_c(t), \kappa S), \tag{2.41}$$

$$B + C \geq \kappa S, \tag{2.42}$$

and the final condition

$$C(S,T) = \max(\kappa S - (F + K), 0); \tag{2.43}$$

and

$$\mathcal{M}B + pRB = 0, \tag{2.44}$$

subject to the free boundary conditions

$$B \leq B_c(t), \tag{2.45}$$

$$B + C \geq B_p(t), \tag{2.46}$$

and the final condition

$$B(S,T) = F + K. \tag{2.47}$$

The functions B and C are artificial derivatives of the underlying stock, and their existence is postulated only for the purpose of incorporating the credit risk into the calculation for the value of the CB, U. Only U is observable in the market and has meaningful constraints. The constraints on B and C are derived from the constraints on U. Therefore, there is no explicit way of splitting the constraints into two independent parts. Rather, the constraints on B and C in the AFV model are coupled together. Therefore, we need to solve for B and C simultaneously.

The equations for B and C can be reformulated as a LCP

$$\begin{aligned}
&\begin{pmatrix} \mathcal{M}C + p\max(\kappa S(1-\eta) - RB, 0) = 0 \\ C \geq \kappa S - B \\ C \leq \max(B_c(t), \kappa S) - B \end{pmatrix} \\
\vee &\begin{pmatrix} \mathcal{M}C + p\max(\kappa S(1-\eta) - RB, 0) \leq 0 \\ C = \kappa S - B \end{pmatrix} \\
\vee &\begin{pmatrix} \mathcal{M}C + p\max(\kappa S(1-\eta) - RB, 0) \geq 0 \\ C = \max(B_c(t), \kappa S) - B \end{pmatrix},
\end{aligned} \tag{2.48}$$

and

$$\begin{pmatrix} \mathcal{M}B + pRB = 0 \\ B \geq B_p(t) - C \\ B \leq B_c(t) \end{pmatrix} \vee \begin{pmatrix} \mathcal{M}B + pRB \leq 0 \\ B = B_p(t) - C \end{pmatrix} \vee \begin{pmatrix} \mathcal{M}B + pRB \geq 0 \\ B = B_c(t) \end{pmatrix}. \tag{2.49}$$

The corresponding final conditions are (2.43) and (2.47).

2.3.3 Connection Between the TF Model and the AFV Model

Both the TF and AFV models are used to price CBs. Thus it is natural to ask how they relate to each other. It is tempting to view the TF model as a special case of the AFV model, since the AFV model supports a more sophisticated credit risk model than the TF model. To clarify this relationship, let us try to fit the TF model into the AFV framework.

Set the credit spread r_c in the TF model to $p(1-R)$ and let $\eta = 0$ in the AFV model.

Then the PDEs from (2.31) and (2.32) for the AFV model reduce to

$$\frac{\partial U}{\partial t} + \frac{1}{2}\sigma^2 S^2 \frac{\partial^2 U}{\partial S^2} + rS\frac{\partial U}{\partial S} - (r+p)U + p\max(\kappa S, RB) = 0, \tag{2.50}$$

$$\frac{\partial B}{\partial t} + \frac{1}{2}\sigma^2 S^2 \frac{\partial^2 B}{\partial S^2} + rS\frac{\partial B}{\partial S} - (r+p)B + pRB = 0. \tag{2.51}$$

The corresponding equations (2.27) and (2.30) for the TF model reduce to

$$\frac{\partial U}{\partial t} + \frac{1}{2}\sigma^2 S^2 \frac{\partial^2 U}{\partial S^2} + rS\frac{\partial U}{\partial S} - rU - p(1-R)B = 0, \tag{2.52}$$

$$\frac{\partial B}{\partial t} + \frac{1}{2}\sigma^2 S^2 \frac{\partial^2 B}{\partial S^2} + rS\frac{\partial B}{\partial S} - (r + p(1-R))B = 0, \tag{2.53}$$

Notice that the PDEs (2.51) and (2.53) for B are equivalent, but the PDEs (2.50) and (2.52) for U are different. The reason is obvious: the splitting strategies in the TF model and the AFV model are different. Moreover, the AFV model assumes the CB holder has the option of converting to stock or retaining the bond at the reduced value of RB upon default, while the TF model doesn't. Further differences arise from slight differences in the conditions for B. However, the constraints in both models for both U, (2.28) and (2.39), are the same, as well as the final conditions (2.29) and (2.37). Hence we can expect that the solutions for the TF and AFV models to be similar in the special case $r_c = p(1-R)$ and $\eta = 0$. We perform some numerical experiments in chapter 5 to explore this relationship further.

2.3.4 Issues Concerning Cash Flows

In both the TF and AFV pricing models for CBs, we assume that cash flows are discrete. Generally, discrete cash flows are referred to as coupon payments and modeled with dirac Delta functions as in (2.20). We describe here in more detail how to handle coupon payments.

Consider the forward time horizontal line depicted in Figure 2.2. Let $t_1, t_2, \ldots, t_n$ be the coupon payment dates, t_i^- be the instant of time immediately before the i^{th} coupon payment, and t_i^+ be the instant of time immediately after the i^{th} coupon payment. If K_i

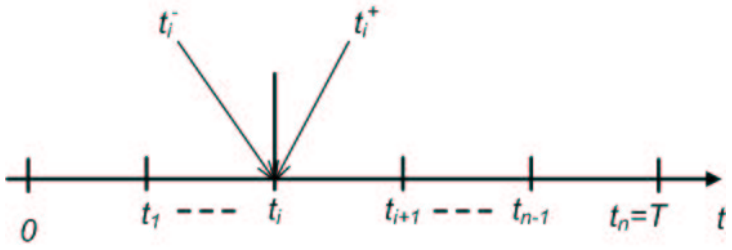


Figure 2.2: Coupon payments

is the amount of the i^{th} coupon payment, then U must satisfy

$$U(S, t_i^-) = U(S, t_i^+) + K_i \tag{2.54}$$

to enforce continuity in the value of the bond and cash flows in the portfolio. If (2.54) did not hold, there would be an obvious arbitrage opportunity. Similarly, we require

$$B(S, t_i^-) = B(S, t_i^+) + K_i \tag{2.55}$$

for the bond component B. On the other hand, the equity component C satisfies

$$C(S, t_i^-) = C(S, t_i^+) \tag{2.56}$$

since the equity component of the CB is unaffected by the coupon payment.

Coupon payments also affect the call and put prices through accrued interest. Accrued interest is the amount of interest deemed to be earned since the last coupon payment. Commonly call and put prices are quoted as the clean prices without including accrued interest, denoted as B_{cc} and B_{pc}, respectively. When computing the bond price, we should use the call and put prices that include accrued interest rather than the clean prices. We refer to these modified prices as the dirty call price B_c and the dirty put price B_p, respectively.

Let $AccI(t)$ be the accrued interest associated with the pending coupon payment. It is standard practice to take

$$AccI(t) = K_i \frac{t - t_{i-1}}{t_i - t_{i-1}}, \tag{2.57}$$

where t is the current time, t_i and t_{i-1} are times of the next pending and previous coupon payment dates, respectively, and $t_{i-1} \leq t < t_i$. Thus, $AccI(t)$ has a "sawtooth" shape, growing from zero after one coupon payment date to K_i at the instant before the next payment date. Therefore, the dirty prices B_c and B_p are

$$B_c(t) = B_{cc} + AccI(t), \tag{2.58}$$

$$B_p(t) = B_{pc} + AccI(t). \tag{2.59}$$

Assuming the clean prices B_{cc} and B_{pc} are constants, B_c and B_p are also "sawtooth" shaped, as depicted in Figure 2.3 for B_c.

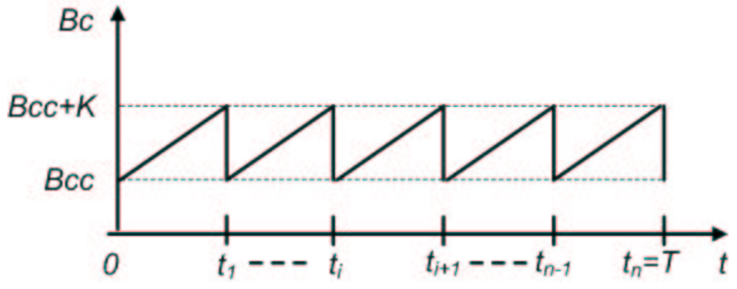


Figure 2.3: Dirty call price including accrued interest as a function of time

Usually we compute the CB price backwards in time, starting from the final condition at time T. Let $\tau = T - t$ and $\tau_i = T - t_i$. Then equation (2.57) for the accrued interest can be reformulated as

$$AccI(\tau) = K_{n-i} \frac{\tau_{i+1} - \tau}{\tau_{i+1} - \tau_i}, \tag{2.60}$$

and equations (2.54), (2.55) and (2.56) become

$$U(S, \tau_i^+) = U(S, \tau_i^-) + K_{n-i}. \tag{2.61}$$

$$B(S, \tau_i^+) = B(S, \tau_i^-) + K_{n-i}. \tag{2.62}$$

$$C(S, \tau_i^+) = C(S, \tau_i^-), \tag{2.63}$$

respectively.

Chapter 3

The Finite Difference Methods

As discussed in Chapter 2, the Black-Scholes analysis for pricing vanilla European or American call or put options leads to the Black-Scholes PDE, a parabolic partial differential equation in time, t, and the stock price, S. Both the TF and AFV models for pricing convertible bonds (CBs) require solving a pair of coupled Black-Scholes-like two-dimensional parabolic PDEs. Final conditions and boundary conditions are also provided to guarantee a unique solution. For simple pricing problems, such as those for European call or put options, a closed-formed analytic formula is relatively easy to derive, but for free boundary problems, such as those for American put options and convertible bonds, no closed-form analytic solution is known. Thus effective numerical solutions are required.

Finite difference methods (FDMs) are a well-known class of numerical method for solving PDEs. In this chapter, we explore the use of FDMs to solve the PDEs associated with the TF and AFV models for pricing CBs. More specifically, we begin with a generalized Black-Scholes-like PDE, and consider some transformations to simplify the problem. After applying finite difference space and time discretization schemes to that transformed PDE, we obtain a general linear system of equations. In section 3.4, we discuss far-field problems and form linear boundary conditions for the far-field. At the end of the chapter, for each of the models for pricing European call options, American put options and CBs, we generate a constrained linear system associated with a FDM for the model.

3.1 General Black-Scholes PDE

The two-dimensional parabolic PDEs given in the previous chapter are each of the form

$$\frac{\partial V}{\partial t} + a_1 S^2 \frac{\partial^2 V}{\partial S^2} + a_2 S \frac{\partial V}{\partial S} + a_3 V + f(S,t) = 0, \quad S \in [0,\infty), \quad t \in [0,T]. \tag{3.1}$$

Each individual PDE in the pricing models can be distinguished by substituting the particular coefficients a_1, a_2, a_3, and auxiliary function $f(S,t)$. More specifically, each particular pricing model is determined by the following choice of parameters.

1. European and American option pricing model (equation (2.2)):

$$a_1 = \frac{\sigma^2}{2}, \quad a_2 = r, \quad a_3 = -r, \quad f(S,t) = 0.$$

2. The TF model (equations (2.27) and (2.30)):

$$\begin{array}{lllll} \text{U:} & a_1 = \frac{\sigma^2}{2}, & a_2 = r, & a_3 = -r, & f(S,B,t) = -r_c B; \\ \text{B:} & a_1 = \frac{\sigma^2}{2}, & a_2 = r, & a_3 = -(r+r_c), & f(S,B,t) = 0. \end{array}$$

3. The AFV model (equations (2.31), (2.32) and (2.33)):

$$\begin{array}{ll} \text{U:} & a_1 = \frac{\sigma^2}{2}, \quad a_2 = r + p\eta, \quad a_3 = -(r+p), \\ & f(S,B,t) = p\max(\kappa S(1-\eta), RB); \\ \text{C:} & a_1 = \frac{\sigma^2}{2}, \quad a_2 = r + p\eta, \quad a_3 = -(r+p), \\ & f(S,B,t) = p\max(\kappa S(1-\eta) - RB, 0); \\ \text{B:} & a_1 = \frac{\sigma^2}{2}, \quad a_2 = r + p\eta, \quad a_3 = -(r+p(1-R)), \\ & f(S,B,t) = 0, \end{array}$$

From the general PDE (3.1), we can conveniently derive a numerical solution technique. Substituting each parameter set above into that general system and combining the associated free boundary constraints, we can generate the individual constrained linear systems for each pricing problem.

3.2 Variable Transformation for PDEs

Usually a pricing problem is expressed as a PDE with a final condition $V(S,T)$. To integrate such an equation, we start with the final condition and solve the equation backward in time. To many readers, it is more natural to introduce the time transformation $\tau = T - t$ and integrate forward in τ.

To this end, we let $\tau = T - t$ and rewrite equation (3.1) as

$$\frac{\partial V}{\partial \tau} = a_1 S^2 \frac{\partial^2 V}{\partial S^2} + a_2 S \frac{\partial V}{\partial S} + a_3 V + f(S,\tau), \quad S \in [0,\infty), \quad \tau \in [0,T]. \tag{3.2}$$

Defining the operator

$$\bar{\mathcal{L}} = a_1 S^2 \frac{\partial^2}{\partial S^2} + a_2 S \frac{\partial}{\partial S} + a_3, \tag{3.3}$$

we can rewrite equation (3.2) in the more compact form

$$\frac{\partial V}{\partial \tau} = \bar{\mathcal{L}} V + f(S,\tau). \tag{3.4}$$

3.3 Discretization

Next, we discretize equation (3.2). The idea underlying finite difference methods is to approximate partial derivatives with finite difference expressions. We call this procedure discretization. Two kinds of discretizations are employed on equation (3.2): space discretization and time discretization.

3.3.1 Space Discretization

Non-Uniformly-Spaced Discretizations

We define an array of unequally spaced grid points $S = \{S_0, S_1, \dots, S_n\}$ with $h_{i,1} = S_{i+1} - S_i$ and $h_{i,-1} = S_i - S_{i-1}$, and introduce the notation $V_i = V(S_i, \tau)$. Figure 3.1 shows the space-steps in the non-uniform grid.

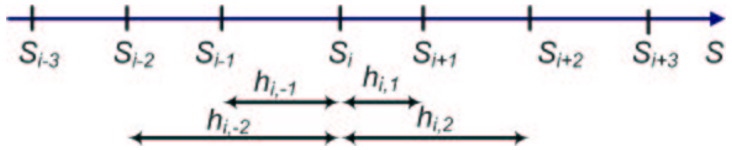


Figure 3.1: A non-uniform grid

For convenience, we write $h_{i,1}$ as h_1, and $h_{i,-1}$ as h_{-1}. Then the second derivative with respect to S is discretized as

$$\frac{\partial^2 V_i}{\partial S^2} = c_1 V_{i-1} + c_2 V_i + c_3 V_{i+1} - TE_1, \quad i = 1, \dots, n-1, \tag{3.5}$$

where

$$\begin{aligned} c_1 &= \frac{2}{h_{-1}(h_{-1}+h_1)}, \\ c_2 &= \frac{-2}{h_{-1}h_1} = -(c_1 + c_3), \\ c_3 &= \frac{2}{h_1(h_{-1}+h_1)}; \end{aligned}$$

and the first derivative is discretized as

$$\frac{\partial V_i}{\partial S} = d_1 V_{i-1} + d_2 V_i + d_3 V_{i+1} - TE_2, \quad i = 1, \dots, n-1, \tag{3.6}$$

where

$$\begin{aligned} d_1 &= \frac{-h_1}{h_{-1}(h_{-1}+h_1)}, \\ d_2 &= \frac{h_1 - h_{-1}}{h_{-1}h_1} = -(d_1 + d_3), \\ d_3 &= \frac{h_{-1}}{h_1(h_{-1}+h_1)}. \end{aligned}$$

TE_1 and TE_2 are the respective truncation errors: the terms in the Taylor series expansion of the finite difference expressions that must be subtracted to recover the derivatives exactly. The finite difference expression in equation (3.5) is the first-order central difference approximation and that in equation (3.6) is the second-order central difference approximation, meaning that the leading order terms in TE_1 and TE_2 are proportional to h and h^2, respectively. Usually we denote this by writing $TE_1 = O(h)$ and $TE_2 = O(h^2)$. Actually TE_1 in (3.5) can be bounded in terms of $(h_1 - h_{-1}) + \max(h_i)^2$. If $h_{-1} = h_1 = h$, then TE_1 is proportional to h^2 and the central difference approximation for the second derivative, $\frac{\partial^2 V}{\partial S^2}$, becomes second order. That is, TE_2 in (3.6) is proportional to $\max(h_i)^2$ if $h_1 = h_{-1}$.

We refer to these discretizations as three-point non-uniformly spaced discretizations. The $V_i, i = 1, \ldots, n-1$ are referred to as interior points. The V_0 and V_n are referred to as boundary points, corresponding to $S = 0$ and $S = \infty$, respectively. Here, we restrict our attention to the interior points, and leave the boundary points to section 3.4.

We also give another two difference approximations for the first derivative – the forward and backward finite difference approximations, respectively:

$$\frac{\partial V_i}{\partial S} = \frac{V_{i+1} - V_i}{h_1} + O(h_{max}), \tag{3.7}$$

$$\frac{\partial V_i}{\partial S} = \frac{V_i - V_{i-1}}{h_{-1}} + O(h_{max}). \tag{3.8}$$

These two finite difference approximations are first-order accurate.

Uniformly-Spaced Discretizations

Uniformly-spaced discretizations use equally spaced grids. By setting $h_{-1} = h_1 = h$, (3.5) becomes

$$\frac{\partial^2 V_i}{\partial S^2} = c_1' V_{i-1} + c_2' V_i + c_3' V_{i+1} + O(h^2), \quad i = 1, \ldots, n-1, \tag{3.9}$$

where

$$c_1' = \frac{1}{h^2}, \quad c_2' = \frac{-2}{h^2} = -(c_1' + c_3'), \quad \text{and } c_3' = \frac{1}{h^2}.$$

Similarly, setting $h_{-1} = h_1 = h$ in (3.6) yields

$$\frac{\partial V_i}{\partial S} = d_1' V_{i-1} + d_2' V_i + d_3' V_{i+1} + O(h^2), \quad i = 1, \ldots, n-1, \tag{3.10}$$

where

$$d_1' = \frac{-1}{2h}, \quad d_2' = 0 = -(d_1' + d_3'), \quad \text{and } d_3' = \frac{1}{2h}.$$

In this case, both finite difference expressions (3.9) and (3.10) are second-order central differences, i.e., their truncation errors can be bounded in terms of h^2.

Space Discretization for $\bar{\mathcal{L}}V$

Now we are ready to derive the discretization formula for $\bar{\mathcal{L}}V$. Ignoring the discretization errors, we substitute the the finite difference approximates for the first and second partial derivatives, equations (3.5) and (3.6), into $\bar{\mathcal{L}}V_i (i = 1, \ldots, n-1)$, where $\bar{\mathcal{L}}V_i$ denotes $\bar{\mathcal{L}}V(S_i, \tau)$, to get

$$\begin{aligned}\bar{\mathcal{L}}V_i &= a_1 S_i^2 (c_1 V_{i-1} + c_2 V_i + c_3 V_{i+1}) + a_2 S_i (d_1 V_{i-1} + d_2 V_i + d_3 V_{i+1}) + a_3 V_i \\ &= (a_1 c_1 S_i^2 + a_2 d_1 S_i) V_{i-1} + (a_1 c_2 S_i^2 + a_2 d_2 S_i + a_3) V_i + (a_1 c_3 S_i^2 + a_2 d_3 S_i) V_{i+1} \\ &= (a_1 c_1 S_i^2 + a_2 d_1 S_i) V_{i-1} + \big(a_3 - a_1(c_1 + c_3) S_i^2 - a_2(d_1 + d_3) S_i\big) V_i + (a_1 c_3 S_i^2 + a_2 d_3 S_i) V_{i+1}.\end{aligned}$$

Let

$$\alpha_i = a_1 c_1 S_i^2 + a_2 d_1 S_i, \tag{3.11}$$

$$\beta_i = a_1 c_3 S_i^2 + a_2 d_3 S_i. \tag{3.12}$$

We have the non-uniformly-spaced discretization formula

$$\bar{\mathcal{L}}V_i = \alpha_i V_{i-1} + \big(a_3 - (\alpha_i + \beta_i)\big) V_i + \beta_i V_{i+1}, \quad i = 1, \ldots, n-1. \tag{3.13}$$

Recall that the finite difference approximation to the second derivative $\frac{\partial^2 V}{\partial S^2}$ is first order if $h_{-1} \neq h_1$. Thus so is (3.13). If the space discretization is uniform, i.e., $h_{-1} = h_1 = h$, then (3.13) becomes second order. If we apply uniform spacing at some points and non-uniform spacing at other points, then first-order accuracy should be expected overall.

However, experimental results show that, if we employ non-uniform spacing at very few points and uniform spacing at all other points, then the convergence of (3.13) approaches second order.

3.3.2 Time Discretization

Applying the spacial discretizations discussed in the previous subsection transforms equation (3.4) into a system of ordinary differential equations with the unknowns $V_i(\tau) = V(S_i, \tau), i = 0, \ldots, n$. More specifically, by substituting equation (3.13) into equation (3.4), we obtain the system of ordinary differential equations

$$\frac{d\mathbf{V}}{d\tau} = \mathcal{A}\mathbf{V} + \mathcal{F} + \mathcal{E}, \tag{3.14}$$

where $\mathbf{V} = (V_0, V_1, \ldots, V_n)^T$, $\mathcal{A}$ is the coefficient matrix, $\mathcal{F}$ is the vector representing the forcing terms $f(S, \tau)$, and $\mathcal{E}$ is the truncation error vector. If the error terms in the vector $\mathcal{E}$ is ignored, an approximation to the solution of equation (3.2) can be obtained from

$$\frac{d\mathbf{V}}{d\tau} = \mathcal{A}\mathbf{V} + \mathcal{F}. \tag{3.15}$$

The next step is to solve (3.15) numerically. To accomplish this, we need to discretize the time derivative $\frac{\partial V}{\partial \tau}$. To this end, define an array of equally spaced grid points in the time dimension $\tau = \{\tau_0 = 0, \tau_1, \ldots, \tau_m = T\}$ for the interval $[0, T]$ with the time step $\Delta\tau_j = \tau_{j+1} - \tau_j$. The time step, $\Delta\tau_j$ could vary from step to step, but we use a constant time throughout this thesis, so we usually drop the subscript j from $\Delta\tau_j$ and refer to it as $\Delta\tau$. To indicate time dependency, we use the convenient notation $V_i^j = V(S_i, \tau_j)$ and $f_i^j = f(S_i, \tau_j)$.

The θ-timestepping discretization scheme applied to equation (3.4) for $i = 1, \ldots, n-1$

and $j = 0, \dots, m-1$ can be written as

$$\begin{aligned}
& \frac{V_i^{j+1} - V_i^j}{\Delta\tau} = \theta\bar{\mathcal{L}}V_i^{j+1} + (1-\theta)\bar{\mathcal{L}}V_i^j + \theta f_i^{j+1} + (1-\theta)f_i^j \\
\Longrightarrow \quad & V_i^{j+1} - \theta\Delta\tau\big(\alpha_i V_{i-1}^{j+1} + \big(a_3 - (\alpha_i + \beta_i)\big)V_i^{j+1} + \beta_i V_{i+1}^{j+1}\big) \\
= \quad & V_i^j + (1-\theta)\Delta\tau\big(\alpha_i V_{i-1}^j + \big(a_3 - (\alpha_i + \beta_i)\big)V_i^j + \beta_i V_{i+1}^j\big) + \theta\Delta\tau f_i^{j+1} + (1-\theta)\Delta\tau f_i^j.
\end{aligned}$$

Writing this in vector-matrix notation, we get

$$\big(\mathbf{I} - \theta\Delta\tau\mathbf{M}\big)V^{j+1} = \big(\mathbf{I} + (1-\theta)\Delta\tau\mathbf{M}\big)V^j + \big(\theta f^{j+1} + (1-\theta)f^j\big)\Delta\tau, \tag{3.16}$$

where $\mathbf{I}$ is the identity matrix and

$$\mathbf{M} = \begin{pmatrix}
\ell & 0 & 0 & 0 & \dots & 0 \\
\alpha_1 & a_3 - (\alpha_1 + \beta_1) & \beta_1 & 0 & \dots & 0 \\
0 & \alpha_2 & a_3 - (\alpha_2 + \beta_2) & \beta_2 & \dots & 0 \\
\vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\
0 & 0 & \dots & \alpha_{n-1} & a_3 - (\alpha_{n-1} + \beta_{n-1}) & \beta_{n-1} \\
0 & 0 & \dots & \gamma_3 & \gamma_2 & \gamma_1
\end{pmatrix}.$$

Here, $\mathbf{M}$ is a sparse matrix, nearly tridiagonal except for γ_3 in the last row. The undefined parameters ℓ, γ_1, γ_2 and γ_3 are associated with the boundary conditions and are discussed in section 3.4.

Depending on the value of θ, we have

- $\theta = 0$, the explicit Euler scheme,
- $\theta = 1$, the implicit Euler scheme,
- $\theta = \frac{1}{2}$, the Crank-Nicolson scheme (or equivalently, the trapezoid rule).

The explicit Euler scheme is conditionally stable with truncation error in time $O(\Delta\tau)$. The implicit Euler scheme is unconditionally stable with truncation error in time also $O(\Delta\tau)$. The more appealing Crank-Nicolson scheme incorporates both explicit and implicit features. It is unconditionally stable with truncation error in time $O(\Delta\tau^2)$. In this thesis, we focus on the Crank-Nicolson scheme, but we use the implicit Euler scheme in the Rannacher smoothing technique discussed in Chapter 5.

3.4 Special Issues

3.4.1 Far Field

Problem (3.2) is posed on the domain $[0, \infty)$. Obviously, an infinite grid cannot be represented in the computer, so we must truncate this domain to $[0, S_{max}]$ and replace the boundary condition at $S = \infty$ by one at $S = S_{max}$. To minimize the deleterious effects of this truncation, we must let S_{max} be large enough and we must carefully choose an appropriate boundary condition at $S = S_{max}$.

In [KN00] a good choice for S_{max} is suggested after a careful analysis. In the case of a European option, the far field boundary satisfies

$$S_{max} > E\mathrm{e}^{\sqrt{2\sigma^2 T|\ln(tol)|}}, \tag{3.17}$$

where E is the strike price and tol is the computing tolerance. An alternative suggested in [WFV04] is

$$S_{max} > E\mathrm{e}^{(r-\frac{\sigma^2}{2})T+\sigma\mu\sqrt{T}}, \tag{3.18}$$

where $\mu = 3$ is the usual choice for a European option. We may choose a bigger value for μ for more complicated problems to push S_{max} further away from the region of interest. For American options and convertible bonds, we suggest setting

$$S_{max} = \max(\phi X, X\mathrm{e}^{\sqrt{2\sigma^2 T|\ln(tol)|}}, X\mathrm{e}^{(r-\frac{\sigma^2}{2})T+\sigma\mu\sqrt{T}}), \tag{3.19}$$

where X is the strike price in the case of an American option or the face value of the bond in the case of a convertible bond. Typically, $\phi \geq 4$ is chosen, as in the experiments performed in [Lee03] and [Li05]. More specifically, $\phi = 5$ is chosen in [Li05] for both American put options and convertible bonds, and $\phi = 4$ is chosen for American put options in [Lee03].

3.4.2 Boundary Conditions

After truncating the domain to $[0, S_{max}]$, we need to impose the boundary conditions at the two ends $S = 0$ and $S = S_{max}$. The boundary condition at $S = 0$ is easily determined. We can derive a Dirichlet condition though financial arguments as we did for European call options in Chapter 2. Alternatively, we can obtain it implicitly by setting $S = 0$ in equation (3.2), which leads to

$$\frac{\partial V_0}{\partial \tau} = a_3 V_0 + f(0, \tau). \tag{3.20}$$

Equation (3.20) is the implicit boundary condition for $S = 0$. We can discretize it in time and incorporate it into (3.16), as we explain in more detail below.

The choice of an appropriate boundary condition at $S = S_{max}$ plays an important role in obtaining an accurate numerical solution. A good choice minimizes the errors caused by the domain truncation. A Dirichlet condition at $S = S_{max}$ may be derived from financial arguments, but for many complicated problems it demands more knowledge about the derivatives and thus is difficult to obtain. If the payoff is almost linear in S and the value of derivative is driven by a well-behaved movement of the underlying asset, then the derivative value V is asymptotically linear in S as $S \to \infty$ [WFV04]. Thus, we can obtain a linear boundary condition for S_{max} based on the assumption that V is approximately linear in S as $S \to \infty$. Thus, $\frac{\partial^2 V}{\partial S^2} \approx 0$ for large S. Setting $\frac{\partial^2 V}{\partial S^2} = 0$ in equation (3.2) yields

$$\frac{\partial V_n}{\partial \tau} = a_2 S_n \frac{\partial V_n}{\partial S} + a_3 V_n + f_n. \tag{3.21}$$

As in the previous sections, we can incorporate a discretization of (3.21) into the system (3.16). In Chapter 5, we present experimental results to show that this approach leads to a good approximation to the exact boundary condition for European options. We can use the same approach to derive boundary conditions at $S = S_{max}$ for convertible bonds. It is not clear how to derive appropriate Dirichlet boundary conditions from financial arguments in this more complicated case.

In the following we discretize equations (3.20) and (3.21) and give formulas for ℓ and γ_1, γ_2 and γ_3 in the coefficient matrix $\mathbf{M}$ in (3.16).

Case of $S = 0$: the left end

If $S = 0$, then from the definition of operator (3.3) and (3.20),

$$\bar{\mathcal{L}}V_0 = a_3 V. \tag{3.22}$$

In the above equation, no derivatives with respect to S exist. Thus, we consider the time discretization only. Using the θ-method, we get

$$\begin{aligned} \frac{V_0^{j+1} - V_0^j}{\Delta\tau} &= \theta\bar{\mathcal{L}}V_0^{j+1} + (1-\theta)\bar{\mathcal{L}}V_0^j + \theta f_0^{j+1} + (1-\theta) f_0^j \\ \Rightarrow \quad V_0^{j+1} - \theta\Delta\tau a_3 V_0^{j+1} &= V_0^j + (1-\theta)\Delta\tau a_3 V_0^j + \left(\theta f_0^{j+1} + (1-\theta) f_0^j\right)\Delta\tau. \end{aligned}$$

Thus, we take

$$\ell = a_3 \tag{3.23}$$

in the matrix $\mathbf{M}$ associated with equation (3.16).

Case of $S \to \infty$: the right end

We truncate S_∞ to S_{max} and assume that V is approximately linear in S at S_{max}. That is,

$$\frac{\partial^2 V_n}{\partial S^2} \approx 0. \tag{3.24}$$

Setting $\frac{\partial^2 V_n}{\partial S^2} = 0$, we obtain

$$\bar{\mathcal{L}}V_n = a_2 S \frac{\partial V_n}{\partial S} + a_3 V_n. \tag{3.25}$$

Only the first derivative appears in $\bar{\mathcal{L}}V_n$. Since S_n is at the right end of the spatial domain, we use a second-order backward difference scheme to discretize $\frac{\partial V_n}{\partial S}$:

$$\frac{\partial V_n}{\partial S} \approx e_1 V_n + e_2 V_{n-1} + e_3 V_{n-2}, \tag{3.26}$$

where

$$e_1 = \frac{3}{2h} = -(e_2 + e_3), \quad e_2 = -\frac{2}{h}, \quad e_3 = \frac{1}{2h}.$$

If we use the first-order backward difference scheme, (3.7), then

$$e_1 = \frac{1}{h} = -(e_2 + e_3), \quad e_2 = -\frac{1}{h}, \quad e_3 = 0.$$

Discretizing the operator $\bar{\mathcal{L}}V_n$, we have

$$\begin{aligned} \bar{\mathcal{L}}V_n &= a_2 S_n(e_1 V_n + e_2 V_{n-1} + e_3 V_{n-2}) + a_3 V_n \\ &= \gamma_1 V_n + \gamma_2 V_{n-1} + \gamma_3 V_{n-2}, \end{aligned} \tag{3.27}$$

where

$$\gamma_1 = a_2 e_1 S_n + a_3 = a_3 - (\gamma_2 + \gamma_3), \quad \gamma_2 = a_2 e_2 S_n, \quad \gamma_3 = a_2 e_3 S_n. \tag{3.28}$$

Incorporating the formula above into the θ-method, we have

$$\begin{aligned} &\frac{V_n^{j+1} - V_n^j}{\Delta\tau} = \theta\bar{\mathcal{L}}V_n^{j+1} + (1-\theta)\bar{\mathcal{L}}V_n^j + \theta f_n^{j+1} + (1-\theta)f_n^j \\ \Longrightarrow\quad & V_n^{j+1} - \theta\Delta\tau(\gamma_1 V_n^{j+1} + \gamma_2 V_{n-1}^{j+1} + \gamma_3 V_{n-2}^{j+1}) \\ =\quad & V_n^j + (1-\theta)\Delta\tau(\gamma_1 V_n^j + \gamma_2 V_{n-1}^j + \gamma_3 V_{n-2}^j) + \big(\theta f_n^{j+1} + (1-\theta)f_n^j\big)\Delta\tau \end{aligned}$$

Now, we rewrite the matrix $\mathbf{M}$ associated with equation (3.16)

$$\mathbf{M} = \begin{pmatrix} a_3 & 0 & 0 & 0 & \cdots & 0 \\ \alpha_1 & a_3 - (\alpha_1 + \beta_1) & \beta_1 & 0 & \cdots & 0 \\ 0 & \alpha_2 & a_3 - (\alpha_2 + \beta_2) & \beta_2 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \alpha_{n-1} & a_3 - (\alpha_{n-1} + \beta_{n-1}) & \beta_{n-1} \\ 0 & 0 & \cdots & a_2 e_3 S_n & a_2 e_2 S_n & a_2 e_1 S_n + a_3 \end{pmatrix}. \tag{3.29}$$

3.4.3 A Few Additional Considerations

The numerical solution to equation (3.2) requires solving (3.16) at each time step. We repeat equation (3.16) here for convenience:

$$\big(\mathbf{I} - \theta\Delta\tau\mathbf{M}\big)V^{j+1} = \big(\mathbf{I} + (1-\theta)\Delta\tau\mathbf{M}\big)V^j + \big(\theta f^{j+1} + (1-\theta)f^j\big)\Delta\tau.$$

It can be rewritten in more compact notation as

$$\mathbf{A}\mathbf{x} = \mathbf{b}, \tag{3.30}$$

where $\mathbf{x}$ is an unknown vector, $\mathbf{A}$ is a square nearly-tridiagonal matrix, and $\mathbf{b}$ is a known vector. More specifically,

$$\mathbf{x} = V^{j+1}, \quad \mathbf{A} = \big(\mathbf{I} - \theta\Delta\tau\mathbf{M}\big),$$
$$\mathbf{b} = \big(\mathbf{I} + (1-\theta)\Delta\tau\mathbf{M}\big)V^j + \big(\theta f^{j+1} + (1-\theta)f^j\big)\Delta\tau.$$

Diagonal Dominance

Consider the system of linear equations (3.30). The matrix $\mathbf{A}$ must satisfy some properties to guarantee that the system (3.30) has a stable and accurate solution. Diagonal dominance is one such desirable property. We give the definition of diagonal dominance below.

Definition 1. *A matrix is diagonally dominant if*

$$|a_{ii}| \geq \sum_{j=1, i\neq j}^{n} |a_{ij}| \tag{3.31}$$

for each value of i. A row is strictly diagonally dominant if the inequality in (3.31) *is a strict inequality and a matrix is strictly diagonally dominant if each row is strictly diagonally dominant.*

A second important property of a matrix is reducibility. Before discussing this property, we introduce the concept of a matrix permutation. By a permutation of a matrix A, we mean a simultaneous permutation of the rows and columns of the matrix, i.e., a_{ij} is replaced by $a_{\Lambda(i),\Lambda(j)}$ for some permutation bijective operator Λ.

Definition 2. *A matrix A is reducible if there is a permutation Λ under which A has the structure*

$$\begin{pmatrix} A_1 & 0 \\ A_{12} & A_2 \end{pmatrix}, \tag{3.32}$$

where A_1 and A_2 are square matrices. A matrix is irreducible if it is not reducible.

Definition 3. *A matrix A is irreducibly (row) diagonally dominant if it is irreducible and diagonally dominant with strict diagonal dominance in at least one row.*

It is well-known that an irreducibly (row) diagonally dominant matrix is nonsingular [Str04]. We want our discretizations of (3.2) to be such that the matrix **A** associated with the discretization is irreducibly (row) diagonally dominant. If **A** is not irreducibly diagonally dominant, a numerical solution to equation (3.2) may exhibit highly undesirable properties, such as large oscillations.

M-Matrix

Consider the general PDE (3.2)

$$\frac{\partial V}{\partial \tau} = a_1 S^2 \frac{\partial^2 V}{\partial S^2} + a_2 S \frac{\partial V}{\partial S} + a_3 V + f(S, \tau).$$

It is a convection-diffusion equation. If a_2 is large, we say (3.2) is a convection-dominated problem. In our pricing models, convection-dominated problems arise if the volatility, σ, is small and the interest rate, r, is big. If (3.2) is a convection-dominated problem, the numerical solution may have small spurious oscillations. These oscillations are frequently more pronounced in the delta associated with the solution, thus making the numerical method unsuitable for delta hedging applications. If **A** is an M-matrix, we can guarantee that the numerical solution associate with (3.2) is stable without spurious oscillations [F$^+$00].

A square matrix **A** is said to be an M-Matrix if it is non-singular, its inverse is non-negative and its off-diagonal elements are less than or equal to 0:

$$\left.\begin{aligned} \mathbf{A}^{-1} &\geq 0 \\ a_{ij} \leq 0 \quad i &\neq j \end{aligned}\right\} \tag{3.33}$$

Sufficient conditions for the inverse of a matrix to be non-negative are given in [F$^+$00].

Lemma 1. *Suppose that the matrix* $\mathbf{A}$ *is irreducibly diagonally dominant and*

$$\left.\begin{aligned} a_{ij} \leq 0, \quad i \neq j \\ a_{ii} \geq 0 \end{aligned}\right\}$$

Then $\mathbf{A}$ *is non-singular and its inverse is strictly positive.*

Stability

We can rewrite the simple matrix formulation

$$\big(\mathbf{I} - \theta\Delta\tau\mathbf{M}\big)V^{j+1} = \big(\mathbf{I} + (1-\theta)\Delta\tau\mathbf{M}\big)V^{j}, \tag{3.34}$$

as

$$V^{j+1} = \big(\mathbf{I} - \theta\Delta\tau\mathbf{M}\big)^{-1}\big(\mathbf{I} + (1-\theta)\Delta\tau\mathbf{M}\big)V^{j}, \tag{3.35}$$

or

$$V^{j+1} = \mathbf{B}V^{j}, \tag{3.36}$$

where

$$\mathbf{B} = \big(\mathbf{I} - \theta\Delta\tau\mathbf{M}\big)^{-1}\big(\mathbf{I} + (1-\theta)\Delta\tau\mathbf{M}\big). \tag{3.37}$$

If V^0 is the initial data, then after j timesteps we have

$$V^{j} = \mathbf{B}^{j}V^{0}. \tag{3.38}$$

Now, we define the following notions of stability [WFV04]:

- $\mathbf{B}$ is strictly stable if $\|\mathbf{B}^j\| \leq 1, \quad \forall j, n > 0$,
- $\mathbf{B}$ is strongly stable if $\|\mathbf{B}^j\| \leq C, \quad \forall j, n > 0$,
- $\mathbf{B}$ is algebraically stable if $\|\mathbf{B}^j\| \leq j^p n^l C, \quad \forall j, n > 0$,

where $C, p, l \geq 0$ are constants independent of j and n, where n is the dimension of $\mathbf{B}$.

If $\mathbf{A} = \mathbf{I} - \theta\Delta\tau\mathbf{M}$ is an M-matrix, it has been proved in [WFV04] that $\mathbf{B}$ of (3.37) is strictly stable for the implicit Euler method, but $\mathbf{B}$ is not guaranteed to be strongly stable for the Crank-Nicolson method, since $\mathbf{A}$ may not be an M-matrix in this case. Strong stability is a sufficient and necessary condition for the convergence of a consistent discretization for all initial data (according to the Lax Equivalence Theorem [Str04]). Equation (3.16), which is the focus of our interest, includes the additional term $\big(\theta f^{j+1} + (1-\theta)f^j\big)\Delta\tau$, and so it is not strictly of the form (3.34). Thus it is not clear that the analysis above applies to (3.16). However, numerical experiments indicate that choosing the discretization so that $\mathbf{A}$ is an M-matrix, or nearly an M-matrix, at each time step, aids the stability and convergence of the numerical method.

3.5 The Constrained Linear System for Each Pricing Model

In this section, we generate the linear system for each pricing model by substituting the particular parameters that we gave in section 3.1 and considering the free boundary conditions and other special issues.

As in the previous sections, $S = \{S_0 = 0, S_1, \ldots, S_n = S_{max}\}$ and $\tau = \{\tau_0 = 0, \tau_1, \ldots, \tau_m = T\}$. Moveover, for $j = 0, \ldots, m$, define

$$\begin{aligned} V^j &= (V_0^j \quad V_1^j \quad \ldots \quad V_{n-1}^j \quad V_n^j)^T, \\ U^j &= (U_0^j \quad U_1^j \quad \ldots \quad U_{n-1}^j \quad U_n^j)^T, \\ B^j &= (B_0^j \quad B_1^j \quad \ldots \quad B_{n-1}^j \quad B_n^j)^T, \\ C^j &= (C_0^j \quad C_1^j \quad \ldots \quad C_{n-1}^j \quad C_n^j)^T. \end{aligned}$$

Let $\mathbf{I}$ to be the identity matrix of size $(n+1)\times(n+1)$. We give the matrix formulation for our discretization of each pricing problem below.

1. European call options:

$$\big(\mathbf{I}-\theta\Delta\tau\mathbf{M}\big)V^{j+1}=\big(\mathbf{I}+(1-\theta)\Delta\tau\mathbf{M}\big)V^{j}, \tag{3.39}$$

where $j=0,\dots,m-1$ and the matrix

$$\mathbf{M}=\begin{pmatrix} -r & 0 & 0 & 0 & \dots & 0 \\ \alpha_1 & -(r+\alpha_1+\beta_1) & \beta_1 & 0 & \dots & 0 \\ 0 & \alpha_2 & -(r+\alpha_2+\beta_2) & \beta_2 & \dots & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & \alpha_{n-1} & -(r+\alpha_{n-1}+\beta_{n-1}) & \beta_{n-1} \\ 0 & 0 & \dots & \gamma_3 & \gamma_2 & \gamma_1 \end{pmatrix}. \tag{3.40}$$

α_i and β_i for $i=1,\dots,n-1$ are defined in (3.11) and (3.12), and γ_1, γ_2 and γ_3 are defined in (3.28).

The initial condition is

$$V_i^0=\max(S_i-E,0), \quad i=0,\dots,n, \tag{3.41}$$

where E is the strike price. There is no free boundary conditions for European call options.

2. American put options:

$$\big(\mathbf{I}-\theta\Delta\tau\mathbf{M}\big)V^{j+1}=\big(\mathbf{I}+(1-\theta)\Delta\tau\mathbf{M}\big)V^{j}, \tag{3.42}$$

where $j=0,\dots,m-1$ and the matrix

$$\mathbf{M}=\begin{pmatrix} -r & 0 & 0 & 0 & \dots & 0 \\ \alpha_1 & -(r+\alpha_1+\beta_1) & \beta_1 & 0 & \dots & 0 \\ 0 & \alpha_2 & -(r+\alpha_2+\beta_2) & \beta_2 & \dots & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & \alpha_{n-1} & -(r+\alpha_{n-1}+\beta_{n-1}) & \beta_{n-1} \\ 0 & 0 & \dots & \gamma_3 & \gamma_2 & \gamma_1 \end{pmatrix}. \tag{3.43}$$

α_i and β_i for $i = 1, \dots, n-1$ are defined in (3.11) and (3.12), and γ_1, γ_2 and γ_3 are defined in (3.28).

Also the following constraints that arise from the early exercise option must be satisfied:

$$V_i^j \geq G_i, \quad i = 1, \dots, n \quad j = 0, \dots, m, \tag{3.44}$$

where $G_i = \max(E - S_i, 0)$, and E is the strike price.

The initial condition is

$$V_i^0 = \max(E - S_i, 0), \quad i = 0, \dots, n. \tag{3.45}$$

3. The TF model for convertible bonds:

U is the total value of the CB and it is computed from

$$\big(\mathbf{I} - \theta\Delta\tau\mathbf{M}_U\big)U^{j+1} = \big(\mathbf{I} + (1-\theta)\Delta\tau\mathbf{M}_U\big)U^j - r_c\Delta\tau\big(\theta B^{j+1} + (1-\theta)B^j\big), \tag{3.46}$$

where $j = 0, \dots, m-1$ and the matrix

$$\mathbf{M}_U = \begin{pmatrix} -r & 0 & 0 & 0 & \dots & 0 \\ \alpha_1 & -(r+\alpha_1+\beta_1) & \beta_1 & 0 & \dots & 0 \\ 0 & \alpha_2 & -(r+\alpha_2+\beta_2) & \beta_2 & \dots & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & \alpha_{n-1} & -(r+\alpha_{n-1}+\beta_{n-1}) & \beta_{n-1} \\ 0 & 0 & \dots & \gamma_3 & \gamma_2 & \gamma_1 \end{pmatrix}. \tag{3.47}$$

α_i and β_i for $i = 1, \dots, n-1$ are defined in (3.11) and (3.12), and γ_1, γ_2 and γ_3 are defined in (3.28).

The initial condition for U is

$$U_i^0 = \max(F + K, \kappa S_i), \quad i = 1, \dots, n, \tag{3.48}$$

where F is the face value of the Bond, K is the coupon payment and κ is the conversion ratio.

Similarly, the COCB component value B of the CB is computed from

$$\big(\mathbf{I} - \theta\Delta\tau\mathbf{M}_B\big)B^{j+1} = \big(\mathbf{I} + (1-\theta)\Delta\tau\mathbf{M}_B\big)B^j, \tag{3.49}$$

where $j = 0, \dots, m-1$ and the matrix

$$\mathbf{M}_B = \begin{pmatrix} -(r+r_c) & 0 & 0 & 0 & \dots & 0 \\ \alpha_1 & -(r+r_c+\alpha_1+\beta_1) & \beta_1 & 0 & \dots & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & \alpha_{n-1} & -(r+r_c+\alpha_{n-1}+\beta_{n-1}) & \beta_{n-1} \\ 0 & 0 & \dots & \gamma_3 & \gamma_2 & \gamma_1 \end{pmatrix}. \tag{3.50}$$

α_i and β_i for $i = 1, \dots, n-1$ are defined in (3.11) and (3.12), and γ_1, γ_2 and γ_3 are defined in (3.28).

The initial condition for B is,

$$B_i^0 = \begin{cases} F+K & \text{if } F+K \geq \kappa S_i, \\ 0 & \text{otherwise,} \end{cases} \tag{3.51}$$

for $i = 0, \dots, n$. The free boundary conditions (2.22), (2.23) and (2.24) should be applied at each time step.

4. The AFV model for convertible bonds:

   U is the total value of CBs and is computed from

$$\begin{aligned} \big(\mathbf{I} - \theta\Delta\tau\mathbf{M}_U\big)U^{j+1} &= \big(\mathbf{I} + (1-\theta)\Delta\tau\mathbf{M}_U\big)U^j \\ &+ p\Delta\tau\Big(\theta \max\big(\kappa S(1-\eta), RB^{j+1}\big) + (1-\theta)\max\big(\kappa S(1-\eta), RB^j\big)\Big), \end{aligned} \tag{3.52}$$

where $j = 0, \dots, m-1$ and the matrix

$$\mathbf{M}_U = \begin{pmatrix} -r_U & 0 & 0 & 0 & \dots & 0 \\ \alpha_1 & -(r_U + \alpha_1 + \beta_1) & \beta_1 & 0 & \dots & 0 \\ 0 & \alpha_2 & -(r_U + \alpha_2 + \beta_2) & \beta_2 & \dots & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & \alpha_{n-1} & -(r_U + \alpha_{n-1} + \beta_{n-1}) & \beta_{n-1} \\ 0 & 0 & \dots & \gamma_3 & \gamma_2 & \gamma_1 \end{pmatrix}. \tag{3.53}$$

Here, $r_U = r + p$, α_i and β_i for $i = 1, \dots, n-1$ are defined in (3.11) and (3.12), and γ_1, γ_2 and γ_3 are defined in (3.28). That initial condition for U is

$$U_i^0 = \max(F + K, \kappa S_i), \quad i = 0, \dots, n. \tag{3.54}$$

For the equity component C of the CB, the parameters a_1, a_2 and a_3 are the same as those of U. Thus C is computed from

$$\begin{aligned} \big(\mathbf{I} - \theta\Delta\tau\mathbf{M}_C\big)C^{j+1} &= \big(\mathbf{I} + (1-\theta)\Delta\tau\mathbf{M}_C\big)C^j \\ &+ p\Delta\tau\Big(\theta \max\big(\kappa S(1-\eta) - RB^{j+1}, 0\big) + (1-\theta)\max\big(\kappa S(1-\eta) - RB^j, 0\big)\Big), \end{aligned} \tag{3.55}$$

where $j = 0, \dots, m-1$ and the matrix $\mathbf{M}_C = \mathbf{M}_U$. The initial condition for C is

$$C_i^0 = \max(\kappa S_i - (F + K), 0), \quad i = 0, \dots, n. \tag{3.56}$$

Similarly, the bond component B is computed from

$$\big(\mathbf{I} - \theta\Delta\tau\mathbf{M}_B\big)B^{j+1} = \big(\mathbf{I} + (1-\theta)\Delta\tau\mathbf{M}_B\big)B^j, \tag{3.57}$$

where $j = 0, \dots, m-1$ and the matrix

$$\mathbf{M}_B = \begin{pmatrix} -r_B & 0 & 0 & 0 & \dots & 0 \\ \alpha_1 & -(r_B + \alpha_1 + \beta_1) & \beta_1 & 0 & \dots & 0 \\ 0 & \alpha_2 & -(r_B + \alpha_2 + \beta_2) & \beta_2 & \dots & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & \alpha_{n-1} & -(r_B + \alpha_{n-1} + \beta_{n-1}) & \beta_{n-1} \\ 0 & 0 & \dots & \gamma_3 & \gamma_2 & \gamma_1 \end{pmatrix}. \tag{3.58}$$

Here, $r_B = r + p(1-R)$, α_i and β_i for $i = 1, \dots, n-1$ are defined in (3.11) and (3.12), and γ_1, γ_2 and γ_3 are defined in (3.28). That initial condition for B is

$$B_i^0 = F + K, \quad i = 0, \dots, n. \tag{3.59}$$

The free boundary conditions (2.39), (2.41), (2.42), (2.45) and (2.46) should be satisfied at each time step.

Chapter 4

Iterative Methods for Solving Free Boundary Problems

In the previous chapter, we review discretizations for the Black-Scholes-like equation (3.2) with appropriate initial and boundary conditions. Options with early-exercise opportunities, such as American puts and convertible bonds, have free boundary conditions. At each step of the resulting numerical method, we must solve a large linear system of equations with nonlinear constraints associated with the free boundary conditions. In this chapter, we consider two iterative methods for solving such systems: the projected successive overrelaxation (PSOR) method and the penalty method. The reason for choosing iterative methods is that the nonlinear constraints associated with the free boundary conditions make the system difficult or impossible to solve with direct methods.

The PSOR method is a well-known scheme for handling free boundary conditions. It is an extension of the well-known successive overrelaxation (SOR) method. The penalty method, suggested by Forsyth and Vetzal [FV02], has been applied more recently to value American options effectively. It has been extended by Nielsen, Skavhaug and Tveito to the continuous penalty method [NST02]. In this chapter, we investigate the application of the PSOR and both the discrete and continuous penalty methods to value American options and convertible bonds. Firstly, we discuss the SOR/PSOR methods, and then the discrete and continuous penalty methods. We also present the algorithms for European call options, American put options and CBs.

4.1 SOR / PSOR Method

4.1.1 Introduction to the SOR/PSOR Method

The successive overrelaxation (SOR) method is an extension of the Gauss-Seidel method. Both can be used to solve the linear system

$$\mathbf{A}\mathbf{x} = \mathbf{b} \tag{4.1}$$

by splitting the matrix $\mathbf{A}$ into three parts

$$\mathbf{A} = \mathbf{L} + \mathbf{D} + \mathbf{U}, \tag{4.2}$$

where $\mathbf{D}$, $\mathbf{L}$ and $\mathbf{U}$ represent the diagonal, strictly lower triangular, and strictly upper triangular parts of $\mathbf{A}$, respectively. Thus, (4.1) can be written as

$$\mathbf{x} = \mathbf{D}^{-1}(-\mathbf{L}\mathbf{x} - \mathbf{U}\mathbf{x} + \mathbf{b}). \tag{4.3}$$

Starting with an initial guess, $\mathbf{x}^0$, to the solution, $\mathbf{x}$, and using previously computed results as soon as they are available, the Gauss-Seidel method calculates the sequence of approximate solutions, x^k, as follows:

$$\mathbf{x}^{k+1} = \mathbf{D}^{-1}(-\mathbf{L}\mathbf{x}^{k+1} - \mathbf{U}\mathbf{x}^k + \mathbf{b}), \quad k = 0, 1, 2, \ldots \tag{4.4}$$

If the sequence converges to a vector $\mathbf{x}$ as k increases, then $\mathbf{x}$ is the solution to (4.1).

In component form, the Gauss-Seidel method can be written as

$$x_i^{k+1} = \frac{1}{a_{ii}}\Big(b_i - \sum_{j=1}^{i-1} a_{ij}x_j^{k+1} - \sum_{j=i+1}^{n} a_{ij}x_j^k\Big). \tag{4.5}$$

The SOR method improves substantially the rate of convergence of the Gauss-Seidel method in many cases by introducing a so-called relaxation parameter ω. In general $0 < \omega < 2$, but most often we have $1 < \omega < 2$. To be more specific, SOR computes the next iterate as a weighted average of the current iterate and the next Gauss-Seidel iterate.

For convenience, we denote the Gauss-Seidel iterate by $\mathbf{y}$ and an element in $\mathbf{y}$ by y_i. The SOR method can be written in component format as

$$\begin{cases} y_i^{k+1} = \dfrac{1}{a_{ii}}\Big(b_i - \sum\limits_{j=1}^{i-1} a_{ij}x_j^{k+1} - \sum\limits_{j=i+1}^{n} a_{ij}x_j^k\Big), \\ x_i^{k+1} = x_i^k + \omega(y_i^{k+1} - x_i^k), \end{cases} \tag{4.6}$$

where $i = 1, \ldots, n$. Notice we use the already available x_j^{k+1}, $j = 1, \ldots, i-1$, to compute y_i^{k+1} to achieve faster convergence.

The Projected SOR (PSOR) method for linear complementarity problems was proposed by Cryer [Cry71]. The method is an extension of the SOR method and widely used for pricing derivatives having an early-exercise feature. The advantage of the method is that it handles the free boundary conditions in a straightforward way and is easily implemented. To illustrate the PSOR method, we consider a problem based on equation (4.1) but subject to an additional constraint

$$\mathbf{x} \geq \mathbf{g}. \tag{4.7}$$

We start with an initial guess $\mathbf{x}^0$. At each iteration of the PSOR method, we check if the new value x_i^{k+1} satisfies constraint (4.7). If not, we adjust x_i^{k+1} to make the constraint satisfied. If $\mathbf{x}^k$ converges to $\mathbf{x}$, then $\mathbf{x}$ is the solution to the problem. More specifically, the PSOR method for solving the above problem is

$$\begin{cases} y_i^{k+1} = \dfrac{1}{a_{ii}}\Big(b_i - \sum\limits_{j=1}^{i-1} a_{ij}x_j^{k+1} - \sum\limits_{j=i+1}^{n} a_{ij}x_j^k\Big) \\ x_i^{k+1} = \max\big(g_i, x_i^k + \omega(y_i^{k+1} - x_i^k)\big), \end{cases} \tag{4.8}$$

where $i = 1, \ldots, n$.

Both the SOR and PSOR methods compute successive approximations to the solution. To be effective, the method must be augmented with a stopping criterion. An ideal stopping criterion would measure the difference between the last iterate and the true solution, but this is not possible in practice. Instead, we can test if the SOR/PSOR iterates are changing significantly. We stop when no apparent progress is being made. To be more specific, the stopping criterion we use is

$$\|\mathbf{x}^{k+1} - \mathbf{x}^{k}\| \leq tol, \tag{4.9}$$

where the norm used is typically either the 2-norm or the ∞-norm, and *tol* is the desired computing tolerance.

Now, consider the relaxation parameter ω, which has a significant impact on the convergence rate of the SOR method. Typically, $1 < \omega < 2$ is used for the SOR method. A suitably chosen ω can reduce the number of iterations dramatically, and therefore choosing an optimal or nearly optimal ω is very important in the SOR method. If $\mathbf{A}$ is a symmetric positive definite (spd) matrix, the unique ω which maximized the rate of convergence of the SOR method may be found [You71]. However, the matrix $\mathbf{A}$ is not spd in many applications and thus it is not easy to calculate the optimal ω in advance, but methods have been developed to approximate it dynamically as the SOR/PSOR iteration progresses. In [WHD95], in the context of using (P)SOR in the numerical solution of time-dependent PDEs, an adaptive approach for choosing an effective value of ω at each time-step is described.

4.1.2 Implementation for European Options

Because European options can be exercised at maturity only, the pricing model has no free boundary condition. Thus, we use the SOR method to solve the linear systems that arise at each step of the numerical solution of the Black-Scholes equation by a θ-method.

It would be more efficient to use a band solver to solve these linear systems, but we chose to use SOR so that we could more easily compare our results for European options to the more complicated problems that use PSOR to solve the associated free-boundary-value problems.

From (3.39), the Crank-Nicolson method (i.e., set $\theta = \frac{1}{2}$) at time τ_{j+1} for a European call option is

$$\big(2\mathbf{I} - \Delta\tau\mathbf{M}\big)V^{j+1} = \big(2\mathbf{I} + \Delta\tau\mathbf{M}\big)V^{j}, \tag{4.10}$$

where $\mathbf{M}$ is defined in (3.40). Let $\mathbf{b} = \big(2\mathbf{I} + \Delta\tau\mathbf{M}\big)V^{j}$. Starting with $V^{j+1,0} = V^{j}$, the $(k+1)^{st}$ SOR iterate for V^{j+1} at time τ_{j+1} is computed by the following procedure.

```
For $i = 0, \ldots, n$
    if $i = 0$, then
        $Y_i^{j+1,k+1} = \frac{1}{2 + r\Delta\tau} b_i$    (4.11)
    if $1 \le i < n$, then
        $Y_i^{j+1,k+1} = \frac{1}{2 + \Delta\tau(r + \alpha_i + \beta_i)}(b_i + \alpha_i\Delta\tau V_{i-1}^{j+1,k+1} + \beta_i\Delta\tau V_{i+1}^{j+1,k})$    (4.12)
    if $i = n$, then
        $Y_i^{j+1,k+1} = \frac{1}{2 - \gamma_1\Delta\tau}(b_i + \gamma_3\Delta\tau V_{i-2}^{j+1,k+1} + \gamma_2\Delta\tau V_{i-1}^{j+1,k+1})$    (4.13)
    $V_i^{j+1,k+1} = V_i^{j+1,k} + \omega(Y_i^{j+1,k+1} - V_i^{j+1,k})$    (4.14)
End
```

Notice that we run the SOR iteration to calculate the two end values, $V_0^{j+1,k+1}$ and $V_n^{j+1,k+1}$, instead of setting the values explicitly as in the case of Dirichlet boundary condition. That is because they are determined by the corresponding PDEs, and interact with their neighbors. For example, $V_n^{j+1,k+1}$ depends on $V_{n-1}^{j+1,k+1}$ which in turn depends on its previous value $V_n^{j+1,k}$.

The pseudocode for the SOR iteration to calculate V^{j+1} at time τ_{j+1} is given in Algorithm 4.1, and the Crank-Nicolson method to calculate the European call options is given in Algorithms 4.2. Notice that Algorithm 4.2 uses a constant time step, but it can be easily generalized to a nonconstant stepsize algorithm.

Algorithm 4.1: SOR method for European call options

```
$V^{j+1,0} = V^j$;
for $k = 0, \ldots, N_{max}$ do
    for $i = 0, \ldots, n$ do
        if $i == 0$ then
            calculate $Y_0^{j+1,k+1}$ using (4.11);
        else if $i == n$ then
            calculate $Y_n^{j+1,k+1}$ using (4.13);
        else
            calculate $Y_i^{j+1,k+1}$ using (4.12);
        end
        calculate $V_i^{j+1,k+1}$ using (4.14);
    end
    $error = \|V^{j+1,k+1} - V^{j+1,k}\|_\infty$;
    if $error \leq tol$ then
        break;
    end
end
$V^{j+1} = V^{j+1,k+1}$;
```

Algorithm 4.2: Crank-Nicolson method for European call options

```
for $i = 0, \dots, n$ do
    $V_i^0 = \max(0, S_i - E)$ ;                  /* $E$ is the strike price.  */
end
$\tau = 0$ ;                                      /* $\tau$ is time to maturity $T$.  */
$\Delta\tau = T/m$;
for $j = 0, \dots, m-1$ do
    $\tau = \tau + \Delta\tau$ ;                  /* $\Delta\tau$ is the timestep size.  */
    use Algorithm (4.1) to calculate $V^{j+1}$ from $V^j$;
end
```

4.1.3 Implementation for American Options

The price of an American put option is given by the solution of a free-boundary-value problem. We impose the free boundary condition (2.12), $V(S,t) \geq G(S)$, at each iteration by comparing the value of the option $\widetilde{V}^{j+1,k}$ that we could obtain if we didn't exercise the option to the payoff value G that we could obtain if we did exercise the option, and taking the bigger value as the option price $V^{j+1,k}$ at this point.

From (3.42) with $\theta = \frac{1}{2}$, the Crank-Nicolson method for an American put option is

$$\big(2\mathbf{I} - \Delta\tau\mathbf{M}\big)V^{j+1} = \big(2\mathbf{I} + \Delta\tau\mathbf{M}\big)V^j, \tag{4.15}$$

where $\mathbf{M}$ is defined in (3.43).

Analogous to the SOR method for a European call option discussed above, we start by setting $V^{j+1,0} = V^j$ and $\mathbf{b} = \big(2\mathbf{I} + \Delta\tau\mathbf{M}\big)V^j$. The PSOR iteration for calculating the $(k+1)^{st}$ iterate of V^{j+1} at time τ_{j+1} is given by

For $i = 1, \ldots, n-1$

if $i = 0$, then

$$Y_i^{j+1,k+1} = \frac{1}{2 + r\Delta\tau} b_i \tag{4.16}$$

if $1 \le i < n$, then

$$Y_i^{j+1,k+1} = \frac{1}{2 + \Delta\tau(r + \alpha_i + \beta_i)} (b_i + \alpha_i \Delta\tau V_{i-1}^{j+1,k+1} + \beta_i \Delta\tau V_{i+1}^{j+1,k}) \tag{4.17}$$

if $i = n$, then

$$Y_i^{j+1,k+1} = \frac{1}{2 - \gamma_1 \Delta\tau} (b_i + \gamma_3 \Delta\tau V_{i-2}^{j+1,k+1} + \gamma_2 \Delta\tau V_{i-1}^{j+1,k+1}) \tag{4.18}$$

$$\widetilde{V}_i^{j+1,k+1} = V_i^{j+1,k} + \omega(Y_i^{j+1,k+1} - V_i^{j+1,k}) \tag{4.19}$$

$$V_i^{j+1,k+1} = \max(G_i, \widetilde{V}_i^{j+1,k+1}) \tag{4.20}$$

End

We give the pseudocode for the PSOR iteration in Algorithm 4.3. The pseudocode for the Crank-Nicolson method is similar to Algorithm 4.2, except that the initial vector is $V^0 = \max(0, E - S)$ and we use the PSOR method to compute V^{j+1}.

4.1.4 Implementation for Convertible Bonds

Under the TF Model

In Chapter 3, we use the θ-method to derive a pair of coupled equations (3.46) and (3.49) with free boundary conditions that are solved for U^j and B^j at time τ_j. If the Crank-Nicolson method is used to discretize in time (i.e., setting $\theta = \frac{1}{2}$), equations (3.46) and (3.49) become

$$\big(2\mathbf{I} - \Delta\tau \mathbf{M}_U\big) U^{j+1} = \big(2\mathbf{I} + \Delta\tau \mathbf{M}_U\big) U^j - r_c \Delta\tau \big(B^{j+1} + B^j\big), \tag{4.21}$$

$$\big(2\mathbf{I} - \Delta\tau \mathbf{M}_B\big) B^{j+1} = \big(2\mathbf{I} + \Delta\tau \mathbf{M}_B\big) B^j, \tag{4.22}$$

Algorithm 4.3: PSOR method for American put options

```
$V^{j+1,0} = V^j$;
for $k = 0, \dots, N_{max}$ do
    for $i = 0, \dots, n$ do
        if $i == 0$ then
            calculate $Y_0^{j+1,k+1}$ using (4.16);
        else if $i == n$ then
            calculate $Y_n^{j+1,k+1}$ using (4.18);
        else
            calculate $Y_i^{j+1,k+1}$ using (4.17);
        end
        calculate $\widetilde{V}_i^{j+1,k+1}$ using (4.19);
        calculate $V_i^{j+1,k+1}$ using (4.20);
    end
    $error = \|V^{j+1,k+1} - V^{j+1,k}\|_\infty$;
    if $error \leq tol$ then
        break;
    end
end
$V^{j+1} = V^{j+1,k+1}$;
```

where $\mathbf{M}_U$ and $\mathbf{M}_B$ are defined in (3.47) and (3.50), respectively.

Ignoring any constraints initially, given U^j and B^j, we can solve (4.22) for B^{j+1} first and then use B^{j+1} in (4.21) to solve for U^{j+1}. After we get the preliminary values of U^{j+1} and B^{j+1} by ignoring the constraints, denoted by $\widetilde{U}^{j+1}$ and $\widetilde{B}^{j+1}$, respectively, we impose the constraints and obtain the values of U^{j+1} and B^{j+1}.

Let the initial values $U^{j+1,0} = U^j$ and $B^{j+1,0} = B^j$. We outline the procedure to solve the $(k+1)^{st}$ PSOR iterates of U^{j+1} and B^{j+1} at time τ_{j+1}. For each $i = 0, \ldots, n$, U_i^{j+1} and B_i^{j+1} are calculated from the following 5 steps:

1. Let $\mathbf{b}_B = \big(2\mathbf{I} + \Delta\tau\mathbf{M}_B\big)B^j$. Ignoring the constraints, we compute the $(k+1)^{st}$ Gauss-Seidel estimate $\bar{B}_i^{j+1,k+1}$ for $B_i^{j+1,k+1}$, and take the weighted average of $\bar{B}_i^{j+1,k+1}$ and $B_i^{j+1,k}$ to compute the SOR value $\widetilde{B}_i^{j+1,k+1}$ for $B_i^{j+1,k+1}$ as follows:

   if $i = 0$, then
   $$\bar{B}_i^{j+1,k+1} = \frac{1}{2 + \Delta\tau(r + r_c)} b_{B,i}, \tag{4.23}$$
   if $1 \leq i < n$, then
   $$\bar{B}_i^{j+1,k+1} = \frac{1}{2 + \Delta\tau(r + r_c + \alpha_i + \beta_i)}(b_{B,i} + \alpha_i \Delta\tau B_{i-1}^{j+1,k+1} + \beta_i \Delta\tau B_{i+1}^{j+1,k}) \tag{4.24}$$
   if $i = n$, then
   $$\bar{B}_i^{j+1,k+1} = \frac{1}{2 - \gamma_1\Delta\tau}(b_{B,i} + \gamma_3\Delta\tau B_{i-2}^{j+1,k+1} + \gamma_2\Delta\tau B_{i-1}^{j+1,k+1}), \tag{4.25}$$
   $$\widetilde{B}_i^{j+1,k+1} = B_i^{j+1,k} + \omega(\bar{B}_i^{j+1,k+1} - B_i^{j+1,k}). \tag{4.26}$$

2. Set $B_i^{j+1,k+1} = \widetilde{B}_i^{j+1,k+1}$ as the preliminary value for $B_i^{j+1,k+1}$, which possibly changes later in step 5 when we apply the constraints.

3. Let $\mathbf{b}_U^{k+1} = \big(2\mathbf{I} + \Delta\tau\mathbf{M}_U\big)U^j - r_c\Delta\tau(B^{j+1,k+1} + B^j)$. Ignoring the constraints, we then compute the $(k+1)^{st}$ Gauss-Seidel estimate $\bar{U}_i^{j+1,k+1}$ for $U_i^{j+1,k+1}$, and compute the SOR value of $\widetilde{U}_i^{j+1,k+1}$ for $U_i^{j+1,k+1}$ by taking the weighted average of

$\bar{U}_i^{j+1,k+1}$ and $U_i^{j+1,k}$ as follows:

if $i = 0$, then

$$\bar{U}_i^{j+1,k+1} = \frac{1}{2 + r\Delta\tau} b_{U,i}^{k+1}, \tag{4.27}$$

if $1 \le i < n$, then

$$\bar{U}_i^{j+1,k+1} = \frac{1}{2 + \Delta\tau(r + \alpha_i + \beta_i)} \left(b_{U,i}^{k+1} + \alpha_i \Delta\tau U_{i-1}^{j+1,k+1} + \beta_i \Delta\tau U_{i+1}^{j+1,k} \right) \tag{4.28}$$

if $i = n$, then

$$\bar{U}_i^{j+1,k+1} = \frac{1}{2 - \gamma_1 \Delta\tau} (b_{U,i}^{k+1} + \gamma_3 \Delta\tau U_{i-2}^{j+1,k+1} + \gamma_2 \Delta\tau U_{i-1}^{j+1,k+1}), \tag{4.29}$$

$$\widetilde{U}_i^{j+1,k+1} = U_i^{j+1,k} + \omega(\bar{U}_i^{j+1,k+1} - U_i^{j+1,k}). \tag{4.30}$$

4. Set $U_i^{j+1,k+1} = \widetilde{U}_i^{j+1,k+1}$ as the pre-value of $U_i^{j+1,k+1}$.

5. Explicitly applying the free boundary conditions (2.22), (2.23) and (2.24) simultaneously to $U_i^{j+1,k+1}$ and $B_i^{j+1,k+1}$ (see Algorithm 4.4), we obtain the post-PSOR values of $U_i^{j+1,k+1}$ and $B_i^{j+1,k+1}$.

After we compute the vectors V^{j+1} and B^{j+1} from the above procedure, we check the following stopping criteria:

$$\begin{cases} \|B^{j+1,k+1} - B^{j+1,k}\|_\infty \le tol, \\ \|U^{j+1,k+1} - U^{j+1,k}\|_\infty \le tol. \end{cases}$$

The pseudocode for the Crank-Nicolson method and PSOR iteration are given in Algorithms 4.5 and 4.6.

Under the AFV Model

The idea behind using the PSOR method to price CBs with the AFV model is similar to that for the TF model, but, in the AFV model, we use the Crank-Nicolson method to

Algorithm 4.4: Explicit application of the free boundary conditions to U_i^{j+1} and B_i^{j+1} in the TF model.

```
Input: $U_i^{j+1}$ and $B_i^{j+1}$
/* Check the minimum-value constraints.                                  */
if $B_p > \kappa S_i$ then
    if $U_i^{j+1} < B_p$ then                 /* the puttability constraint.  */
        $B_i^{j+1} = B_p$;    $U_i^{j+1} = B_p$;
    end
else
    if $U_i^{j+1} < \kappa S_i$ then              /* the conversion constraint.  */
        $B_i^{j+1} = 0$;    $U_i^{j+1} = \kappa S_i$;
    end
end
/* Check the maximum-value constraints.                                  */
if $U_i^{j+1} > \max(B_c, \kappa S_i)$ then         /* the callability constraint.  */
    $B_i^{j+1} = 0$;    $U_i^{j+1} = \max(B_c, \kappa S_i)$;
end
```

Algorithm 4.5: Crank-Nicolson method for convertible bonds using the TF model

```
/* $F$ is the face value of the CB; $K$ is the coupon payment.        */
$\tau = 0$; $\Delta\tau = T/m$;
$U^0 = F + K$; $B^0 = F + K$;
for $i = 0, \ldots, n$ do
    if $U_i^0 < \kappa S_i$ then
        $U_i^0 = \kappa S_i$; $B_i^0 = 0$;
    end
end
for $j = 0, \ldots, m-1$ do
    /* Adjust $\Delta\tau$ if necessary to make $\tau_i$ match the coupon dates
       exactly.                                                         */
    $\tau = \tau + \Delta\tau$;
    calculate $AccI(\tau)$ using (2.60);
    if $\tau \in \{call\ period\}$ then      /* call period is in time backwards.  */
        $B_c = B_{cc} + AccI$;
    else
        $B_c = \infty$;
    end
    if $\tau \in \{put\ period\}$ then       /* put period is in time backwards.  */
        $B_p = B_{pc} + AccI$;
    else
        $B_p = -\infty$;
    end
    compute $B^{j+1}$ and $U^{j+1}$ by calling the PSOR Algorithm (4.6);
    /* coupon payment dates are in time backwards.                       */
    if $\tau \in \{coupon\ payment\ dates\}$ then
        $U^{j+1} = U^{j+1} + K$;
        $B^{j+1} = B^{j+1} + K$;
    end
end
```

Algorithm 4.6: PSOR method for convertible bonds using the TF model

```
$U^{j+1,0} = U^j$ ;
$B^{j+1,0} = B^j$;
for $k = 0, \dots, N_{max}$ do
    for $i = 0, \dots, n$ do
        calculate $\bar{B}_i^{j+1,k+1}$ using (4.23),(4.24) and (4.25);
        calculate $\widetilde{B}_i^{j+1,k+1}$ using (4.26);
        $B_i^{j+1,k+1} = \widetilde{B}_i^{j+1,k+1}$;
        calculate $\bar{U}_i^{j+1,k+1}$ using (4.27),(4.28) and (4.29);
        calculate $\widetilde{U}_i^{j+1,k+1}$ using (4.30);
        $U_i^{j+1,k+1} = \widetilde{U}_i^{j+1,k+1}$;
        Apply the free boundary conditions explicitly to updated $U_i^{j+1,k+1}$ and
        $B_i^{j+1,k+1}$ using Algorithm 4.4;
    end
    $error_U = \|U^{j+1,k+1} - U^{j+1,k}\|_\infty$;
    $error_B = \|B^{j+1,k+1} - B^{j+1,k}\|_\infty$;
    if $error_U \leq tol$ and $error_B \leq tol$ then
        break;
    end
end
$B^{j+1} = B^{j+1,k+1}$;
$U^{j+1} = U^{j+1,k+1}$;
```

solve the PDEs, (3.52), (3.55) and (3.57), associated with the AFV model, resulting in the formulas

$$\begin{aligned}
\big(2\mathbf{I} - \Delta\tau\mathbf{M}_U\big)U^{j+1} &= \big(2\mathbf{I} + \Delta\tau\mathbf{M}_U\big)U^j \\
&\quad + p\Delta\tau\Big(\max\big(\kappa S(1-\eta), RB^{j+1}\big) + \max\big(\kappa S(1-\eta), RB^j\big)\Big), \quad (4.31)\\
\big(2\mathbf{I} - \Delta\tau\mathbf{M}_B\big)B^{j+1} &= \big(2\mathbf{I} + \Delta\tau\mathbf{M}_B\big)B^j, \quad (4.32)\\
\big(2\mathbf{I} - \Delta\tau\mathbf{M}_C\big)C^{j+1} &= \big(2\mathbf{I} + \Delta\tau\mathbf{M}_C\big)C^j \\
&\quad + p\Delta\tau\Big(\max\big(\kappa S(1-\eta) - RB^{j+1}, 0\big) + \max\big(\kappa S(1-\eta) - RB^j, 0\big)\Big). \quad (4.33)
\end{aligned}$$

$\mathbf{M}_U$, $\mathbf{M}_B$ and $\mathbf{M}_C$ are defined in (3.53), (3.58) and (3.53), respectively.

Since $U = B + C$, we compute B and C at each time step and then sum them to obtain the value U. Let $r_U = r_C = r + p$ and $r_B = r + p(1 - R)$. The PSOR iteration to compute U^{j+1}, B^{j+1} and C^{j+1} at time τ_{j+1} as described below. For each $i = 0, \ldots, n$, U_i^{j+1}, B_i^{j+1} and C_i^{j+1} are calculated from the following 6 steps:

1. Let $\mathbf{b}_B = \big(2\mathbf{I} + \Delta\tau\mathbf{M}_B\big)B^j$. Ignoring any constraints, we compute the $(k+1)^{st}$ Gauss-Seidel estimate $\bar{B}_i^{j+1,k+1}$ for $B_i^{j+1,k+1}$, and take the weighted average of $\bar{B}_i^{j+1,k+1}$ and $B_i^{j+1,k}$ to compute the SOR value $\widetilde{B}_i^{j+1,k+1}$ for $B_i^{j+1,k+1}$ as follows:

   if $i = 0$, then
$$\bar{B}_i^{j+1,k+1} = \frac{1}{2 + r_B\Delta\tau} b_{B,i}, \quad (4.34)$$
   if $1 \le i < n$, then
$$\bar{B}_i^{j+1,k+1} = \frac{1}{2 + \Delta\tau(r_B + \alpha_i + \beta_i)}(b_{B,i} + \alpha_i\Delta\tau B_{i-1}^{j+1,k+1} + \beta_i\Delta\tau B_{i+1}^{j+1,k}) \quad (4.35)$$
   if $i = n$, then
$$\bar{B}_i^{j+1,k+1} = \frac{1}{2 - \gamma_1\Delta\tau}(b_{B,i} + \gamma_3\Delta\tau B_{i-2}^{j+1,k+1} + \gamma_2\Delta\tau B_{i-1}^{j+1,k+1}), \quad (4.36)$$
$$\widetilde{B}_i^{j+1,k+1} = B_i^{j+1,k} + \omega(\bar{B}_i^{j+1,k+1} - B_i^{j+1,k}). \quad (4.37)$$

2. Set $B_i^{j+1,k+1} = \widetilde{B}_i^{j+1,k+1}$ as the preliminary value of $B_i^{j+1,k+1}$.

3. Let

$$\mathbf{b}_C^{k+1} = \big(2\mathbf{I}+\Delta\tau\mathbf{M}_C\big)C^j\big)+p\Delta\tau\Big(\max\big(\kappa S(1-\eta)-RB^{j+1,k+1},0\big)+\max\big(\kappa S(1-\eta)-RB^j,0\big)\Big).$$

Ignoring any constraints, we then compute the $(k+1)^{st}$ Gauss-Seidel estimate $\bar{C}_i^{j+1,k+1}$ for $C_i^{j+1,k+1}$, and compute the SOR value $\widetilde{C}_i^{j+1,k+1}$ for $C_i^{j+1,k+1}$ by taking the weighted average of $\bar{C}_i^{j+1,k+1}$ and $C_i^{j+1,k}$:

if $i=0$, then

$$\bar{C}_i^{j+1,k+1} = \frac{1}{2+r_C\Delta\tau} b_{C,i}^{k+1}, \tag{4.38}$$

if $1 \le i < n$, then

$$\bar{C}_i^{j+1,k+1} = \frac{1}{2+\Delta\tau(r_C+\alpha_i+\beta_i)}\big(b_{C,i}^{k+1} + \alpha_i\Delta\tau C_{i-1}^{j+1,k+1} + \beta_i\Delta\tau C_{i+1}^{j+1,k}\big) \tag{4.39}$$

if $i=n$, then

$$\bar{C}_i^{j+1,k+1} = \frac{1}{2-\gamma_1\Delta\tau}(b_{C,i}^{k+1} + \gamma_3\Delta\tau C_{i-2}^{j+1,k+1} + \gamma_2\Delta\tau C_{i-1}^{j+1,k+1}), \tag{4.40}$$

$$\widetilde{C}_i^{j+1,k+1} = C_i^{j+1,k} + \omega(\bar{C}_i^{j+1,k+1} - C_i^{j+1,k}). \tag{4.41}$$

4. Set $C_i^{j+1,k+1} = \widetilde{C}_i^{j+1,k+1}$ as the pre-value of $C_i^{j+1,k+1}$.

5. Explicitly applying the free boundary conditions (2.41), (2.42), (2.45) and (2.46) simultaneously to $B_i^{j+1,k+1}$ and $C_i^{j+1,k+1}$ (see Algorithm 4.7), we obtain the post-PSOR values of $B_i^{j+1,k+1}$ and $C_i^{j+1,k+1}$.

6. Compute $U_i^{j+1,k+1} = B_i^{j+1,k+1} + C_i^{j+1,k+1}$.

As for the TF model, we check the stopping criteria after obtaining the vectors U^{j+1}, B^{j+1} and C^{j+1}:

$$\begin{cases} \|B^{j+1,k+1} - B^{j+1,k}\|_\infty \le tol, \\ \|C^{j+1,k+1} - C^{j+1,k}\|_\infty \le tol. \end{cases}$$

Algorithm 4.7: Explicit application of the free boundary conditions to C_i^{j+1} and B_i^{j+1} in the AFV model.

```
Input: $B_i^{j+1}$ and $C_i^{j+1}$
/* Check the minimum-value constraints.                                   */
$B_i^{j+1} = \min(B_c, B_i^{j+1})$;
if $B_p > \kappa S_i$ then
    $B_i^{j+1} = \max(B_i^{j+1}, B_p - C_i^{j+1})$ ;      /* the puttability constraint.  */
else
    $C_i^{j+1} = \max(\kappa S_i - B_i^{j+1}, C_i^{j+1})$ ;      /* the conversion constraint.  */
end
/* Check the maximum-value constraints.                                   */
$C_i^{j+1} = \min(C_i^{j+1}, \max(\kappa S_i, B_c) - B_i^{j+1})$ ; /* the callability constraint.  */
```

The pseudocode for the Crank-Nicolson method and PSOR iteration for the AFV model is similar to that for the TF model, but, for comparison, we outline it for the AFV model in Algorithms 4.8 and 4.9.

4.2 Penalty Method

As we have seen, the free boundary conditions for American options and convertible bonds result in a nonlinear problem that can be solved as a LCP. At each time τ_j, we need to determine the optimal exercise point to compute V^j. The PSOR method solves this LCP by imposing the free boundary conditions explicitly at each iteration. Now we introduce another iterative method to solve the LCP: the penalty method. In fact, we consider two forms of the penalty method: the discrete and the continuous versions. Both versions of the penalty method impose the free boundary condition implicitly by adding a nonlinear penalty term to the original linear PDE. The nonlinear algebraic equations

Algorithm 4.8: Crank-Nicolson timestepping for convertible bonds using the AFV model

```
/* $F$ is the face value of the CB; $K$ is the coupon payment            */
$\tau = 0$; $\Delta\tau = T/m$;
$B^0 = (F + C)$ ;
$C^0 = \max(\kappa S - (F + C), 0)$ ;
$U^0 = B^0 + C^0$ ;
for $j = 0, \ldots, m - 1$ do
    /* Adjust $\Delta\tau$ if necessary to make $\tau_i$ match the coupon dates
       exactly.                                                            */
    $\tau = \tau + \Delta\tau$;
    calculate $AccI(\tau)$ using (2.60);
    if $\tau \in \{call\ period\}$ then
        $B_c = B_{cc} + AccI(\tau)$;
    else
        $B_c = \infty$;
    end
    if $\tau \in \{put\ period\}$ then
        $B_p = B_{pc} + AccI(\tau)$;
    else
        $B_p = -\infty$;
    end
    calculate $C^{k+1}, B^{k+1}$, and $U^{k+1}$ by calling Algorithm (4.9);
    if $\tau \in \{coupon\ payment\ dates\}$ then
        $U^{j+1} = U^{j+1} + K$;
        $B^{j+1} = B^{j+1} + K$;
    end
end
```

Algorithm 4.9: PSOR Iteration for convertible bonds using the AFV model

```
$B^{j+1,0} = B^j$ ;
$C^{j+1,0} = C^j$;
for $k = 0, \dots, N_{max}$ do
    for $i = 0, \dots, n$ do
        calculate $\bar{B}_i^{j+1,k+1}$ using (4.34),(4.35) and (4.36);
        calculate $\widetilde{B}_i^{j+1,k+1}$ using (4.37);
        $B_i^{j+1,k+1} = \widetilde{B}_i^{j+1,k+1}$;
        calculate $\bar{C}_i^{j+1,k+1}$ using (4.38),(4.39) and (4.40);
        calculate $\widetilde{C}_i^{j+1,k+1}$ using (4.41);
        $C_i^{j+1,k+1} = \widetilde{C}_i^{j+1,k+1}$;
        apply explicitly the free boundary condition to $B_i^{j+1,k+1}$ and $C_i^{j+1,k+1}$ using
        Algorithm 4.7;
    end
    $error_B = \|B^{j+1,k+1} - B^{j+1,k}\|_\infty$;
    $error_C = \|C^{j+1,k+1} - C^{j+1,k}\|_\infty$;
    if $error_B \leq tol$ and $error_C \leq tol$ then
        break;
    end
end
$B^{j+1} = B^{j+1,k+1}$;
$C^{j+1} = C^{j+1,k+1}$;
$U^{j+1} = B^{j+1,k+1} + C^{j+1,k+1}$;
```

that result from discretizing the PDE are solved with Newton's iteration or some other iterative schemes. We discuss the penalty method for an American option first, and then extend our approach to a convertible bond.

4.2.1 Discrete Penalty Method for American Options

Consider the PDE associated with an American option problem

$$\frac{\partial V}{\partial \tau} = \frac{1}{2}\sigma^2 S^2 \frac{\partial^2 V}{\partial S^2} + rS\frac{\partial V}{\partial S} - rV, \tag{4.42}$$

which is subject to the free boundary condition

$$V \geq G, \tag{4.43}$$

where G is the payoff condition. We can rewrite this problem as a LCP

$$\begin{pmatrix} \mathcal{L}V = 0 \\ V \geq G \end{pmatrix} \vee \begin{pmatrix} \mathcal{L}V \leq 0 \\ V = G \end{pmatrix}, \tag{4.44}$$

where

$$\mathcal{L}V = -\frac{\partial V}{\partial \tau} + \frac{1}{2}\sigma^2 S^2 \frac{\partial^2 V}{\partial S^2} + rS\frac{\partial V}{\partial S} - rV. \tag{4.45}$$

For an American put option, the payoff function is

$$G(S) = \max(E - S, 0). \tag{4.46}$$

The discrete penalty method proposed by Forsyth and Vetzal [FV02] adds a positive penalty term $\xi \max(G - V, 0)$ to the original Black-Scholes PDE (4.42) to enforce the free boundary condition $V \geq G$. For $\xi \to \infty$ and $0 < \epsilon \ll 1$, the penalty term effectively ensures that the solution satisfies $V \geq G - \epsilon$. Therefore, we solve the following nonlinear PDE for approximating the solution to the LCP (4.44):

$$\frac{\partial V}{\partial \tau} = \frac{1}{2}\sigma^2 S^2 \frac{\partial^2 V}{\partial S^2} + rS\frac{\partial V}{\partial S} - rV + \xi \max(G - V, 0). \tag{4.47}$$

Except for the additional penalty term, the discretization of (4.47) is the same as that for the original PDE described in Chapter 3. When calculating V^{j+1} at time τ_{j+1}, the penalty term is replaced with a discrete penalty term $P(V_i^{j+1})(G_i - V_i^{j+1})$, where $P(V_i^{j+1})$ is defined as

$$P(V_i^{j+1}) = \begin{cases} Large & \text{if } V_i^{j+1} < G_i, \\ 0 & \text{otherwise.} \end{cases}$$

Thus, following the discretization procedure discussed in Chapter 3, we have

$$\begin{aligned} & V_i^{j+1} - V_i^j = \theta\Delta\tau\bar{\mathcal{L}}V_i^{j+1} + (1-\theta)\Delta\tau\bar{\mathcal{L}}V_i^j + P(V_i^{j+1})(G_i - V_i^{j+1}) \\ \Longrightarrow\quad & V_i^{j+1} - \theta\Delta\tau\big(\alpha_i V_{i-1}^{j+1} - (r+\alpha_i+\beta_i)V_i^{j+1} + \beta_i V_{i+1}^{j+1}\big) + P(V_i^{j+1})V_i^{j+1} \\ =\quad & V_i^j + (1-\theta)\Delta\tau\big(\alpha_i V_{i-1}^j - (r+\alpha_i+\beta_i)V_i^j + \beta_i V_{i+1}^j\big) + P(V_i^{j+1})G_i. \end{aligned}$$

We can write this in matrix form as

$$[\mathbf{I} - \theta\Delta\tau\mathbf{M} + \mathbf{P}(V^{j+1})]V^{j+1} = [\mathbf{I} + (1-\theta)\Delta\tau\mathbf{M}]V^j + \mathbf{P}(V^{j+1})G, \tag{4.48}$$

where $\mathbf{M}$ is given in (3.43) and $\mathbf{P}(V^{j+1})$ is a diagonal matrix of size $(n+1)\times(n+1)$ with each diagonal element defined by $P(V_i^{j+1})$,

$$\mathbf{P}(V^{j+1})_{ik} = \begin{cases} Large & \text{if } V_i^{j+1} < G_i \text{ and } i = k, \\ 0 & \text{otherwise.} \end{cases} \tag{4.49}$$

The parameter *Large* is closely related to the maximum relative error in enforcing the free boundary condition using the penalty method. This maximum relative error is measured by the quantity

$$\mathbf{Err} = \max_{i,j} \frac{\max[0, (G_i - V_i^j)]}{\max(1, G_i)}.$$

To ensure **Err** is small enough, the parameter *Large* must be chosen large enough. In [FV02], a practical criterion is suggested in term of the required accuracy:

$$Large \approx \frac{1}{tol}, \tag{4.50}$$

where *tol* is the tolerance associated with the stopping criterion for Newton's iteration used to solve (4.48).

Now we introduce Newton's method to solve the discrete nonlinear equation (4.48). Define

$$\mathcal{F}(V^{j+1}) = [\mathbf{I} - \theta\Delta\tau\mathbf{M}]V^{j+1} - [\mathbf{I} + (1-\theta)\Delta\tau\mathbf{M}]V^{j} - \mathbf{P}(V^{j+1})(G - V^{j+1}), \qquad (4.51)$$

Solving (4.48) is equivalent to solving $\mathcal{F}(V^{j+1}) = 0$ for V^{j+1}. To apply Newton's method to solve (4.51), we need to compute the Jacobian of $\mathcal{F}(V^{j+1})$. However, $P(V^{j+1})$ is a discontinuous function. So we need to define its "derivative" carefully. Forsyth and Vetzal [FV02] suggest using

$$\frac{\partial P_{i,k}^{j+1}(G_i - V_i^{j+1})}{\partial V_i^{j+1}} = \begin{cases} -Large & \text{if } V_i^{j+1} < G_i, \quad i = k, \\ 0 & \text{otherwise,} \end{cases} \qquad (4.52)$$

whence the Jacobian of $\mathcal{F}(V^{j+1})$ is

$$\mathbf{J}(V^{j+1}) = \mathbf{I} - \theta\Delta\tau\mathbf{M} + \mathbf{P}(V^{j+1}).$$

Therefore, Newton's iteration for solving $\mathcal{F}(V^{j+1}) = 0$ is

$$V^{j+1,k+1} = V^{j+1,k} - \mathbf{J}^{-1}(V^{j+1,k})\mathcal{F}(V^{j+1,k}), \qquad (4.53)$$

where $V^{j+1,k}$ is the k^{th} iterate for V^{j+1}. Combining (4.51)-(4.53), we obtain the iteration formula

$$[\mathbf{I} - \theta\Delta\tau\mathbf{M} + \mathbf{P}(V^{j+1,k})]V^{j+1,k+1} = [\mathbf{I} + (1-\theta)\Delta\tau\mathbf{M}]V^{j} + \mathbf{P}(V^{j+1,k})G. \qquad (4.54)$$

Setting $\theta = \frac{1}{2}$, we obtain the Crank-Nicolson scheme for American put options:

$$[2\mathbf{I} - \Delta\tau\mathbf{M} + 2\mathbf{P}(V^{j+1,k})]V^{j+1,k+1} = [2\mathbf{I} + \Delta\tau\mathbf{M}]V^{j} + 2\mathbf{P}(V^{j+1,k})G. \qquad (4.55)$$

Let $V^{j+1,k}$ be the k^{th} iterate for V^{j+1}, $\mathbf{P}^k = \mathbf{P}(V^{j+1,k})$ and the initial iteration vector $V^{j+1,0} = V^j$. The psuedocode for Newton's iteration to solve V^{j+1} at time τ_{j+1} is given in Algorithm 4.10. The stopping criterion to exit Newton's iteration is

$$\max_i \frac{|V_i^{j+1,k+1} - V_i^{j+1,k}|}{\max(1, |V_i^{j+1,k+1}|)} < tol, \tag{4.56}$$

or

$$\mathbf{P}^{k+1} = \mathbf{P}^k. \tag{4.57}$$

The pseudocode for the Crank-Nicolson method using the discrete penalty method for an American option shown in Algorithm 4.10 is similar to that shown in Algorithm 4.2, except that the SOR method is replaced by the penalty method to solve for V^{j+1}.

Algorithm 4.10: Discrete penalty iteration for American put options

```
/* $tol$ is the required relative error.                                   */
$G = \max(E - S, 0)$;
$V^{j+1,0} = V^j$;
calculate $\mathbf{P}^0$ using (4.49);
for $k = 0, \ldots, N_{max}$ do
    calculate $V^{j+1,k+1}$ by solving (4.55);
    calculate $\mathbf{P}^{k+1}$ using (4.49) ;
    $error = \max_i \frac{|V_i^{j+1,k+1} - V_i^{j+1,k}|}{\max(1,|V_i^{j+1,k+1}|)}$;
    if $[error \leq tol]$ or $[\mathbf{P}^{k+1} == \mathbf{P}^k]$ then
        break;
    end
end
$V^{j+1} = V^{j+1,k+1}$
```

4.2.2 Continuous Penalty Method for American Options

The continuous penalty method proposed in [NST02] adds a continuous penalty term to the original Black-Scholes PDE (4.42), giving rise to the following nonlinear PDE posed on a fixed domain:

$$\frac{\partial V}{\partial \tau} = \frac{1}{2}\sigma^2 S^2 \frac{\partial^2 V}{\partial S^2} + rS\frac{\partial V}{\partial S} - rV + \frac{\epsilon\chi}{V + \epsilon - Q}, \tag{4.58}$$

where $0 < \epsilon \ll 1$, χ is a positive constant, and $Q = (E - S)$ for American put options. Note that we use $Q = (E - S)$ instead of the payoff function $Q = \max(E - S, 0)$ since the Black-Scholes equation ensures that $V \geq 0$. The penalty term $\frac{\epsilon\chi}{V+\epsilon-Q}$ is of order ϵ if $V = V(S,t) \gg Q$, and increases towards χ as $V \to Q$.

In [NST02], it is recommended that we treat the penalty term explicitly by replacing V with V_i^j when we discretize the penalty term, but the conditions $\chi \geq rE$ and $\Delta\tau \leq \frac{\epsilon}{rE}$ must be satisfied to ensure that the method is stable. Discretizing the penalty term as suggested above and the other terms in (4.58) as usual, we have

$$\begin{aligned}
& V_i^{j+1} - V_i^j = \theta\Delta\tau\bar{\mathcal{L}}V_i^{j+1} + (1-\theta)\Delta\tau\bar{\mathcal{L}}V_i^j + \frac{\epsilon\chi}{V_i^j + \epsilon - Q_i} \\
\Longrightarrow \quad & V_i^{j+1} - \theta\Delta\tau\big(\alpha_i V_{i-1}^{j+1} - (r + \alpha_i + \beta_i)V_i^{j+1} + \beta_i V_{i+1}^{j+1}\big) \\
= \quad & V_i^j + (1-\theta)\Delta\tau\big(\alpha_i V_{i-1}^j - (r + \alpha_i + \beta_i)V_i^j + \beta_i V_{i+1}^j\big) + \frac{\epsilon\chi}{V_i^j + \epsilon - Q_i}.
\end{aligned}$$

We can write this in matrix form as

$$[\mathbf{I} - \theta\Delta\tau\mathbf{M}]V^{j+1} = [\mathbf{I} + (1-\theta)\Delta\tau\mathbf{M}]V^j + R(V^j), \tag{4.59}$$

where $\mathbf{M}$ is given in (3.43). $R(V^j)$ is the function associated with the penalty term:

$$R(V^j)_i = \frac{\epsilon\chi}{V_i^j + \epsilon - Q_i}. \tag{4.60}$$

Solving (4.59) is straightforward. We use the direct method to compute V^{j+1}.

4.2.3 Implementation of the Discrete Penalty Method for the TF Model

In this subsection, we discuss a procedure to price a convertible bond using the discrete penalty method in the TF model. To begin, we ignore any constraints and solve B^{j+1} from

(4.22) and then solve U^{j+1} from (4.21) using the direct method. We then explicitly impose the free boundary conditions on U^{j+1} and B^{j+1} to obtain approximations to U and B at time τ_{j+1}. We could now proceed to the next time step. However, even though we use the Crank-Nicolson method, we would not achieve a second-order rate of convergence with this scheme since we enforce the free boundary conditions explicitly. Thus, an additional step is suggested in [AFV03] to enhance the convergence. By computing U^{j+1} with the penalty method, we enforce the free boundary conditions in an implicit way and then adjust B^{j+1} according to the current U^{j+1}. We expect the convergence of this numerical method incorporating the penalty method to be better than the explicit method outlined above, but quadratic convergence still is not attained since we apply the penalty method to U only, and B is still computed explicitly. The convergence of B directly affects the convergence of U.

In the following, we describe the method to compute U using the discrete penalty method, and then present the procedure to solve the whole problem.

The Penalty Method for Computing U

Under the TF model, the PDE governing the convertible bond value U is

$$\frac{\partial U}{\partial \tau} = \frac{1}{2}\sigma^2 S^2 \frac{\partial^2 U}{\partial S^2} + rS\frac{\partial U}{\partial S} - rU - r_c B \tag{4.61}$$

subject to the free boundary conditions

$$\begin{aligned} U &\geq \max(B_p, \kappa S), \\ U &\leq \max(B_c, \kappa S). \end{aligned} \tag{4.62}$$

The problem (4.61)-(4.62) can be written as the LCP

$$\begin{pmatrix} \mathcal{L}U - r_c B = 0 \\ U \geq \max(B_p, \kappa S) \\ U \leq \max(B_c, \kappa S) \end{pmatrix} \vee \begin{pmatrix} \mathcal{L}U - r_c B \leq 0 \\ U = \max(B_p, \kappa S) \\ U \leq \max(B_c, \kappa S) \end{pmatrix} \vee \begin{pmatrix} \mathcal{L}U - r_c B \geq 0 \\ U \geq \max(B_p, \kappa S) \\ U = \max(B_c, \kappa S) \end{pmatrix}. \tag{4.63}$$

It can be solved with the discrete penalty method. The description of the algorithm and experiments are presented in [Mo06]. Here, we introduce a slightly different way to solve (4.61)-(4.62), and describe the algorithm in the next subsection. The experiments and comparison with the results in [Mo06] are presented in Chapter 5. Instead of applying the penalty method directly to (4.63), we separate the solution domain into two regions: $B_c > \kappa S$ and $B_c \leq \kappa S$, and solve the LCP only in domain of $B_c > \kappa S$, using penalty method. More explicitly, we reformulate the problem as follows:

- $B_c > \kappa S$

$$\begin{pmatrix} \mathcal{L}U - r_c B = 0 \\ U \geq \max(B_p, \kappa S) \\ U \leq B_c \end{pmatrix} \vee \begin{pmatrix} \mathcal{L}U - r_c B \leq 0 \\ U = \max(B_p, \kappa S) \\ U \leq B_c \end{pmatrix} \vee \begin{pmatrix} \mathcal{L}U - r_c B \geq 0 \\ U \geq \max(B_p, \kappa S) \\ U = B_c \end{pmatrix}, \tag{4.64}$$

- $B_c \leq \kappa S$

$$U = \kappa S. \tag{4.65}$$

Clearly, (4.64) is a LCP and thus we can solve it with the penalty method. We can enforce (4.65) directly.

Now, we describe the discrete penalty method for solving the LCP (4.64). Define the lower boundary and the upper boundary as follows:

$$\begin{aligned} G_l(S) &= \max(B_p, \kappa S), \\ G_u(S) &= B_c. \end{aligned} \tag{4.66}$$

Since (4.64) is restricted by two boundary conditions, we need to add two penalty terms to the original Black-Scholes-like PDE (4.61) to enforce these two conditions. Thus, we propose the following nonlinear PDE to approximate the solution to the LCP (4.64):

$$\frac{\partial U}{\partial \tau} = \frac{1}{2}\sigma^2 S^2 \frac{\partial^2 U}{\partial S^2} + rS\frac{\partial U}{\partial S} - rU - r_c B + \xi_1 \max(G_l - U, 0) - \xi_2 \max(U - G_u, 0). \tag{4.67}$$

From the analysis in [FV02], as $\xi_1 \to \infty$ and $\xi_2 \to \infty$, we expect the solution to (4.67) to satisfy $G_l - \epsilon_1 \leq U \leq G_u + \epsilon_2$ for $0 < \epsilon_1 \ll 1$ and $0 < \epsilon_2 \ll 1$. In other worlds, the solution U approaches the solution to the LCP (4.61) when $\xi_1 \to \infty$ and $\xi_2 \to \infty$.

Analogously to the discretization of (4.47) in the previous section, we introduce the discrete penalty terms $P_l(U^{j+1})(G_l - U^{j+1})$ and $P_u(U^{j+1})(U^{j+1} - G_u)$ to replace $\xi_1 \max(G_l - U, 0)$ and $\xi_2 \max(U - G_u, 0)$ in (4.67). They are given by

$$P_l(U_i^{j+1}) = \begin{cases} Large_1 & \text{if } U_i^{j+1} < G_{l,i}, \\ 0 & \text{otherwise,} \end{cases}$$

and

$$P_u(U_i^{j+1}) = \begin{cases} Large_2 & \text{if } U_i^{j+1} > G_{u,i}, \\ 0 & \text{otherwise,} \end{cases}$$

respectively. Thus, we have

$$\begin{aligned} U_i^{j+1} - U_i^j &= \theta\Delta\tau\bar{\mathcal{L}}U_i^{j+1} + (1-\theta)\Delta\tau\bar{\mathcal{L}}U_i^j - r_c\Delta\tau\big(\theta B_i^{j+1} + (1-\theta)B_i^j\big) \\ &\quad + P_l(U_i^{j+1})(G_{l,i} - U_i^{j+1}) - P_u(U_i^{j+1})(U_i^{j+1} - G_{u,i}), \end{aligned}$$

which can be written in matrix form as

$$\begin{aligned} & [\mathbf{I} - \theta\Delta\tau\mathbf{M}_U + \mathbf{P}_l(U^{j+1}) + \mathbf{P}_u(U^{j+1})]U^{j+1} \\ = \; & [\mathbf{I} + (1-\theta)\Delta\tau\mathbf{M}_U]U^j - r_c\Delta\tau[\theta B^{j+1} + (1-\theta)B^j] \\ & + \mathbf{P}_l(U^{j+1})G_l + \mathbf{P}_u(U^{j+1})G_u, \end{aligned} \tag{4.68}$$

where $\mathbf{M}_U$ is defined in (3.47), and $\mathbf{P}_l(U^{j+1})$ and $\mathbf{P}_u(U^{j+1})$ are diagonal matrices of size $(n+1) \times (n+1)$ with diagonal elements defined by $P_l(U_i^{j+1})$ and $P_u(U_i^{j+1})$, respectively. That is,

$$\mathbf{P}_l(U^{j+1})_{ik} = \begin{cases} Large_1 & \text{if } U_i^{j+1} < G_{l,i} \text{ and } i = k, \\ 0 & \text{otherwise,} \end{cases} \tag{4.69}$$

and

$$\mathbf{P}_u(U^{j+1})_{ik} = \begin{cases} Large_2 & \text{if } U_i^{j+1} > G_{u,i} \text{ and } i = k, \\ 0 & \text{otherwise,} \end{cases} \tag{4.70}$$

respectively. Similar to the case of American options, we set $Large_1 = Large_2 \approx \frac{1}{tol}$, where tol is the tolerance associated with Newton's iteration used to solve (4.68).

We use Newton's method to solve (4.68). To this end, define

$$\begin{aligned}\mathcal{F}(U^{j+1}) &= [\mathbf{I} - \theta\Delta\tau\mathbf{M}_U]U^{j+1} - [\mathbf{I} + (1-\theta)\Delta\tau\mathbf{M}_U]U^j + r_c\Delta\tau\big(\theta B^{j+1} + (1-\theta)B^j\big) \\ &\quad + \mathbf{P}_u(U^{j+1})(U^{j+1} - G_u) - \mathbf{P}_l(U^{j+1})(G_l - U^{j+1}).\end{aligned} \tag{4.71}$$

As in subsection 4.2.1, we define the "derivatives" of the penalty terms to be

$$\frac{\partial P^{j+1}_{l,ik}(G_{l,i} - U^{j+1}_i)}{\partial U^{j+1}_i} = \begin{cases} -Large_1 & \text{if } U^{j+1}_i < G_{l,i}, \quad i = k, \\ 0 & \text{otherwise,} \end{cases}$$

and

$$\frac{\partial P^{j+1}_{u,ik}(U^{j+1}_i - G_{u,i})}{\partial U^{j+1}_i} = \begin{cases} Large_2 & \text{if } U^{j+1}_i > G_{u,i}, \quad i = k, \\ 0 & \text{otherwise.} \end{cases}$$

Thus, the Jacobian of $\mathcal{F}(U^{j+1})$ is

$$\mathbf{J}(U^{j+1}) = \mathbf{I} - \theta\Delta\tau\mathbf{M}_U + \mathbf{P}_u(U^{j+1}) + \mathbf{P}_l(U^{j+1}), \tag{4.72}$$

and Newton's iteration is

$$U^{j+1,k+1} = U^{j+1,k} - \mathbf{J}^{-1}(U^{j+1,k})\mathcal{F}(U^{j+1,k}). \tag{4.73}$$

We can rewrite (4.71)-(4.73) in the simpler form

$$\begin{aligned} & [\mathbf{I} - \theta\Delta\tau\mathbf{M}_U + \mathbf{P}_l(U^{j+1,k}) + \mathbf{P}_u(U^{j+1,k})]U^{j+1,k+1} \\ = \; & [\mathbf{I} + (1-\theta)\Delta\tau\mathbf{M}_U]U^j - r_c\Delta\tau[\theta B^{j+1} + (1-\theta)B^j] \\ & + \mathbf{P}_l(U^{j+1,k})G_l + \mathbf{P}_u(U^{j+1,k})G_u. \end{aligned} \tag{4.74}$$

The Algorithm for Pricing CBs Under the TF Model

Based on the Crank-Nicolson method, the coupled equations (4.21) and (4.22) can be used to approximate the price of a convertible bond. In the following we present an algorithm to decouple (4.21) and (4.22) and solve for U^{j+1} at time τ_{j+1}. The psuedocode is given in Algorithm 4.12.

Given the initial values for U^j and B^j, the method proceeds as follows:

1. Ignoring the constraints, we estimate the value of B^{j+1} by solving (4.22) using a direct method. Denote this estimate by $\bar{B}^{j+1}$:
$$\big(2\mathbf{I} - \Delta\tau\mathbf{M}_B\big)\bar{B}^{j+1} = \big(2\mathbf{I} + \Delta\tau\mathbf{M}_B\big)B^j. \tag{4.75}$$

2. Ignoring the constraints, we estimate the value of U^{j+1} by solving (4.21) using a direct method. Denote this estimate by $\bar{U}^{j+1}$:
$$\big(2\mathbf{I} - \Delta\tau\mathbf{M}_U\big)\bar{U}^{j+1} = \big(2\mathbf{I} + \Delta\tau\mathbf{M}_U\big)U^j - r_c\Delta\tau\big(\bar{B}^{j+1} + B^j\big). \tag{4.76}$$

3. Apply the free boundary conditions explicitly to the estimates $\bar{U}^{j+1}$ and $\bar{B}^{j+1}$ using Algorithm 4.4 to obtain the estimates $\widetilde{U}^{j+1}$ and $\widetilde{B}^{j+1}$ for U^{j+1} and B^{j+1}, respectively.

4. Given $\widetilde{B}^{j+1}$, compute U^{j+1} using the penalty method described in the previous section. Setting $\theta = \frac{1}{2}$, we have the Crank-Nicolson formula for the region $B_c > \kappa S$:
$$\begin{aligned} &[2\mathbf{I} - \Delta\tau\mathbf{M}_U + 2\mathbf{P}_l(U^{j+1,k}) + 2\mathbf{P}_u(U^{j+1,k})]U^{j+1,k+1} \\ = \quad &[2\mathbf{I} + \Delta\tau\mathbf{M}_U]U^j - r_c\Delta\tau[B^{j+1} + B^j] + 2\mathbf{P}_l(U^{j+1,k})G_l + 2\mathbf{P}_u(U^{j+1,k})G_u. \end{aligned} \tag{4.77}$$
Let $U^{j+1,k}$ be the k^{th} estimate for U^{j+1}, $\mathbf{P}_l^k = \mathbf{P}_l(U^{j+1,k})$ and $\mathbf{P}_u^k = \mathbf{P}_u(U^{j+1,k})$. Starting with the initial value $U^{j+1,0} = U^j$, the psuedocode for Newton's iteration to solve U^{j+1} at time τ_{j+1} is given in Algorithm 4.11. The stopping criterion is
$$\max_i \frac{|U_i^{j+1,k+1} - U_i^{j+1,k}|}{\max(1, |U_i^{j+1,k+1}|)} < tol, \tag{4.78}$$
or
$$\mathbf{P}_l^{k+1} = \mathbf{P}_l^k \quad \text{and} \quad \mathbf{P}_u^{k+1} = \mathbf{P}_u^k. \tag{4.79}$$
If $B_c \leq \kappa S$, set $U_i^{j+1} = \kappa S_i$.

5. Adjust B^{j+1} with U^{j+1},

$$B^{j+1} = \min(\widetilde{B}^{j+1}, U^{j+1}). \tag{4.80}$$

Pseudocode for the Crank-Nicolson method is similar to Algorithm 4.5.

Algorithm 4.11: Discrete penalty iteration to solve for U^{j+1} in the TF model

```
/* Apply the penalty method only to $U^{j+1}$.                                 */
/* Apply the free boundary constraints to $U^{j+1}$ implicitly.                */
$U^{j+1,0} = U^j$;
compute the penalty matrix $\mathbf{P}_l(U^{j+1,0})$ using (4.69);
compute the penalty matrix $\mathbf{P}_u(U^{j+1,0})$ using (4.70);
for $k = 0, \ldots, N_{max}$ do
    calculate $U^{j+1,k+1}$ by solving (4.77) ;
    calculate $\mathbf{P}_l(U^{j+1,k+1})$ using (4.69) ;
    calculate $\mathbf{P}_u(U^{j+1,k+1})$ using (4.70) ;
    $error = \max_i \frac{|U_i^{k+1,j+1} - U_i^{k+1,j}|}{\max(1, |U_i^{k+1,j+1}|)}$;
    if $[error \leq tol]$ or $[\mathbf{P}_u^{k+1} == \mathbf{P}_u^k$ and $\mathbf{P}_l^{k+1} == \mathbf{P}_l^k]$ then
        break;
    end
end
$U^{j+1} = U^{j+1,k+1}$;
```

4.2.4 Implementation of the Discrete Penalty Method for the AFV Model

Our approach to computing the price U of a CB using the AFV model is similar to that described above for the TF model, but we solve for the two components B and C in the AFV model instead of solving for U and B in the TF model. To begin, we compute B^{j+1} and C^{j+1} from (4.32) and (4.33) using the direct method, and then set

Algorithm 4.12: Discrete penalty iteration for the TF model

```
$G_l = \max(B_p, \kappa S)$; $G_u = B_c$;
calculate $\bar{B}^{j+1}$ using (4.75) ;
calculate $\bar{U}^{j+1}$ using (4.76);
for $i = 0, \ldots, n$ do
    calculate $\widetilde{U}_i^{j+1}$ and $\widetilde{B}_i^{j+1}$ by applying constraints explicitly to $\bar{U}_i^{j+1}$ and $\bar{B}_i^{j+1}$
    using Algorithm 4.4;
end
/* The additional step to calculate $U^{j+1}$ using the penalty method */
compute $U^{j+1}$ by calling Algorithm 4.11 given $\widetilde{B}^{j+1}$;
for $i = 0, \ldots, n$ do
    if $B_c \leq \kappa S_i$ then
        $U_i^{j+1} = \kappa S_i$;
    end
end
$B^{j+1} = \min(\widetilde{B}^{j+1}, U^{j+1})$;
```

$U^{j+1} = B^{j+1} + C^{j+1}$. Next, we impose the free boundary conditions explicitly on U^{j+1}, B^{j+1} and C^{j+1}. We could now proceed to the next step, but instead we compute U^{j+1} by the penalty method to enhance the convergence. After computing U^{j+1} with the penalty method, we adjust B^{j+1} and C^{j+1}, then proceed the next time step. Clearly, this procedure to compute U has similar problems as those described above for the TF model. As a result, quadratic convergence may not be attained.

In the following, we describe how to compute U using the penalty method, and then describe how to solve the whole problem.

The Penalty Method for Computing U

Under the AFV model, the PDE governing the convertible bond price U is

$$\frac{\partial U}{\partial \tau} = \frac{1}{2}\sigma^2 S^2 \frac{\partial^2 U}{\partial S^2} + (r + p\eta)S\frac{\partial U}{\partial S} - (r+p)U + p\max(\kappa S(1-\eta), RB) \tag{4.81}$$

with the free boundary conditions

$$\begin{aligned} U &\geq \max(B_p, \kappa S), \\ U &\leq \max(B_c, \kappa S). \end{aligned} \tag{4.82}$$

As we did for the TF model, we separate the solution domain into two regions: $B_c > \kappa S$ and $B_c \leq \kappa S$. Then we solve the LCP (2.35) for U using the penalty method if $B_c > \kappa S$ and enforce the Dirichlet boundary condition (2.36) on U directly otherwise.

Following the same approach as discussed for the TF model, we apply the penalty method to the LCP (2.35) only, giving rise to the nonlinear PDE

$$\begin{aligned} \frac{\partial U}{\partial \tau} = \frac{1}{2}\sigma^2 S^2 \frac{\partial^2 U}{\partial S^2} + (r + p\eta)S\frac{\partial U}{\partial S} - (r+p)U + p\max(\kappa S(1-\eta), RB) \\ + \xi_1 \max(G_l - U, 0) - \xi_2 \max(U - G_u, 0), \end{aligned} \tag{4.83}$$

where ξ_1 and ξ_2 are positive penalty parameters, and G_l and G_u are defined in (4.66). We discretize equation (4.83) in our usual manner and write the resulting equation in matrix form as

$$\begin{aligned}
&[\mathbf{I} - \theta\Delta\tau\mathbf{M}_U + \mathbf{P}_l(U^{j+1}) + \mathbf{P}_u(U^{j+1})]U^{j+1} \\
= \;& [\mathbf{I} + (1-\theta)\Delta\tau\mathbf{M}_U]U^j + \mathbf{P}_l(U^{j+1})G_l + \mathbf{P}_u(U^{j+1})G_u \\
&+ p\Delta\tau[\theta\max\left(\kappa S(1-\eta), RB^{j+1}\right) + (1-\theta)\max\left(\kappa S(1-\eta), RB^j\right)],
\end{aligned} \tag{4.84}$$

where $\mathbf{M}_U$ is defined in (3.53), and $\mathbf{P}_l$ and $\mathbf{P}_u$ are defined in (4.69) and (4.70), respectively.

We solve (4.84) using Newton's method. Newton's iteration at each time step is

$$\begin{aligned}
&[\mathbf{I} - \theta\Delta\tau\mathbf{M}_U + \mathbf{P}_l(U^{j+1,k}) + \mathbf{P}_u(U^{j+1,k})]U^{j+1,k+1} \\
= \;& [\mathbf{I} + (1-\theta)\Delta\tau\mathbf{M}_U]U^j + \mathbf{P}_l(U^{j+1,k})G_l + \mathbf{P}_u(U^{j+1,k})G_u \\
&+ p\Delta\tau[\theta\max\left(\kappa S(1-\eta), RB^{j+1}\right) + (1-\theta)\max\left(\kappa S(1-\eta), RB^j\right)].
\end{aligned} \tag{4.85}$$

The pseudocode for using Newton's method to solve (4.85) for U^{j+1} is the similar to Algorithm 4.11. The only difference is that we compute $U^{j+1,k+1}$ using (4.85) instead of (4.77).

The Algorithm for Pricing CBs Under the AFV Model

Unlike the TF model, in the AFV model, we solve for the two components B and C instead of U, and then compute U using (4.84) based on the estimated value of B. Given the initial values of B^j, C^j and U^j, the descriptions of the algorithm for solving for U^{j+1} at time τ_{j+1} under the AFV model is as follows.

1. Ignoring the constraints, we compute $\bar{B}^{j+1}$, an estimate for B^{j+1}:

$$\left(2\mathbf{I} - \Delta\tau\mathbf{M}_B\right)\bar{B}^{j+1} = \left(2\mathbf{I} + \Delta\tau\mathbf{M}_B\right)B^j. \tag{4.86}$$

2. Ignoring the constraints, we compute $\bar{C}^{j+1}$, an estimate for C^{j+1}:

$$\begin{aligned}
&\left(2\mathbf{I} - \Delta\tau\mathbf{M}_C\right)\bar{C}^{j+1} = \left(2\mathbf{I} + \Delta\tau\mathbf{M}_C\right)C^j \\
&\quad + p\Delta\tau\Big(\max\left(\kappa S(1-\eta) - R\bar{B}^{j+1}, 0\right) + \max\left(\kappa S(1-\eta) - RB^j, 0\right)\Big).
\end{aligned} \tag{4.87}$$

3. Apply the free boundary conditions explicitly to the estimates $\bar{B}^{j+1}$ and $\bar{C}^{j+1}$ using Algorithm 4.7 to obtain the estimates $\widetilde{B}^{j+1}$ and $\widetilde{C}^{j+1}$ for B^{j+1} and C^{j+1}, respectively.

4. Compute U^{j+1} using the penalty method (4.84), given $\widetilde{B}^{j+1}$. The psuedocode for Newton's iteration to solve (4.84) is given in Algorithm 4.11, and the psuedocode for computing U^{j+1} at time τ_{j+1} is given in Algorithm 4.13.

5. Adjust B^{j+1} and C^{j+1} based on U^{j+1},

$$B^{j+1} = \min(\widetilde{B}^{j+1}, U^{j+1}), \tag{4.88}$$

$$C^{j+1} = U^{j+1} - B^{j+1}. \tag{4.89}$$

The pseudocode for the Crank-Nicolson method is similar to Algorithm 4.8.

Algorithm 4.13: Discrete penalty iteration for the AFV model

```
$G_l = \max(B_p, \kappa S)$; $G_u = B_c$;
calculate $\bar{B}^{j+1}$ using (4.86);
calculate $\bar{C}^{j+1}$ using (4.87);
for $i = 0, \ldots, n$ do
    apply the constraints to $\bar{B}_i^{j+1}$ and $\bar{C}_i^{j+1}$ to obtain $\widetilde{B}_i^{j+1}$ and $\widetilde{C}_i^{j+1}$ using
    Algorithm 4.7;
end
/* The additional step to calculate $U^{j+1}$ using the penalty method */
compute $U^{j+1}$ by calling Algorithm 4.11 given $\widetilde{B}^{j+1}$;
for $i = 0, \ldots, n$ do
    if $B_c \le \kappa S_i$ then
        $U_i^{j+1} = \kappa S_i$;
    end
end
$U^{j+1} = U^{j+1,k+1}$;
$B^{j+1} = \min(\widetilde{B}^{j+1}, U^{j+1})$;
$C^{j+1} = U^{j+1} - B^{j+1}$;
```

Chapter 5

Observations and Improvements

The numerical experiments performed in this chapter explore the effectiveness of the Crank-Nicolson method used to solve the parabolic PDEs and associated boundary conditions that arise in the TF and AFV models. Because the PDEs in both models are Black-Scholes-like equations, the analysis of the difficulties associated with pricing convertible bonds is similar to that for pricing European and American options, such as the far-field linear boundary conditions, discontinuities in the payoff functions, and free boundary conditions. Thus, even though we are primarily interested in pricing convertible bonds, we start by testing our methods on much simpler European and American call or put options.

After studying the effectiveness of our numerical methods for the European and American options, we recommend some techniques to enhance the stability and convergence of the Crank-Nicolson method, and then apply these techniques to convertible bonds to achieve more stable and accurate solutions.

In the following sections, unless we explicitly specify otherwise, we use the test parameter settings shown in Tables 5.1, 5.2 and 5.3. Those parameter settings are chosen to allow us to conveniently compare our experimental results with those that appear in other papers, such as in [TF98], [FV02], [AFV03], [Li05] and [Mo06]. Table 5.1 lists the parameters we used for pricing a European call option and an American put option. Table 5.2 lists the parameters we used for pricing a convertible bond using the TF model, and Table 5.3 for pricing a convertible bond using the AFV model. In Chapter 2, we discussed the relationship between the TF and AFV models. We noted that these two models are similar if $\eta = 0$ and $r_c = p(1 - R)$. We use this choice of parameters in our experiments to exploit this relationship, thereby making it easier to compare the numerical results for the two models.

In addition to testing our numerical methods for pricing derivatives, we also need to test the effectiveness of our schemes for computing the Delta and Gamma associated with the derivatives because an accurate Delta and Gamma are needed in practice for important applications such as hedging. Delta and Gamma are defined by

$$\text{Delta} = \frac{\partial V}{\partial S}, \tag{5.1}$$

$$\text{Gamma} = \frac{\partial^2 V}{\partial S^2}, \tag{5.2}$$

and approximated using the formulas (3.6) and (3.5), respectively. However, in some situations, even though the value of derivatives appears smooth, oscillations appear in Delta (particularly near the strike price) and are magnified in Gamma when we use simple numerical methods to solve pricing problems. Thus, testing Delta and Gamma is necessary.

In the following, we define some conventions used in connection with our experimental results. “Value” is the price of a derivative (a European option, an American option, or a CB) at $t = 0$ and $S = 100$; “Grid Size” is the number of gridpoints in the spatial dimension; “Time-Steps” is the number of steps in the time dimension; “No. of Iterations” is the number of (P)SOR or Newton’s iterations for each time-step; “Max” is the maximum number of (P)SOR or Newton’s iterations used over all time-steps; “Min” is the minimum number of (P)SOR or Newton’s iterations; and “Avg.” is the average number of (P)SOR or Newton’s iterations over all time-steps. Moreover, the average value of ω over all time-steps for a certain "Grid Size" , as computed by the adaptive technique described in [WHD95] is denoted by "Avg ω " . We double the "Grid Size" and "Time-Steps" at each refinement.

For European options, we can use the analytical solution to calculate errors and convergence rates for the numerical methods, but, for American options and convertible bonds, we do not have such an analytical solution. Thus, we use the difference between the numerical solutions computed with different grid sizes to approximate the convergence rate of the numerical solution. However, since we are more interested in calculating the price when at-the-money, we study the properties of the solutions for that single point. Thus, we use the difference between the numerical solutions for the exercise price computed with different grid sizes to approximate its convergence rate. More specifically, let $V_{(k)}$ denote the numerical solution for the exercise price S obtained with grid size $2^k n$, $k = 0, \ldots, l-1$. For example, $V_{(3)}$ denote the numerical solution for an American put option at $S = 100$ computed with $k = 3$ and $n = 8$ (i.e., grid size is 64). We calculate

$$(\text{Difference})_k = V_{(k+1)} - V_{(k)} \quad k = 0, \ldots, l-1, \tag{5.3}$$

$$(\text{Ratio})_k = \frac{(\text{Difference})_{k+1}}{(\text{Difference})_k}, \quad k = 0, \ldots, l-2, \tag{5.4}$$

and denote these quantities by "Diff." and "Ratio", respectively.

We also calculate the matrix condition number associated with the pricing problems. The condition number of a matrix is defined by

$$\text{Condition number} = \|\mathbf{A}\| \cdot \|\mathbf{A}^{-1}\|. \tag{5.5}$$

It measures the sensitivity of the solution of the linear system $\mathbf{Ax} = \mathbf{b}$ with respect to changes in the vector $\mathbf{b}$ and in the matrix $\mathbf{A}$. A problem with a small condition number (close to 1) is said to be well-conditioned. That is, small changes in $\mathbf{b}$ or $\mathbf{A}$ produce small changes in $\mathbf{x}$. On the other hand, a problem with a large condition number is said to be ill-conditioned. That is, small changes in $\mathbf{b}$ or $\mathbf{A}$ may produce large changes in $\mathbf{x}$. Thus, a numerical solution to such a system cannot be trusted. In the extreme situation, when the condition number approaches infinity, the matrix is almost singular and the solution of the linear system is prone to large numerical errors. We denote the condition number by “Cond. No.”.

Table 5.1: Model parameters for pricing the European/American options

Parameter	Value
Time to maturity	0.25 years
Interest rate r	10% or 0.10
Strike price E	100
Volatility	80% or 0.8

Table 5.2: TF model parameters for pricing the convertible bond

Parameter	Value
Time to maturity	5 years
Conversion	0 to 5 years into 1 share
Conversion ratio κ	1.0
Bond face value F	100
Clean call price B_{cc}	110 from year 3 to year 5
Clean put price B_{pc}	105 at year 3 (during the third year)
Coupon payments K	4.0
Coupon dates	$0.5, 1.0, 1.5, \ldots, 5.0$
Risk-free interest rate r	5% or 0.05
Credit risk r_c	2% or 0.02
Volatility σ	20% or 0.20

Table 5.3: AFV model parameters for pricing the convertible bond

Parameter	Value
Time to maturity	5 years
Conversion	0 to 5 years into 1 share
Conversion ratio κ	1.0
Bond face value F	100
Clean call price B_{cc}	110 from year 3 to year 5
Clean put price B_{pc}	105 at year 3 (during the third year)
Coupon payments K	4.0
Coupon dates	$0.5, 1.0, 1.5, \ldots, 5.0$
Risk-free interest rate r	5% or 0.05
Hazard rate p	2% or 0.02
Recovery factor R	0.0
Stock price jump factor η	0.0
Volatility σ	20% or 0.20

5.1 Observations

5.1.1 M-matrix

Consider a European call option or an American put option. The PDE (2.2) governing the option price is a convection-diffusion equation. The Crank-Nicolson method for PDE (2.2) written in matrix formula is given by

$$\big(2\mathbf{I}-\Delta\tau\mathbf{M}\big)V^{j+1}=\big(2\mathbf{I}+\Delta\tau\mathbf{M}\big)V^{j}, \tag{5.6}$$

where

$$\mathbf{M}=\begin{pmatrix} -r & 0 & 0 & 0 & \dots & 0\\ \alpha_1 & -(r+\alpha_1+\beta_1) & \beta_1 & 0 & \dots & 0\\ 0 & \alpha_2 & -(r+\alpha_2+\beta_2) & \beta_2 & \dots & 0\\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots\\ 0 & 0 & \dots & \alpha_{n-1} & -(r+\alpha_{n-1}+\beta_{n-1}) & \beta_{n-1}\\ 0 & 0 & \dots & \gamma_3 & \gamma_2 & \gamma_1 \end{pmatrix}. \tag{5.7}$$

Here,

$$\alpha_i=\frac{\sigma^2S_i^2}{h_{-1}D_i}-\frac{rh_1S_i}{h_{-1}D_i}, \tag{5.8}$$

$$\beta_i=\frac{\sigma^2S_i^2}{h_1D_i}+\frac{rh_{-1}S_i}{h_1D_i}. \tag{5.9}$$

where $D_i=(h_{i,-1}+h_{i,1})$. As usual, $h_{i,-1}=S_i-S_{i-1}$ and $h_{i,1}=S_{i+1}-S_i$, and we denote them as h_1 and h_{-1}, respectively, for the node i for convenience. At the rightmost node n, we assume that $h_{-1}=h_{-2}=h$. If we use the second-order backward difference scheme for node n, then

$$\gamma_1=\frac{3rS_n}{2h}-r,\quad \gamma_2=-\frac{2rS_n}{h},\quad \gamma_3=\frac{rS_n}{2h}. \tag{5.10}$$

Now, we discuss some properties of the matrix $\mathbf{M}$. If the convection term $\frac{\partial V}{\partial S}$ dominates the diffusion term $\frac{\partial^2 V}{\partial S^2}$ in (2.2), the use of the central difference approximation for the convection term $\frac{\partial V}{\partial S}$ may lead to negative off-diagonal elements α_i in $\mathbf{M}$, thus resulting in positive off-diagonal elements in $\mathbf{A} = 2\mathbf{I} - \Delta\tau\mathbf{M}$. As discussed in section 3.4.3, if $\mathbf{A}$ has positive off-diagonal elements, we cannot guarantee that it is an M-matrix. Therefore, we replace the central difference approximation (3.6) for $\frac{\partial V}{\partial S}$ by the forward difference approximation (3.7). This ensures the off-diagonal elements in $\mathbf{A}$ are non-positive , whence rows 1 through $n-1$ in A are strictly diagonally dominant (Row 0 is strictly diagonally dominated). Thus, we have

$$\alpha_i = \frac{\sigma^2 S_i^2}{h_{-1} D_i}, \tag{5.11}$$

$$\beta_i = \frac{\sigma^2 S_i^2}{h_1 D_i} + \frac{r S_i}{h_1}. \tag{5.12}$$

This modification may result in first-order convergence overall, but, if a few nodes only are modified and they are far away from the region of interest, the rate of convergence may still be approximately second-order. Alternatively, we may require that the spatial stepsize satisfies

$$h_{i,1} \le \frac{\sigma^2 S_i}{r} \quad i = 1, 2, \ldots, \tag{5.13}$$

which ensures that $\alpha_i \ge 0$. For the constant spatial step sizes, if node 1 satisfies (5.13), then nodes $2, 3, \ldots$ satisfy it too. Therefore, we need only check the inequality $\frac{\sigma^2}{r} \ge 1$, since $S_0 = 0$ and $S_1 = h_{1,1}$. For the problems associated with Table 5.1 for both the European and American options, $r = 0.1$ and $\sigma = 0.8$, whence $\frac{\sigma^2}{r} \ge 1$ is ensured. Thus, rows 1 through $n-1$ of the matrix $\mathbf{A}$ are diagonally dominant. If $r = 0.05$ and $\sigma = 0.2$, as in Tables 5.2 and 5.3, $\frac{\sigma^2}{r} \ge 1$ is not satisfied at node 1, but (5.13) is satisfied for nodes $n = 2, 3, \ldots$. In this situation, we use the forward difference scheme to replace the central difference scheme for the convection term at the node 1, as we discussed above.

So far we have discussed how to ensure rows 1 through $n-1$ are diagonally dominant. Now, we consider the nth row. Notice that in (5.10), $\gamma_2 < 0$, $\gamma_3 > 0$, and $S_n \gg 0$ and thus we normally have that $|\gamma_1| = |-r-(\gamma_2+\gamma_3)| < |\gamma_2|+|\gamma_3|$. Hence, we cannot guarantee the last row in $\mathbf{A}$ is diagonally dominant. Therefore, $\mathbf{A}$ may not be an M-matrix when we use the linear boundary condition for $S = S_{max}$. This may cause convergence and stability problems, but our numerical experiments show that this is not the case in practice. It can be seen from Table 5.4 for the European call option and Table 5.5 for the American put option that our numerical method for the price of the option at $S = E = 100$ is stable and approximately second-order. Moreover, Figure 5.1 shows that the computed value, Delta, and Gamma are well behaved for the range of S values of interest. This excellent performance is obtained despite the nth row of $\mathbf{A}$ not being diagonally dominant. We believe this is because the right end is far away from the region of interest ($S_{max} = 400$ vs. $S_{interst} = 100$) and thus the potential loss of accuracy at the right end does not have a significant effect on the region of interest. Therefore, we believe we can safely use this linear boundary approximation to solve Black-Scholes-like problems.

Table 5.4: Numerical results for a European call option at $S = 100$ using the SOR method ($1 < \omega < 2$) to solve the associated linear equations. The numerical method uses the linear boundary condition (5.10), $tol = 1.0\text{e}^{-7}$, and $S_{max} = 400$. The analytical solution is 16.92091465.

Grid	Time-	Value at	Error	Error	Avg.	No. of Iterations		
Size	Steps	$S = 100$		Ratio	ω	Min.	Max.	Avg.
40	40	16.80412510	1.173e-001		1.13	16	20	16.425
80	80	16.89195291	2.942e-002	3.988	1.23	22	34	23.038
160	160	16.91368848	7.341e-003	4.008	1.36	29	59	30.975
320	320	16.91910899	1.834e-003	4.002	1.49	40	102	42.525
640	640	16.92046328	4.586e-004	4.000	1.60	52	174	55.080
1280	1280	16.92080178	1.147e-004	3.999	1.70	69	288	72.464
2560	2560	16.92088633	2.873e-005	3.992	1.77	88	451	91.925

We apply techniques similar to those discussed above to the TF and AFV models,

Table 5.5: Numerical results for an American put option at $S = 100$ using the PSOR method ($1 < \omega < 2$) to solve the nonlinear equations associated with the free-boundary condition. The numerical method uses linear boundary conditions (5.10), $tol = 1.0\text{e}^{-7}$ and $S_{max} = 400$. Compare our results with the solution 14.67883348 in [Li05] and the solution 14.67882 in [FV02].

Grid	Time	Value at	Difference	Ratio	Avg.	No. of Iterations		
Size	Steps	$S = 100$			ω	Min.	Max.	Avg.
40	40	14.54681346			1.12	6	11	7.875
80	80	14.64461606	9.780e-002		1.24	6	13	9.088
160	160	14.66999545	2.538e-002	3.854	1.37	7	16	10.656
320	320	14.67656676	6.571e-003	3.862	1.51	8	19	12.400
640	640	14.67826506	1.698e-003	3.869	1.64	9	27	14.313
1280	1280	14.67870874	4.437e-004	3.828	1.74	11	45	17.698
2560	2560	14.67881773	1.090e-004	4.071	1.78	12	79	23.569

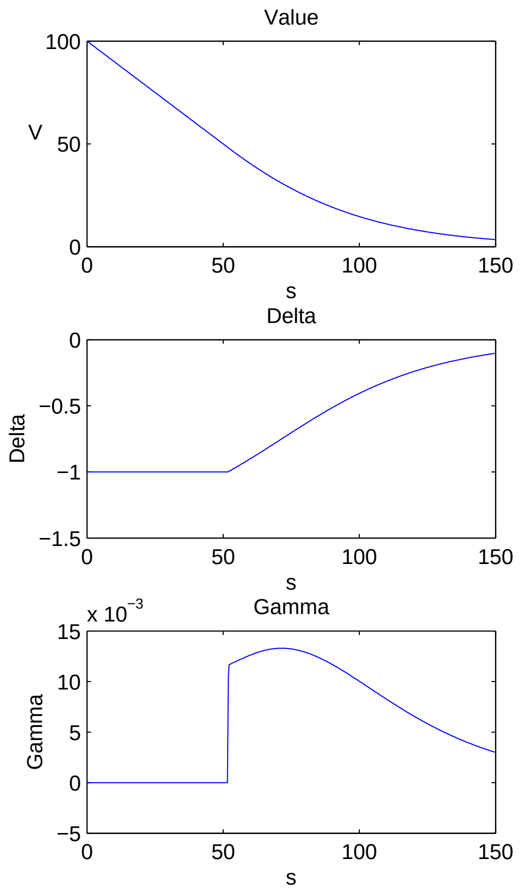


Figure 5.1: The value $V(S,0)$ and the associated Delta and Gamma functions for an American put option at time $t=0$ using PSOR with $n=1280$ and $m=1280$.

since each of the PDEs in each of these models is similar to the PDE (2.2). Under the TF model, the setting of $\mathbf{M}$, α_i, β_i and γ_1, γ_2 and γ_3 for U are the same as those defined in (5.7)-(5.10). The constants are also the same for B, except that γ_1, γ_2 and γ_3 for B are

$$\gamma_1 = \frac{3rS_n}{2h} - (r + r_c), \quad \gamma_2 = -\frac{2rS_n}{h}, \quad \gamma_3 = \frac{rS_n}{2h}. \tag{5.14}$$

For the parameters given in Table 5.2, node 1 only in the discretization suffers from convection-dominance. As discussed above, we use a forward difference for this node only to maintain diagonal dominance. In the later chapters, we see that applying the lower-order approximation at node 1 only does not affect the convergence rate for the the region of interest if the left end is far away from it. We still obtain approximately second-order convergence as expected. (See Tables 5.9 and 5.10.) As for European and American options, $|\gamma_1| = |\frac{3rS_n}{2h} - (r + r_c)| < |\gamma_2| + |\gamma_3|$ at the right end.

Under the AFV model, the parameters α_i, β_i for each of U, B and C are

$$\alpha_i = \frac{\sigma^2 S_i^2}{h_{-1}D_i} - \frac{(r + p\eta)h_1 S_i}{h_{-1}D_i}, \tag{5.15}$$

$$\beta_i = \frac{\sigma^2 S_i^2}{h_1 D_i} + \frac{(r + p\eta)h_{-1} S_i}{h_1 D_i}. \tag{5.16}$$

The γ_1, γ_2 and γ_3 for U and C are

$$\gamma_1 = \frac{3(r + p\eta)S_n}{2h} - (r + p), \quad \gamma_2 = -\frac{2(r + p\eta)S_n}{h}, \quad \gamma_3 = \frac{(r + p\eta)S_n}{2h}, \tag{5.17}$$

For B, they are

$$\gamma_1 = \frac{3(r + p\eta)S_n}{2h} - (r + p(1 - R)), \quad \gamma_2 = -\frac{2(r + p\eta)S_n}{h}, \quad \gamma_3 = \frac{(r + p\eta)S_n}{2h}. \tag{5.18}$$

However, as noted in Table 5.3, in our experiments we use $\eta = 0$ and $R = 0$. Thus, α_i, β_i, γ_2 and γ_3 are the same as those used for European and American options. Similarly, node 1 suffers from convection-dominance. We apply the lower-order approximation at this node only. As for the other problems discussed above, $|\gamma_1| = |\frac{3rS_n}{2h} - (r + p)| < |\gamma_2| + |\gamma_3|$ at the right end.

5.1.2 Numerical Results for Linear Boundary Conditions

Usually, we apply Dirichlet boundary conditions at the two ends of $[0, S_{max}]$ when solving a Black-Scholes-like PDE. However, the Dirichlet boundary conditions are not always easy to derive from financial arguments for complicated derivatives, such as Convertible Bonds. In these more complicated cases, we recommend using linear boundary conditions instead. In Chapter 3, we discussed imposing the conditions (3.20) at the left end $S = 0$ and (3.21) at the right end $S = S_{max}$ to approximate the two boundary conditions. These two boundary conditions (3.20) and (3.21) are derived implicitly from the Black-Scholes PDE itself, rather than from financial arguments directly. The right boundary condition is based on the assumption that V is approximately linear in S as $S \to \infty$.

Now, we want to test how well these linear boundary conditions approximate the real boundary conditions. To keep things simple, we consider a European call option with the parameters shown in Table 5.1, since an analytic solution for the European call option is available (see (2.8)) to check the accuracy of the numerical solution. We also compare numerical solutions based on the Dirichlet boundary conditions, (2.5) and (2.6), with those based on the linear boundary conditions, (3.20) and (3.21). The numerical results are shown in Table 5.6.

We also test linear boundary conditions for American put options. Even though we do not have an analytic solution for the price of an American put option, we can compare our numerical solutions with linear boundary conditions to those based on the Dirichlet boundary conditions, and to the solutions reported in [FV02] and [Li05]. The numerical results are shown in Table 5.7.

From Table 5.6 and Table 5.7, we can see that the numerical values obtained using linear boundary conditions are very close to those obtained using the Dirichlet boundary conditions. For the European call option, the difference is 1.1×10^{-7}, which is of the same magnitude as $tol = 10^{-7}$. The difference between the numerical and analytical solution is of the magnitude of 10^{-5} when $n = m = 2560$. For the American put option, the difference between the solution using the linear boundary conditions and Dirichlet boundary conditions is about 10^{-5}. Both are very close to the solutions reported in [Li05] that use Dirichlet boundary conditions. We also can see that our solutions using the PSOR method and penalty method are close to the ones in [FV02] that use linear boundary conditions. This supports our belief that our numerical methods are accurate. Thus, the linear boundary conditions (3.20) and (3.21) are appropriate approximations to the real boundary conditions. This suggests that we can follow the same approach to approximate the boundary conditions at the ends $S = 0$ and $S = S_{max}$ for a convertible bond, as long as its payoff function is linear in S as $S \to \infty$ and the value of the convertible bond is driven by a well-behaved movement of the underlying, which is true in the TF and AFV models.

Table 5.6: Comparison of the numerical results for a European call option at $S = 100$ based on the linear boundary conditions (5.10) with those based on Dirichlet boundary conditions. In both cases, we used $tol = 1.0\text{e}^{-7}$ and $S_{max} = 400$. The analytical solution at $S = 100$ is given.

Grid Size	Time-Steps	Value at $S = 100$		
(n)	(m)	Linear Bound. Cond.	Dirichlet Bound. Cond.	Analytical
40	40	16.80412510	16.80412510	
80	80	16.89195291	16.89195291	
160	160	16.91368848	16.91368849	
320	320	16.91910899	16.91910899	16.92091465
640	640	16.92046328	16.92046329	
1280	1280	16.92080178	16.92080181	
2560	2560	16.92088633	16.92088644	

Table 5.7: Comparison of the numerical results for an American put option at $S = 100$ based on the linear boundary conditions (5.10) with those based on Dirichlet boundary conditions. In both cases, we used $S_{max} = 400$ and $tol = 1.0\text{e}^{-7}$. The solution is 14.67883348 from [Li05] and 14.67882 from [FV02].

Grid Size (n)	Time-Steps (m)	Value at $S = 100$		
		Linear Boundary Cond.		Dirichlet Boundary Condition (PSOR)
		PSOR	Penalty	
40	40	14.54681346	14.54681347	14.54681346
80	80	14.64461606	14.64461610	14.64461610
160	160	14.66999545	14.66999558	14.66999556
320	320	14.67656676	14.67656709	14.67656705
640	640	14.67826506	14.67826609	14.67826597
1280	1280	14.67870874	14.67871226	14.67871199
2560	2560	14.67881773	14.67883127	14.67883067

5.1.3 Far-Field Effects

In Chapter 3, we discuss the far-field effects and suggest the criterion (3.19) for selecting $S = S_{max}$. Here, we want to test the effects of different values of $S = S_{max}$ on numerical solutions. See Table 5.8 for the American put option, Table 5.9 for the CB using the TF model and Table 5.10 using the AFV model.

For the American put option, we use model parameters given in Table 5.1. Thus, according to (3.19) with $\phi = 4$, $\mu = 3$ and $tol = 10e^{-7}$,

$$\begin{aligned} S_{max} &= \max(\phi X, X\mathrm{e}^{\sqrt{2\sigma^2 T|\ln(tol)|}}, X\mathrm{e}^{(r-\frac{\sigma^2}{2})T+\sigma\mu\sqrt{T}}) \\ &= \max(400, 447, 314). \end{aligned}$$

We test S_{max}=200, 320, 400, and 500 on the American put option. We choose these numbers to ensure the point of interest $S = 100$ is on the grid. The experimental results are given in Table 5.8.

For the CB, we use the parameters given in Table 5.2 for the TF model and Table 5.3 for the AFV model. Choosing $\phi = 4$, $\mu = 3$ and $tol = 10e^{-7}$, we calculate

$$\begin{aligned} S_{max} &= \max(\phi X, X\mathrm{e}^{\sqrt{2\sigma^2 T|\ln(tol)|}}, X\mathrm{e}^{(r-\frac{\sigma^2}{2})T+\sigma\mu\sqrt{T}}) \\ &= \max(400, 533, 444). \end{aligned}$$

We choose S_{max}=300, 400, 500, and 600 for our tests. The results are shown in Tables 5.9 and 5.10.

From Table 5.8, we can see that as S_{max} increases, the convergence rate tends to stabilize and approaches a higher value, but after S_{max} reaches a certain level, the quality of the solution tends to decrease as S_{max} increases further. When $S_{max} = 200$, the convergence ratio is low compared with other values of S_{max} and it decreases as the grid is refined. When $S_{max} = 320$, the convergence ratio is around 3.8 and it is more stable than when $S_{max} = 200$. The solution at $n = 1280$ (14.67876091) is close to the value observed by other authors, 14.67883348 in [Li05] and 14.67882 in [FV02]. For $S_{max} = 400$, the convergence ratio is about 3.85 as the grid is refined. The solution is also close to that reported by other authors. However, for $S_{max} = 500$, because the grid becomes coarser compared with smaller S_{max}, the numerical solutions tend to have bigger errors. Thus, $S_{max} = 400$ is a good choice for this American put option. This suggests that the formula (3.19) gives an appropriate value for S_{max} in this case.

Consider the TF and AFV models for pricing a CB. The model settings are shown in Tables 5.2 and 5.3, respectively. We test $S_{max} = 300$, 400, 500, and 600 for the simple CB, i.e., no coupon payments and no call or put features. See Table 5.9 for the TF model. When S_{max}= 400 or 500, the convergence tends to stable and consistent, but as S_{max} becomes bigger, the errors increase and the convergence ratio oscillates. (See the case of $n = 1280$ when S_{max}.) This is due to the same reason we discussed above. Similarly, the same phenomena appear in the AFV model, which are demonstrated in Table 5.10. Therefore, we choose $S_{max} = 400$ to be the appropriate far-field. This value agrees with the criterion we discussed earlier.

Table 5.8: The numerical results for an American put option at $S = 100$ using the discrete penalty method with different far-field values S_{max}. In all cases, $tol = 1.0\text{e}^{-7}$.

$S = S_{max}$	Grid Size (n)	Time-Steps (m)	Value at $S = 100$	Difference	Ratio
200	40	40	14.64160513		
	80	80	14.66703000	2.542e-002	
	160	160	14.67383615	6.806e-003	3.74
	320	320	14.67565057	1.814e-003	3.75
	640	640	14.67614240	4.918e-004	3.69
	1280	1280	14.67628527	1.429e-004	3.44
320	40	40	14.74429200		
	80	80	14.65668017	8.761e-002	
	160	160	14.67305465	1.637e-002	5.35
	320	320	14.67734077	4.286e-003	3.82
	640	640	14.67846148	1.121e-003	3.82
	1280	1280	14.67876091	2.994e-004	3.74
400	40	40	14.54681347		
	80	80	14.64461610	9.780e-002	
	160	160	14.66999558	2.538e-002	3.85
	320	320	14.67656709	6.572e-003	3.86
	640	640	14.67826609	1.699e-003	3.87
	1280	1280	14.67871226	4.462e-004	3.81
500	40	40	14.47318686		
	80	80	14.62635659	1.532e-001	
	160	160	14.66529207	3.894e-002	3.93
	320	320	14.67537425	1.008e-002	3.86
	640	640	14.67796435	2.590e-003	3.89
	1280	1280	14.67863609	6.717e-004	3.86

Table 5.9: The numerical results for the price of a simple CB (without coupon payments and without any call or put features) under the TF model at $S = 100$ using different far-field values S_{max}. The discrete penalty method with $tol = 1.0\text{e}^{-7}$ was used in the computation.

$S = S_{max}$	Grid Size (n)	Time-Steps (m)	Value at $S = 100$	Difference	Ratio
300	40	40	104.3101724		
	80	80	104.2927398	-1.743e-002	
	160	160	104.2879946	-4.745e-003	3.674
	320	320	104.2868619	-1.133e-003	4.190
	640	640	104.2865719	-2.901e-004	3.905
	1280	1280	104.2865002	-7.166e-005	4.048
400	40	40	104.3269714		
	80	80	104.2970855	-2.989e-002	
	160	160	104.2892061	-7.879e-003	3.793
	320	320	104.2871691	-2.037e-003	3.868
	640	640	104.2866504	-5.187e-004	3.927
	1280	1280	104.2865194	-1.309e-004	3.962
500	40	40	104.3508349		
	80	80	104.3032674	-4.757e-002	
	160	160	104.2907987	-1.247e-002	3.815
	320	320	104.2875744	-3.224e-003	3.867
	640	640	104.2867527	-8.217e-004	3.924
	1280	1280	104.2865451	-2.075e-004	3.959
600	40	40	104.3944690		
	80	80	104.3118939	-8.258e-002	
	160	160	104.2930497	-1.884e-002	4.382
	320	320	104.2880888	-4.961e-003	3.799
	640	640	104.2868827	-1.206e-003	4.113
	1280	1280	104.2865768	-3.059e-004	3.943

Table 5.10: The numerical results for the price of a simple CB (without coupon payments and without any call or put features) under the AFV model at $S = 100$ using different far-field values S_{max}. The discrete penalty method with $tol = 1.0\text{e}^{-7}$ was used in the computation.

$S = S_{max}$	Grid Size	Time-Steps	Value at	Difference	Ratio
	(n)	(m)	$S = 100$		
	40	40	106.2979836		
	80	80	106.3369147	3.893e-002	
300	160	160	106.3474130	1.050e-002	3.708
	320	320	106.3499245	2.512e-003	4.180
	640	640	106.3505667	6.422e-004	3.911
	1280	1280	106.3507254	1.587e-004	4.047
	40	40	106.2539403		
	80	80	106.3267315	7.279e-002	
400	160	160	106.3447778	1.805e-002	4.034
	320	320	106.3492805	4.503e-003	4.008
	640	640	106.3504056	1.125e-003	4.002
	1280	1280	106.3506869	2.813e-004	4.000
	40	40	106.1991674		
	80	80	106.3132778	1.141e-001	
500	160	160	106.3414286	2.815e-002	4.054
	320	320	106.3484440	7.015e-003	4.013
	640	640	106.3501966	1.753e-003	4.003
	1280	1280	106.3506346	4.381e-004	4.001
	40	40	106.1242338		
	80	80	106.2979093	1.737e-001	
600	160	160	106.3371849	3.928e-002	4.422
	320	320	106.3474410	1.026e-002	3.829
	640	640	106.3499386	2.498e-003	4.107
	1280	1280	106.3505711	6.325e-004	3.949

5.1.4 Effects of Nonsmooth Payoff Functions

Payoff functions for options and CBs are often nonsmooth. For example, for European and American options, the payoff function is a hockey-stick function, such as (3.41) for a European call option and (3.45) for an American put option. In the case of convertible bonds under the TF model, the payoff function (3.48) for U is a hockey-stick function, but that for B is a step function, (3.51). For convertible bonds under the AFV model, the payoff functions for both U and C are hockey-stick functions, (3.54) and (3.56), and the payoff function for B is a continuous straight line, (3.59). In this subsection and following subsections, we define a critical point as a kink point for the hockey-stick function or a jump point for the step function.

Now, we want to test the effects of these nonsmooth payoff functions on numerical solutions. Before we study the effects of the payoff functions under the TF and AFV models, we consider the simpler European call option.

In order to clearly see the effects of nonsmooth payoff functions on numerical solutions, we test a particular European call option to exaggerate the effect: $r = 0.05$, $\sigma = 0.25$, $E = 1$ and $T = 2$. In this experiment, we put the kink in the payoff function at a grid point to avoid interpolation, and keep the ratio $\lambda = \frac{dt}{h} = 10$ fixed. Also, $\frac{\sigma^2}{r} > 1$ is used to ensure that rows 1 through $n-1$ of the associated solution matrix are diagonally dominant. Figure 5.2 shows the value of $V(S,0)$ and the associated Delta and Gamma at time $t = 0$ obtained on a uniform grid $0 \leq S \leq S_{max} = 4$ using the parameters above. Figure 5.2(A) shows the results from the Crank-Nicolson method without any enhancements to handle the nonsmooth payoff function. We can observe that the Delta and Gamma have sharp spikes around the region of interest, $S = 1$, which is the location of the discontinuity in the first derivative of the payoff function. Table 5.11 shows the convergence ratios for this numerical method are close to 2, whence the rate of convergence is approximately linear. However, if we enhance our simple Crank-Nicolson numerical method with Rannacher smoothing [Ran84], we can restore the quadratic convergence rate and obtain much smoother Delta and Gamma functions. See Table 5.11 and Figure 5.2(B). A more detailed discussion on Rannacher smoothing is in the next section.

Table 5.11: Comparison of the numerical results for a European call option at $S = 1$ with Rannacher smoothing and without. We ensure that the kink point of the payoff function is at a grid point. $S_{max} = 4$, $\lambda = \frac{\Delta\tau}{h} = 10$, and $tol = 1.0\text{e}^{-7}$.

Gride	Time-	No-Rannacher Smoothing			Rannacher Smoothing		
Size	Steps	Value	Diff.	Ratio	Value	Diff.	Ratio
80	4	0.18186118			0.18393727		
160	8	0.18414572	2.285e-003		0.18583911	1.902e-003	
320	16	0.18530699	1.161e-003	1.967	0.18631039	4.713e-004	4.035
640	32	0.18588911	5.821e-004	1.995	0.18643016	1.198e-004	3.935
1280	64	0.18618006	2.910e-004	2.001	0.18646053	3.037e-005	3.944
2560	128	0.18632545	1.454e-004	2.001	0.18646819	7.659e-006	3.965

Now we consider the initial conditions under the TF and AFV models. The initial condition for the TF model is a hockey-stick function for U and a step function for B. We have discussed the effects of a hockey-stick function above. Now we test the effects of a step function on the numerical solution of the TF model using the Crank-Nicolson method. If we place the point of discontinuity of the step function and the kink in the hockey-stick function on the grid, only linear convergence is observed in both U and B even though we use the Crank-Nicolson method with Rannacher smoothing. See Table 5.12. Notice that with Rannacher smoothing, the convergence rate is enhanced and the value of B is more accurate than the corresponding value of B in the case without Rannacher smoothing, the associated Delta and Gamma are much smoother around the region of discontinuity (i.e., $S = 100$), shown in Figure 5.3. However, the

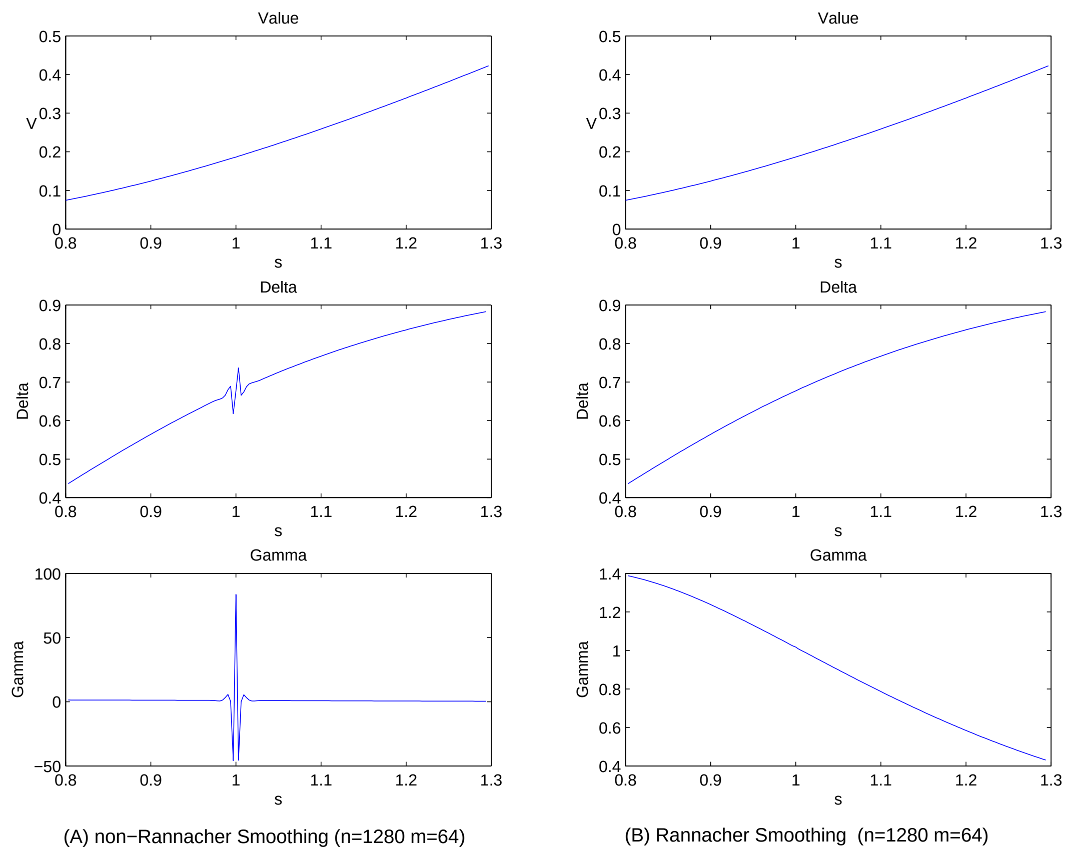


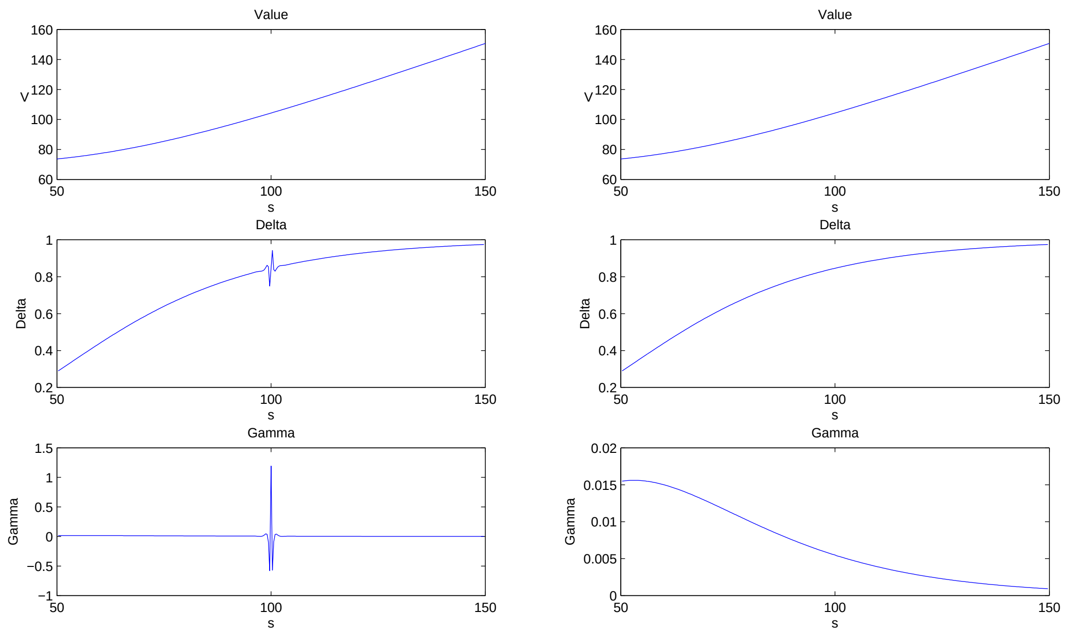
Figure 5.2: The value $V(S, 0)$ and the associated Delta and Gamma functions for a European Call option with a nonsmooth hockey-stick payoff function. The parameters are $r = 0.05$, $\sigma = 0.25$, $E = 1$, and T=2. (A) shows the numerical results without Rannacher smoothing; (B) is with Rannacher smoothing.

numerical solutions for both U and B exhibit a linear rate of convergence, even though we use Rannacher smoothing. (Recall that the convergence is restored to quadratic when applying Rannacher smoothing for a hockey-stick initial condition in the case of the European call option.) We believe this is because the initial condition for B is a step function. The convergence rate for B is approximately linear whether we use Rannacher smoothing or not. We believe the convergence rate of U is affected by that of B, since B is coupled to U in the PDE (4.61). Thus, the convergence rate for U is linear too. We discuss how to remedy this convergence degradation in section 5.2.2.

For the AFV model, the Crank-Nicolson method without any enhancement to handle nonsmooth initial functions achieves a linear rate of convergence for U and C, even though it attains a second-order rate of convergence for B. See Table 5.13 and Figure 5.4. We believe this is because the initial functions for U and C are hockey-stick functions and hence only linear convergence is obtained for reasons similar to those discussed above for the European and American options. On the other hand, we believe a second-order rate of convergence is attained for B because its initial function is smooth and the coupling from U and C to B is weak. (The coupling is through the boundary conditions only, not through the PDE (2.32) for B.) However, unlike the TF model, a quadratic convergence rate is achieved in the AFV mode when we apply Rannacher smoothing to the Crank-Nicolson method, because the initial condition for B is smooth and the initial conditions for U and C are hockey-stick functions. With Rannacher smoothing, U, B and C all achieve a quadratic convergence rate. Moreover, with Rannacher smoothing, Figure 5.4(B) shows that the Delta and Gamma functions associated with U are smooth around the region of interest.

Table 5.12: Comparison of the numerical results for the Crank-Nicolson method with Rannacher smoothing and without for a simple CB (without coupon payments and without any call/put features) at $S = 100$ under the TF model. We put the critical points of the initial functions at a grid point, and keep $\lambda = 0.25$ constant. Here, $tol = 1.0\text{e}^{-7}$.

Grid	Time-	No-Rannacher Smoothing			Rannacher Smoothing		
Size	Steps	U			U		
(n)	(m)	Value	Diff.	Ratio	Value	Diff.	Ratio
80	4	103.4072213			103.9181577		
160	8	103.8394829	4.323e-001		104.1562642	2.381e-001	
320	16	104.0615871	2.221e-001	1.946	104.2341716	7.791e-002	3.056
640	32	104.1737489	1.122e-001	1.980	104.2635891	2.942e-002	2.648
1280	64	104.2300502	5.630e-002	1.992	104.2758665	1.228e-002	2.396
2560	128	104.2580250	2.797e-002	2.013	104.2813402	5.474e-003	2.243
Grid	Time-	B			B		
Size	Steps	Value	Diff.	Ratio	Value	Diff.	Ratio
80	4	46.7777373			28.0884960		
160	8	45.6834053	-1.094e+000		26.8641499	-1.224e+000	
320	16	45.1145209	-5.689e-001	1.924	26.3844959	-4.797e-001	2.553
640	32	44.8254119	-2.891e-001	1.968	26.1729041	-2.116e-001	2.267
1280	64	44.6798063	-1.456e-001	1.986	26.0737563	-9.915e-002	2.134
2560	128	44.6060374	-7.377e-002	1.974	26.0256462	-4.811e-002	2.061



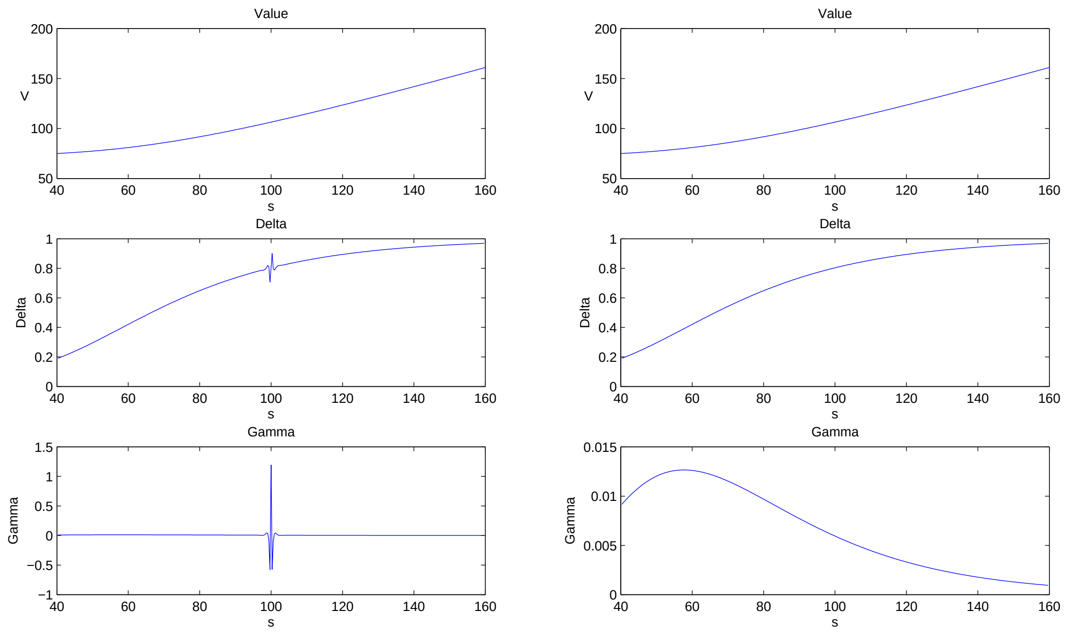
(A) Non−Rannacher smoothing (n=1280 m=64)

(B) Rannacher smoothing (n=1280 m=64)

Figure 5.3: Comparison of $U(S, 0)$ and the associated Delta and Gamma functions for a simple CB (without coupon payments and without any call/put features) at $S = 100$ under the TF model computed by the Crank-Nicolson method with Rannacher smoothing and without. We put the critical points of the initial functions at a grid point, and keep $\lambda = 0.25$ constant. (A) shows the numerical results without Rannacher smoothing; (B) is with Rannacher smoothing.

Table 5.13: Comparison of the numerical results for the Crank-Nicolson with Rannacher smoothing and without for a simple CB (without coupon payments and without any call/put features) at $S = 100$ under the AFV model. We put the discontinuity points of the initial functions at a grid point, and keep $\lambda = 0.25$ constant. Here, $tol = 1.0\mathrm{e}^{-7}$

Grid Size (n)	Time-Steps (m)	No-Rannacher Smoothing			Rannacher Smoothing		
		U			U		
		Value	Differ.	Ratio	Value	Diff.	Ratio
80	4	105.6195104			106.2478855		
160	8	105.9799950	3.605e-001		106.3258636	7.798e-002	
320	16	106.1644959	1.845e-001	1.954	106.3443383	1.847e-002	4.221
640	32	106.2574730	9.298e-002	1.984	106.3491197	4.781e-003	3.864
1280	64	106.3040930	4.662e-002	1.994	106.3503573	1.238e-003	3.863
2560	128	106.3274155	2.332e-002	1.999	106.3506642	3.069e-004	4.033
Grid Size	Time-Steps	B			B		
		Value	Diff.	Ratio	Value	Diff.	Ratio
80	4	70.4530564			70.9727099		
160	8	70.4648739	1.182e-002		70.5970465	-3.757e-001	
320	16	70.4678254	2.951e-003	4.004	70.5011924	-9.585e-002	3.919
640	32	70.4685631	7.377e-004	4.001	70.4769481	-2.424e-002	3.954
1280	64	70.4687475	1.844e-004	4.000	70.4708493	-6.099e-003	3.975
2560	128	70.4687936	4.610e-005	4.000	70.4693198	-1.530e-003	3.987
Grid Size	Time-Steps	C			C		
		Value	Differ.	Ratio	Value	Differ.	Ratio
80	4	35.1664539			35.2751756		
160	8	35.5151211	3.487e-001		35.7288170	4.536e-001	
320	16	35.6966706	1.815e-001	1.921	35.8431460	1.143e-001	3.968
640	32	35.7889099	9.224e-002	1.968	35.8721716	2.903e-002	3.939
1280	64	35.8353455	4.644e-002	1.986	35.8795080	7.336e-003	3.956
2560	128	35.8586219	2.328e-002	1.995	35.8813444	1.836e-003	3.995



(A) Non−Rannacher Smoothing (n=1280 m=64)

(B) Rannacher Smoothing (n=1280 m=64)

Figure 5.4: Comparison of $U(S, 0)$ and the associated Delta and Gamma functions for a simple CB (without coupon payments and without any call/put features) at $S = 100$ under the AFV model computed by the Crank-Nicolson method with Rannacher Smoothing and without. We put the discontinuity points of the initial functions at a grid point, and keep $\lambda = 0.25$ constant. (A) shows the numerical solutions without Rannacher smoothing; (B) is with Rannacher smoothing.

5.2 Improvements

5.2.1 Rannacher Smoothing and Controlling $\lambda = \frac{\Delta\tau}{h}$ or $\Gamma = \frac{\Delta\tau}{h^2}$

Even though the second-order accurate Crank-Nicolson is unconditionally stable, it is not strongly A-stable [FV02]. That means that the method does not rapidly damp out high frequency oscillatory components in the initial values, whence it may produce oscillations in the numerical solution, especially when the initial condition is a non-smooth function. This is the source of the spikes in the Delta and Gamma functions in Figure 5.2. To avoid these undesired oscillations associated with the Crank-Nicolson method, we augment the basic method with Rannacher time-stepping [Ran84]. That is, we take the first few time-steps of the numerical integration with the fully implicit method and then switch to the the Crank-Nicolson method for the remaining time-steps. This enhanced method damps out the high frequency error components, and thus helps to eliminate oscillations in the solutions that arise from the nonsmooth initial conditions. Moreover, it restores the second-order rate of convergence associated with the Crank-Nicolson method since it damps the high frequency errors and only a few first-order fully-implicit steps are taken. However, if the initial function is a step function, only first-order accuracy is observed for the Crank-Nicolson method even if Rannacher smoothing is used, as we have seen in section 5.1.4. This degradation in the order is not due solely to the high frequency error components discussed above; other errors are also important here, as we discuss in more detail in the next section.

In subsection 5.1.4, we discussed the effects of the two types of nonsmooth initial functions on numerical solutions. When we take constant time-steps with the Crank-Nicolson method, two ways are recommended in [FV02] to reduce oscillations while maintaining second-order accuracy depending on the type of the initial function: (i) Rannacher time-stepping should be used for a nonsmooth function, i.e., first we take *two* fully implicit time-steps after each nonsmooth initial state, and then we use Crank-Nicolson time-steps thereafter; and (ii) if the initial condition can be approximated by a continuous piecewise linear function such as a hockey-stick function in our case, we choose the grid so that the kink points occur at grid points.

From Table 5.11 for the European call option, we see that Rannacher smoothing helps to eliminate the oscillations and restores the convergence rate to second-order. From Figures 5.2(B), we can see that the option price and associated Delta and Gamma are smooth after applying the Rannacher time-stepping, in contrast to Figure 5.2(A), where Rannacher smoothing is not used. Even though we put the kink point at a grid point (i.e., method (ii)), the oscillations still occur in Figure 5.2(A) and only linear convergence is observed. Thus, (ii) may not be as effective as (i) in reducing oscillation problems but it may help to a limited extent. Similar observations apply to our numerical results for convertible bonds under both the TF and AFV models. See Table 5.12 and Figure 5.3 for the TF model, and Table 5.13 and Figure 5.4 for the AFV model.

Now, we study the effects of the ratio $\lambda = \frac{\Delta\tau}{h}$ on convergence and oscillations. Using the same parameters as in subsection 5.1.4 for the European option, we test different values of λ. The results for λ =5, 3 and 2.5 are shown in Table 5.14. The results for $\lambda = 10$ are given in Table 5.11. We can see that if $\lambda > 3$, the convergence rate is approximately linear and oscillations occur around the kink point. If $\lambda \leq 3$ in this particular case, the convergence rate is enhanced and is approximately quadratic. As well, we see from Figure 5.5 that the Delta and Gamma approximations become smoother as λ decreases. In plots (A) of Figure 5.5, the Delta approximation is smooth, but the Gamma approximation is rough around the kink point. In plots (B), both the Delta and Gamma approximations are smooth. Thus, we can conclude that the smaller λ helps to reduce oscillations. In the previous experiments for the European and American options with the parameters listed in Table 5.1, we chose $S_{max} = 400$, therefore $\lambda = \frac{0.25n}{400m} = 0.000625$ when $n = m$. For this small value of λ, the experiments show that numerical solutions converge, and the Delta and Gamma approximations are smooth at the kink point, even without Rannacher smoothing. (See Figure 5.1.)

Besides studying λ, we also are interested in $\Gamma = \frac{\Delta\tau}{h^2}$. Table 5.15 and Figure 5.6 show the numerical results for the same European option discussed above, but, we keep Γ fixed, rather than λ, this time. The numerical results show that the Delta and Gamma approximations are smooth and convergence rate is effectively restored to quadratic. Thus, this is also considered to be an alternative to reduce oscillations. Notice that λ is cut in half when n and m are doubled. Thus, keeping Γ constant, reduces λ as the grid is refined. This approach of keeping Γ constant as the grid is refined is generally much more expensive than the other approaches to reduce oscillations discussed in this section. Hence, we recommend it as a last resort only.

Table 5.14: Comparison of the numerical results using the Crank-Nicolson method for a European call option at $S = 1$ with and without Rannacher smoothing and with different values of the ratio $\lambda = \frac{\Delta\tau}{h}$. We put the kink point of the payoff function at a grid point. $S_{max} = 4$ and $tol = 1.0\text{e}^{-7}$.

λ	Grid	Time-	No-Rannacher Smoothing			Rannacher Smoothing		
$\frac{\Delta\tau}{h}$	Size	Steps	Value	Diff.	Ratio	Value	Diff.	Ratio
	40	4	0.18484582			0.18287543		
	80	8	0.18588186	1.036e-003		0.18558598	2.711e-003	
5	160	16	0.18623657	3.547e-004	2.921	0.18624774	6.618e-004	4.096
	320	32	0.18636938	1.328e-004	2.671	0.18641453	1.668e-004	3.968
	640	64	0.18642409	5.470e-005	2.428	0.18645662	4.209e-005	3.962
	1280	128	0.18644844	2.435e-005	2.478	0.18646721	1.059e-005	3.975
	24	4	0.18292849			0.18034360		
	48	8	0.18560609	2.678e-003		0.18498528	4.642e-003	
3	96	16	0.18625485	6.488e-004	4.127	0.18609925	1.114e-003	4.167
	192	32	0.18641638	1.615e-004	4.016	0.18637749	2.782e-004	4.004
	384	64	0.18645694	4.056e-005	3.983	0.18644737	6.988e-005	3.982
	768	128	0.18646719	1.025e-005	3.957	0.18646490	1.753e-005	3.986
	20	4	0.18114364			0.17838958		
	40	8	0.18519185	4.048e-003		0.18456419	6.175e-003	
2.5	80	16	0.18615396	9.621e-004	4.208	0.18599657	1.432e-003	4.311
	160	32	0.18639172	2.378e-004	4.047	0.18635197	3.554e-004	4.030
	320	64	0.18645100	5.928e-005	4.011	0.18644100	8.902e-005	3.992
	640	128	0.18646581	1.481e-005	4.001	0.18646331	2.231e-005	3.990

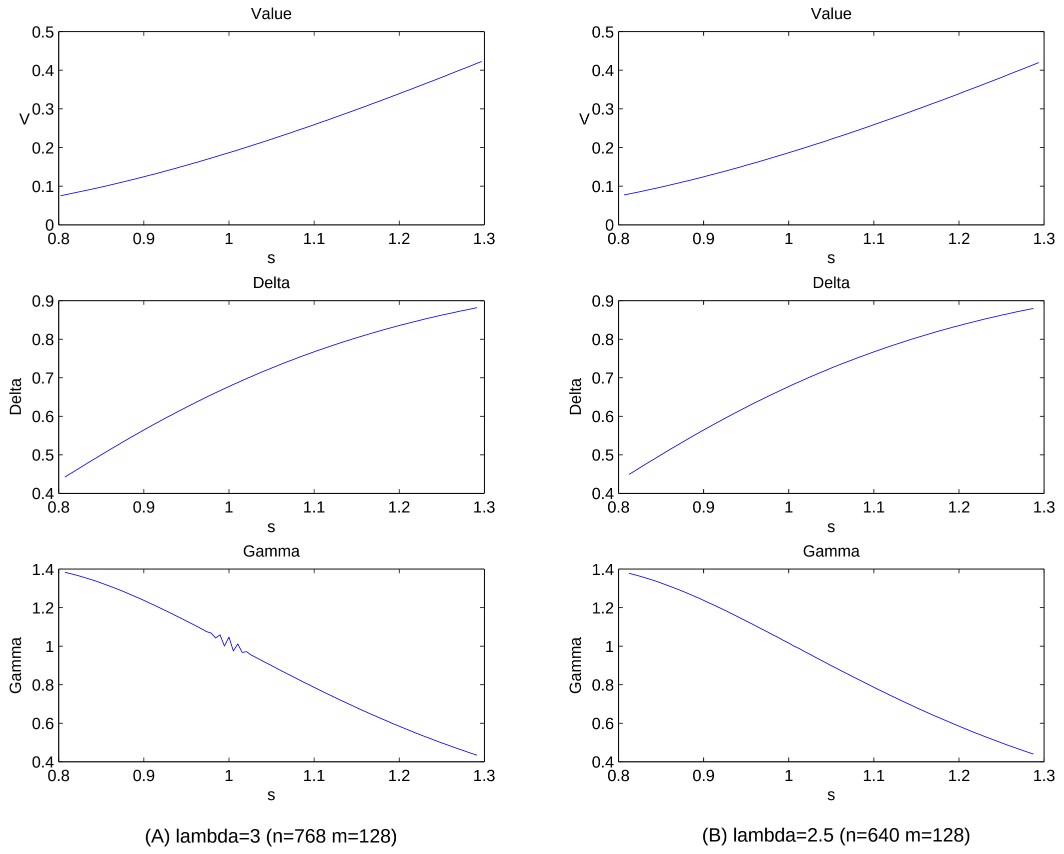


Figure 5.5: The value $V(S,0)$ and the associated Delta and Gamma functions for a European call option with a piecewise linear payoff function. The parameters are $r = 0.05$, $\sigma = 0.25$, $E = 1$, and T=2. The Crank-Nicolson method without Rannacher smoothing is used. We put the kink point of the payoff function at a grid point and keep $\lambda = \frac{\Delta\tau}{h}$ constant. For plots (A), $\lambda = 3$. For plots (B), $\lambda = 2.5$.

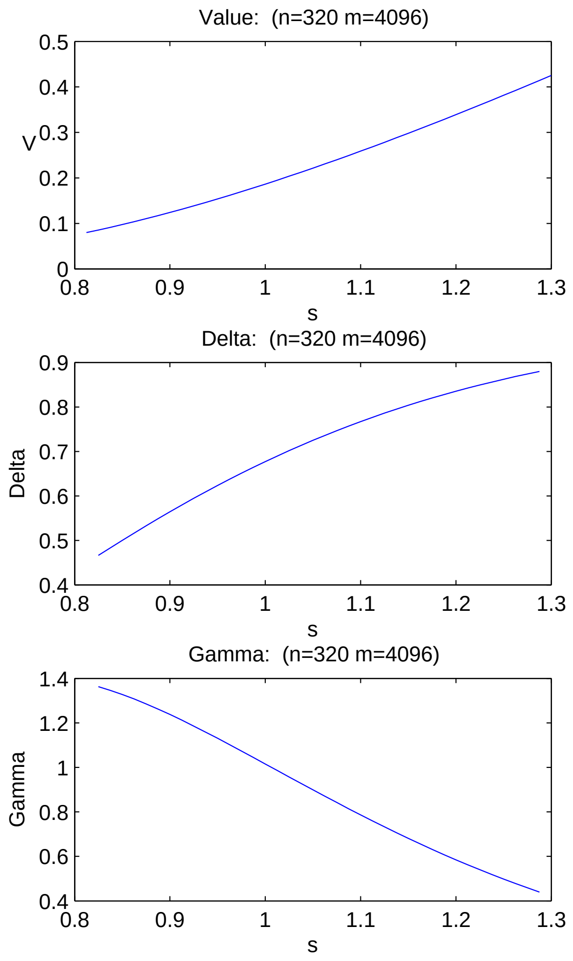


Figure 5.6: The value $V(S,0)$ and the associated Delta and Gamma functions for a European call option with a piecewise linear payoff function. Parameters are $r = 0.05$, $\sigma = 0.25$, $E = 1$, and T=2. The Crank-Nicolson method without Rannacher smoothing is used and we put the kink point of the payoff function at a grid. Here, $\Gamma = 3.125$ and $\lambda = 0.039$.

Table 5.15: The numerical results for the Crank-Nicolson method without Rannacher smoothing for a European call option at $S = 1$. We put the kink point of the payoff function at a grid point and keep $\Gamma = \frac{\Delta\tau}{h^2}$ constant. $S_{max} = 4$ and $tol = 1.0\text{e}^{-7}$.

Γ	λ	Gride	Time-	No-Rannacher Smoothing		
$\frac{\Delta\tau}{h^2}$	$\frac{\Delta\tau}{h}$	Size	Steps	Value	Diff.	Ratio
	1.250	10	4	0.16163256		
	0.625	20	16	0.18078626	1.915e-002	
3.125	0.313	40	64	0.18511910	4.333e-003	4.421
	0.156	80	256	0.18613627	1.017e-003	4.260
	0.078	160	1024	0.18638734	2.511e-004	4.051
	0.039	320	4096	0.18644992	6.258e-005	4.012

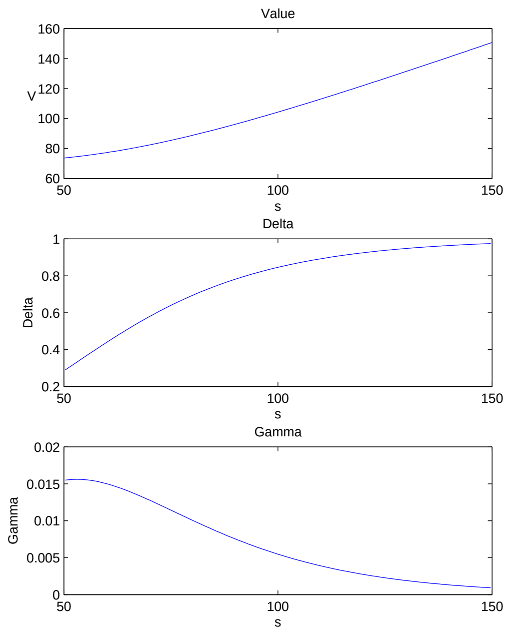
5.2.2 Grid Positioning

From subsections 5.1.4 and 5.2.1, we can see that using Rannacher time-stepping, controlling λ or Γ, or putting the kink in a piecewise linear payoff function at a grid point helps remove the oscillations occurring in the numerical solutions, and thus improves the convergence rate to the method. However, if the initial function is a step function, the Rannacher smoothing may eliminate the oscillations, but it does not help much to improve the convergence rate. Even with Rannacher smoothing, first-order accuracy is observed when the initial function is a step function. This is shown in Tables 5.16 and 5.12 for a convertible bond under the TF model. When λ is reduced from 0.25 to 0.01, the oscillations are removed (See Figures 5.7(A) and 5.3(A)), but the convergence rate remains first-order, even if Rannacher smoothing is applied. However, for the AFV model, after reducing λ to 0.01, even without applying Rannacher smoothing, the oscillations are removed and the convergence rate is restored to the second-order. (See Table 5.17 and Figure 5.8.) This is because the TF model is associated with a step initial function while the AFV model is not.

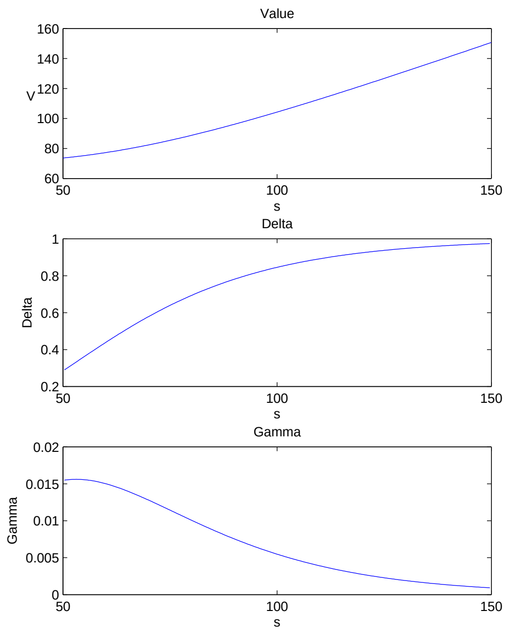
Tavella and Randall conclude in [TR00] that quantization errors caused by non-smooth initial functions seriously degrade convergence rates of numerical method. They suggest that grid positioning can increase accuracy. By shifting the grid such that discontinuities occur midway between grid points and at each grid refinement aligning the grid once again to maintain this desired property of the grid, the quantization error can be reduced substantially and quadratic convergence can be restored. The numerical results shown in Table 5.18 illustrate the effectiveness of this approach. After applying grid positioning for the step function, the convergence rates for U and B are approximately quadratic. This should be contrasted to the approximately linear convergence rate shown in Table 5.16, where grid positioning is not used. Also, grid positioning does not affect the approximate Delta and Gamma functions. They remain smooth as illustrated in Figure 5.9.

Table 5.16: Comparison of the numerical results for the Crank-Nicolson with and without Rannacher smoothing for a simple CB (without coupon payments and without any call/put features) at $S = 100$ under the TF model. We put the critical point of the initial functions at a grid point, and keep $\lambda = 0.01$ constant. Here, $tol = 1.0\text{e}^{-7}$.

Grid Size	Time-Steps	No-Rannacher Smoothing			Rannacher Smoothing		
		U			U		
		Value	Diff.	Ratio	Value	Diff.	Ratio
40	40	103.8694109			103.8669642		
80	80	104.1045491	2.351e-001		104.1039814	2.370e-001	
160	160	104.2019810	9.743e-002	2.413	104.2018439	9.786e-002	2.422
320	320	104.2458269	4.385e-002	2.222	104.2457933	4.395e-002	2.227
640	640	104.2665488	2.072e-002	2.116	104.2665406	2.075e-002	2.118
1280	1280	104.2766113	1.006e-002	2.059	104.2766093	1.007e-002	2.061
Grid Size	Time-Steps	B			B		
		Value	Diff.	Ratio	Value	Diff.	Ratio
40	40	28.9369847			28.9449665		
80	80	27.4585776	-1.478e+000		27.4601910	-1.485e+000	
160	160	26.7198612	-7.387e-001	2.001	26.7202185	-7.400e-001	2.007
320	320	26.3497815	-3.701e-001	1.996	26.3498641	-3.704e-001	1.998
640	640	26.1644492	-1.853e-001	1.997	26.1644688	-1.854e-001	1.998
1280	1280	26.0716957	-9.275e-002	1.998	26.0717004	-9.277e-002	1.998



(A) Non−Rannacher Smoothing (n=1280 m=1280)



(B) Rannacher Smoothing (n=1280 m=1280)

Figure 5.7: Comparison of $U(S, 0)$ and the associated Delta and Gamma functions of a simple CB (without coupon payments and without any call/put features) at $S = 100$ under the TF model computed by the Crank-Nicolson method with and without Rannacher smoothing. We put the critical point of the initial functions at a grid point, and keep $\lambda = 0.25$ constant. Plots (A) show the numerical results with Rannacher smoothing; Plots (B) are without Rannacher smoothing.

Table 5.17: Numerical results for the Crank-Nicolson method without Rannacher smoothing for a simple CB (without coupon payments and call/put features) at $S = 100$ under the AFV model. We put the critical point of the initial functions at a grid point, and keep $\lambda = 0.01$ constant. Here, $tol = 1.0\text{e}^{-7}$.

Grid Size	Time-Steps	U		
(n)	(m)	Value	Difference	Ratio
40	40	106.2539403		
80	80	106.3267315	7.279e-002	
160	160	106.3447778	1.805e-002	4.034
320	320	106.3492805	4.503e-003	4.008
640	640	106.3504056	1.125e-003	4.002
1280	1280	106.3506869	2.813e-004	4.000
Grid Size	Time-Steps	B		
(n)	(m)	Value	Difference	Ratio
40	40	70.4740237		
80	80	70.4701155	-3.908e-003	
160	160	70.4691360	-9.795e-004	3.990
320	320	70.4688908	-2.452e-004	3.995
640	640	70.4688294	-6.134e-005	3.997
1280	1280	70.4688141	-1.534e-005	3.999
Grid Size	Time-Steps	C		
(n)	(m)	Value	Difference	Ratio
40	40	35.7799167		
80	80	35.8566160	7.670e-002	
160	160	35.8756418	1.903e-002	4.031
320	320	35.8803897	4.748e-003	4.007
640	640	35.8815762	1.186e-003	4.002
1280	1280	35.8818728	2.966e-004	4.000

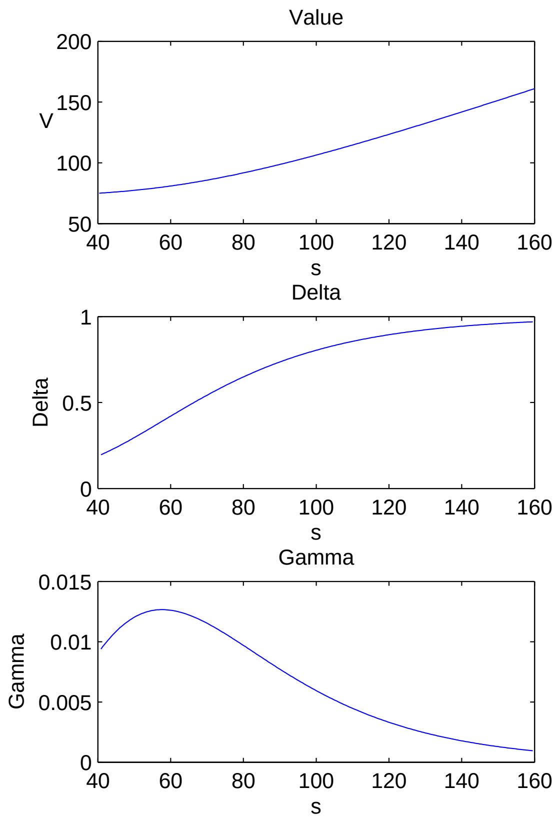
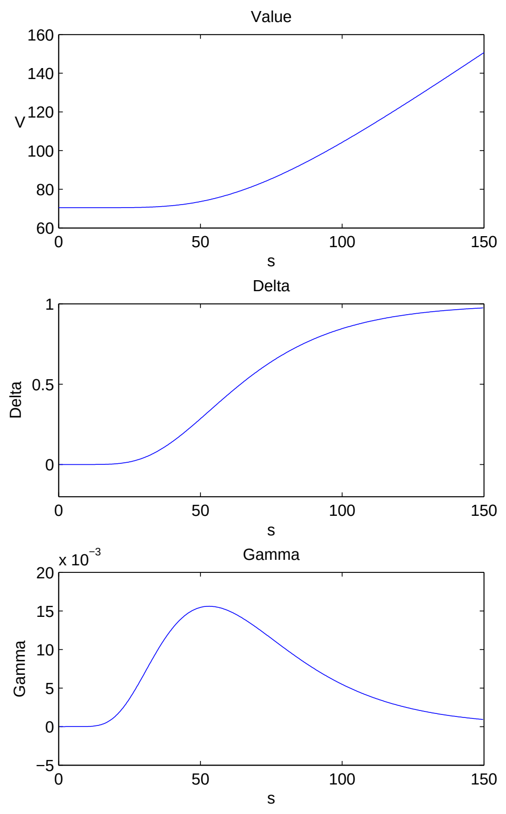
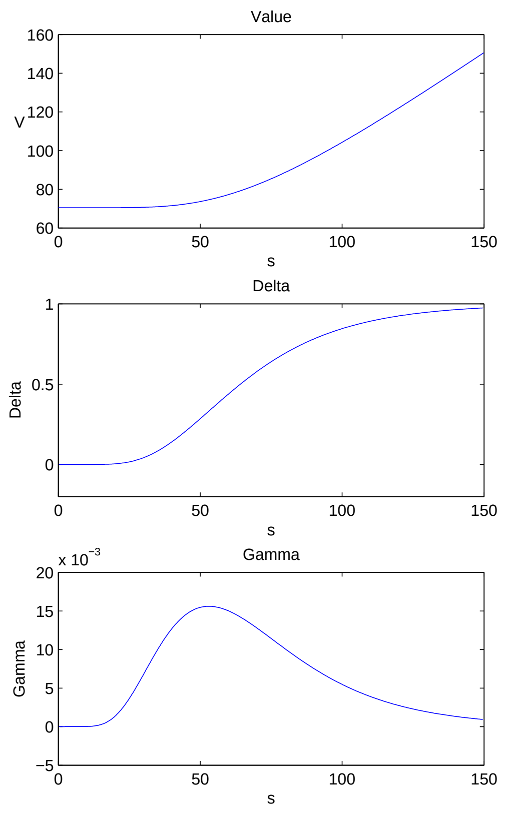


Figure 5.8: The value $U(S,0)$ and the associated Delta and Gamma functions for a simple CB (without coupon payments and without any call/put features) under the AFV model. We used the Crank-Nicolson method without Rannacher smoothing and we put the critical point of the initial function at a grid point. $F = 100$, $n = 1280$, and $m = 1280$.



(A) non−Grid positioning (n=1280 m=1280)



(B) Grid positioning (n=1280 m=1280)

Figure 5.9: The value $U(S,0)$ and the associated Delta and Gamma functions for a CB under the TF model for which the associated U has a piecewise linear initial function and B has a step initial function. The Crank-Nicolson method with Rannacher smoothing is used. $\lambda = \frac{\Delta\tau}{h} = 0.01$.

Table 5.18: Numerical results for the Crank-Nicolson method without Rannacher smoothing, but using grid positioning for B for a simple CB (without coupon payments and without any call/put features) under the TF model at $S = 100$. Here, $\lambda = 0.01$ and $tol = 1.0\text{e}^{-7}$.

Grid	Time-	U			B		
Size	Steps	Value	Diff.	Ratio	Value	Diff.	Ratio
40	40	104.3269714			26.0958267		
80	80	104.2970855	-2.989e-002		26.0080977	-8.773e-002	
160	160	104.2892061	-7.879e-003	3.793	25.9862282	-2.187e-002	4.011
320	320	104.2871691	-2.037e-003	3.868	25.9807240	-5.504e-003	3.973
640	640	104.2866504	-5.187e-004	3.927	25.9793410	-1.383e-003	3.980
1280	1280	104.2865194	-1.309e-004	3.962	25.9789944	-3.466e-004	3.990

5.2.3 Removing Oscillations in Penalty Methods

The PSOR method is often used in numerical schemes for pricing American-style options, but the big disadvantage of the PSOR method is that the number of iterations may grow rapidly as the discretization is refined, as reported in [Li05, Mo06]. This makes the PSOR method computationally inefficient for finer discretizations. On the other hand, the discrete penalty method transforms a constrained linear system into a nonlinear system, and solves it using Newton's method at each time-step. The discrete penalty method is often much more efficient than PSOR. However, the discrete penalty method may excite oscillations in the numerical solution which affect the convergence rate of the numerical method. Moreover, in some situations, Newton's method does not converge due to oscillations appearing in the penalty matrix $\mathbf{P}$. Thus, we suggest using the continuous penalty method [NST02] to avoid this problem. We implemented the discrete and the continuous penalty methods for an American put option and present the results in Tables 5.19 and 5.20, respectively. Using the continuous penalty method results in a better convergence rate than using the discrete penalty method. The convergence rate is quadratic for the continuous penalty method. From Tables 5.19 and 5.20, we notice that the two solutions are different in the first decimal, with the value of the solution by the continuous method being smaller. This disagreement of the two results may be caused by the different numerical methods. Unfortunately, [NST02] does not give option values computed; it only reports errors and convergence rates. Therefore, we cannot compare the values we calculated with any values in [NST02]. Thus, we just report the results here for future reference. Furthermore, we can see from these two tables that the number of iterations using the continuous penalty method is close to that using the discrete penalty method. We believe that the continuous penalty method warrants further research.

Table 5.19: Numerical results at $S = 100$ using the Crank-Nicolson method with the discrete penalty method for an American put option. Here, $tol = 1.0\text{e}^{-7}$.

Grid	Time-				No. of Iterations		
Size	Steps	Value	Difference	Ratio	Min.	Max.	Avg. (A)
40	40	14.54681348			1	2	1.100
80	80	14.64461610	9.780e-002		1	2	1.113
160	160	14.66999558	2.538e-002	3.85	1	3	1.106
320	320	14.67656709	6.572e-003	3.86	1	3	1.109
640	640	14.67826609	1.699e-003	3.87	1	3	1.111
1280	1280	14.67871226	4.462e-004	3.81	1	4	1.113

5.2.4 Adaptive Spatial Grids

We adapt the spatial grid in the region around the discontinuity, to reduce errors and more accurately locate the free boundary point. At each time step, we must find the free

Table 5.20: Numerical results at $S = 100$ using the Crank-Nicolson method with the continuous penalty method for an American put option. Here, $tol = 1.0\text{e}^{-7}$ and $\chi = 2rE$.

Grid	Time-				No. of Iterations		
Size	Steps	Value	Difference	Ratio	Min.	Max.	Avg.
40	40	14.33250099			2	3	2.025
80	80	14.42229483	8.979e-002		2	3	2.013
160	160	14.44451510	2.222e-002	4.04	2	3	2.006
320	320	14.45005696	5.542e-003	4.01	2	3	2.013
640	640	14.45143903	1.382e-003	4.01	2	6	2.028
1280	1280	14.45178546	3.464e-004	3.99	2	6	2.014

boundary point and calculate the solution V simultaneously. The accuracy with which we locate the free boundary is related to the quality of the computed price. To locate the free boundary in a more accurate way, we refine the grid near the free boundary, leaving a coarse grid in other regions. See Figure 5.10.

In brief, we refine the grid as follows. At time $\tau = 0$, we know the free boundary coincides with the exercise price, E, which is normally a grid point, say S_i, on the original coarse grid. We typically refine the grid in the interval $[S_{i-2}, S_{i+2}]$, where S_{i-2} and S_{i+2} are grid points on the original coarse grid. That is, we typically start by refining the grid in two intervals on either side of $E = S_i$. The refined grid normally has one new grid point halfway between every two original grid points. Thus the distance between grid points in the refined region is half the distance between grid points in the unrefined region. Figure 5.10 shows a refined grid on the interval $[S_{i-1}, S_{i+1}]$, where S_{i-1} and S_{i+1} are grid points on the original coarse grid.

As τ increases, we keep the number of additional grid points fixed, but, as explained in more detail below, we move the refined region to follow the free boundary and also

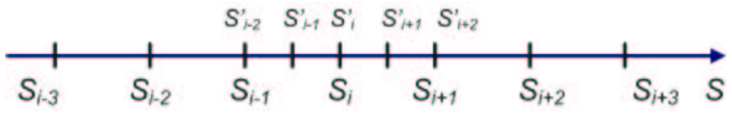


Figure 5.10: Adaptive spatial grids.

narrow the refined region as the free boundary curve flattens. See Figure 5.11. We leave the grid in other regions unchanged, putting the additional points around the free boundary only.

Now, we describe in more detail how to determine how to move the refined region to follow the free boundary and how to narrow the refined region as the free boundary curve flattens. At $\tau = 0$, we choose an initial refined region centered at $E = S_i$, as described above. We refer to this point as S_{i_0} below to emphasize that it is a point on the free boundary curve at time $\tau = 0$. For the lack of a better choice, we predict that the free boundary will remain at S_{i_0} at time τ_1. Then we take the time step from time $\tau = 0$ to time τ_1 and find the approximation S_{i_1} to the free boundary at time τ_1. The point S_{i_1} will be a grid point typically in the refined region, $[S_{i-2}, S_{i+2}]$. However, S_{i_1} may be either a point on the original grid or a new point on the refined grid.

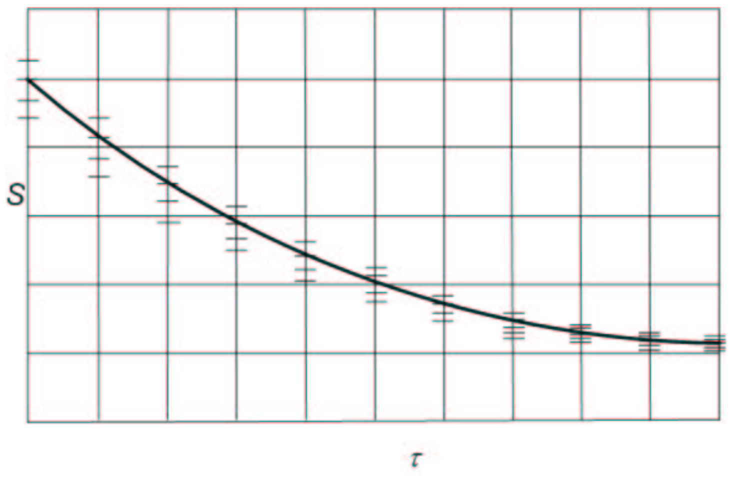


Figure 5.11: The spatial grid near the free boundary is refined as τ increases.

On the next step to time τ_2, we use linear interpolation through the points S_{i_0} at time

$\tau = 0$ and S_{i_1} at time τ_1 to predict the free boundary position $\hat{S}_{i_2}$ at time τ_2. Normally $\hat{S}_{i_2}$ is a grid point either on the original grid or the refined grid. For the time step to τ_2, we center the refined grid at $\hat{S}_{i_2}$ and decrease its width if $|S_{i_0} - S_{i_1}|$ is less than half the distance between grid points on our original unrefined spatial mesh. This allows the refined region to narrow as the free boundary curve flattens. See Figure 5.11. Then we take the time step from time τ_1 to time τ_2 and find the approximation S_{i_2} to the free boundary at time τ_2. The point S_{i_2} will be a grid point typically in the refined region. However, S_{i_2} may be either a point on the original grid or a new point on the refined grid.

Each time step from τ_j to τ_{j+1} for $j \geq 2$ follows the approach described in the preceding paragraph. We first use the approximations to the free boundary curve at times τ_{j-1} and τ_j to predict the free boundary point at time τ_{j+1}, we then possibly adjust the refined region and finally we take the time step from τ_j to τ_{j+1} and find an approximation to the free boundary at time τ_{j+1}.

Table 5.21 shows the numerical results computed on a uniform spatial grid for the American put option which uses the parameters in Table 5.1. Table 5.22 shows the numerical results for the same option but with the adaptive spatial grid described above. Using the adaptive spatial grid with $m = 84$, we compute $V = 14.67871967$, which compares favorably with $V = 14.67871226$ computed on a uniform grid with $m = 1280$. Both have about four digits of accuracy. Thus, using a refined grid around the free boundary can produce accurate solutions efficiently. Moreover, we see that the average iteration number is about 1.3 only to get four digits accuracy. This may be related to the accurate location of the free boundary at each time step. See Figure 5.12. Each point in the figure is the computed free boundary point found at each time step. The free boundary moves downward as τ moves to the right end. Comparing the computed boundary in these two graphs, we see that free boundary points are located more precisely when using the adaptive spatial grid than when using a uniform spatial grid. For example, for $m = 160$, the free boundary curve is both more accurate and much smoother for the adaptive grid than for the uniform grid.

However, after the fourth digit of accuracy is reached, we cannot get more accuracy as the grids are refined. Notice that the value computed with $m = 160$ decreases instead of increasing compared with the previous two values. That is, the values start to fluctuate. We believe this is because the condition number of the associated linear system is large. The machine epsilon for the computer we used is about 10^{-16} and the condition number is about 10^8. Thus, we expect about 8 digits accuracy in the solution of the linear systems. Other errors arise as well, limiting precision to about 7 digits accuracy. Thus, after 10^{-5}, the errors reduce very slowly. In more complicated cases, such as CBs, the situation is even worse. Moreover, locating the free boundary for a CB is more difficult, since the problem involves a "sawtooth" accrued interest function. We leave the development of better methods to track the free boundary associated with CBs to future research.

Table 5.21: Numerical results for the Crank-Nicolson method using the discrete penalty method for an American put option at $S = 100$ using uniform grid spacing and $tol = 1.0e^{-8}$.

Grid	Time-				No. of Iterations			Cond. Num.
Size	Steps	Value	Difference	Ratio	Min.	Max.	Avg.	(A)
40	40	14.54681348			1	2	1.100	1.82e+008
80	80	14.64461610	9.780e-002		1	2	1.113	2.32e+008
160	160	14.66999558	2.538e-002	3.85	1	3	1.106	3.04e+008
320	320	14.67656709	6.572e-003	3.86	1	3	1.109	4.07e+008
640	640	14.67826609	1.699e-003	3.87	1	3	1.111	5.54e+008
1280	1280	14.67871226	4.462e-004	3.81	1	4	1.113	7.63e+008

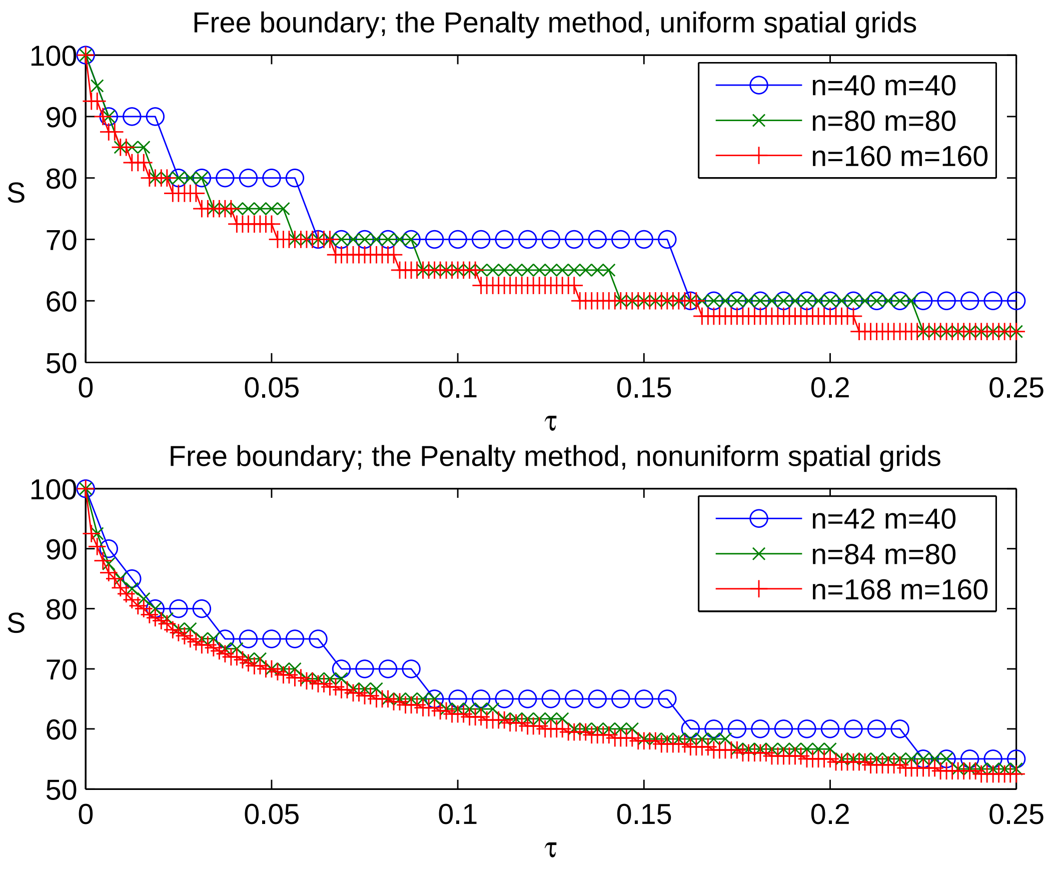


Figure 5.12: Comparison of the free boundary curves using uniform spatial grids and adaptive spatial grids with refinement around the free boundary.

Table 5.22: Numerical results for the Crank-Nicolson method using the discrete penalty method for an American put option at $S = 100$ using adaptive spatial grids and $tol = 1.0\mathrm{e}^{-8}$.

Grid	Time-				No. of Iterations			Cond. Num.
Size	Steps	Value	Difference	Ratio	Min.	Max.	Avg.	(A)
42	40	14.65240859			1	2	1.200	1.82e+008
84	80	14.67871967	2.631e-002		1	3	1.300	2.32e+008
168	160	14.67810374	-6.159e-004	-42.72	1	5	1.488	3.04e+008
336	320	14.67857919	4.755e-004	-1.30	1	12	1.906	4.07e+008
672	640	14.67875437	1.752e-004	2.71	1	22	2.720	5.54e+008
1344	1280	14.67883331	7.894e-005	2.22	1	39	4.399	7.63e+008

5.2.5 Adaptive Time-Stepping

Forsyth and Vetzal [FV02] extend the time-step selector suggested in [Joh87] to better handle American options. Given an initial time-step $\Delta\tau^{j+1}$, a new time-step is determined by the formula

$$\Delta\tau^{j+2} = \left(\min_i \left[\frac{dnorm}{\frac{|V(S_i,\tau^j+\Delta\tau^{j+1})|-V(S_i,\tau^j)}{\max(D,|V(S_i,\tau^j+\Delta\tau^{j+1})|,|V(S_i,\tau^j)|)}}\right]\right)\Delta\tau^{j+1}, \tag{5.19}$$

where *dnorm* is a target relative change (during the time-step), and D is a scalar. The value $D = 1$ is recommended in [FV02]. From (5.19), we can see that the new time-step is selected based on changes observed over the last time-step, with the goal that the change in the solution (relative to D and the size of the solution) should be about *dnorm*.

In [FV02], the authors explain that the error should be proportional to $dnorm^2$ for smooth problems when using the time-step selector (5.19). We select an initial stepsize for the coarsest grid, $(\Delta\tau)^0$, and then reduce the initial $(\Delta\tau)^0$ by a factor of four at each grid refinement. We used $(\Delta\tau)^0 = 10^{-3}$ and $dnorm = 0.2$ on the coarsest grid. The value of $dnorm$ was divided by two at each grid refinement. The numerical results are shown in Tables 5.23 and 5.24. From these two tables, it is clear that the method with the adaptive time-stepping has a smooth convergence ratio of about 4.0, as expected, and the value of "Difference" is slightly smaller than with constant time-stepping. Certainly, the results with adaptive time-stepping are not significantly better than those with constant time-stepping.

Table 5.23: Numerical results for the Crank-Nicolson method using a constant time-step and the discrete penalty method for an American put option at $S = 100$. Here, $tol = 1.0\text{e}^{-7}$.

Grid Size	Time-Steps				No. of Iterations		
(n)	(m)	Value	Difference	Ratio	Min.	Max.	Avg.
40	40	14.54681347			1	2	1.100
80	80	14.64461610	9.780e-002		1	2	1.113
160	160	14.66999558	2.538e-002	3.85	1	3	1.106
320	320	14.67656709	6.572e-003	3.86	1	3	1.109
640	640	14.67826609	1.699e-003	3.87	1	3	1.111
1280	1280	14.67871226	4.462e-004	3.81	1	4	1.113

Table 5.24: Numerical results for the Crank-Nicolson method using an adaptive time-step scheme and the discrete penalty method for an American put option at $S = 100$. Here, $tol = 1.0\text{e}^{-7}$.

Grid Size	Time-Steps				No. of Iterations		
(n)	(m)	Value	Difference	Ratio	Min.	Max.	Average
40	29	14.54842150			1	2	1.138
80	72	14.64527502	9.685e-002		1	2	1.125
160	159	14.67031183	2.504e-002	3.87	1	2	1.113
320	332	14.67671245	6.401e-003	3.91	1	2	1.111
640	677	14.67833366	1.621e-003	3.95	1	2	1.111
1280	1364	14.67874175	4.081e-004	3.97	1	2	1.111

5.3 Numerical Results for CBs

In this section, we present the final set of numerical results for the CB using the TF and the AFV models using the parameters in Tables 5.2 and 5.3, respectively.

5.3.1 TF Model

Table 5.25 lists the numerical results for a simple CB without coupon payments and without any call/put features. Table 5.26 is for a "semi-full" CB with coupon payments only. We can see from these two tables that the convergence rate is quadratic. We believe we achieve the expected rate of convergence in these two cases because the free boundary (due to convertibility only in these two cases) is not overly complicated. However, when call and put features are added, the convergence rate degrades to approximately linear. See Table 5.27. We believe this is mostly because the accrued interest is a "sawtooth" function, which causes both the lower free boundary condition and upper free boundary condition to vary with time. To deal effectively with these discontinuous free boundary conditions, we must apply the techniques discussed in the previous sections more carefully. For example, grid positioning requires adjusting the critical points with time, rather than using fixed critical points all the time, as we did in the examples given above. The nonsmooth free boundaries make the problems much more complicated. In this thesis, we do not apply grid positioning dynamically. Therefore, only linear convergence is achieved for the full CB problem. We leave the important question of how to handle discontinuous free boundary value problems to future work.

Table 5.25: Numerical results for a simple CB (without coupon payments and without any call/put features) at $S = 100$ using the TF model. We used $\lambda = 0.01$ and $tol = 1.0\text{e}^{-7}$.

Grid Size	Time-Steps	Value U	Difference	Ratio	CPU Time	No. of Iterations (Average)
100	100	104.3034176			3.281e-001	1.610
200	200	104.2905909	-1.283e-002		7.344e-001	1.755
400	400	104.2875199	-3.071e-003	4.177	1.969e+000	1.915
800	800	104.2867346	-7.853e-004	3.911	6.797e+000	2.000
1600	1600	104.2865405	-1.941e-004	4.046	2.533e+001	2.000
3200	3200	104.2864918	-4.879e-005	3.978	1.007e+002	2.000

Table 5.26: Numerical results for a semi-full CB (with coupon payments but without any call and put features) at $S = 100$ using the TF model. We used $\lambda = 0.01$ and $tol = 1.0\text{e}^{-7}$.

Grid Size	Time-Steps	Value U	Difference	Ratio	CPU Time	No. of Iterations (Average)
100	100	135.4694530			1.719e-001	1.100
200	200	135.4650858	-4.367e-003		4.375e-001	1.100
400	400	135.4639711	-1.115e-003	3.918	1.188e+000	1.100
800	800	135.4636854	-2.857e-004	3.901	5.547e+000	1.100
1600	1600	135.4636138	-7.168e-005	3.986	2.027e+001	1.100
3200	3200	135.4635957	-1.803e-005	3.975	7.984e+001	1.100

Table 5.27: Numerical results for a full CB (with coupon payments and with call and put features) at $S = 100$ using the TF model. We used $\lambda = 0.01$ and $tol = 1.0\text{e}^{-7}$.

Grid Size	Time-Steps	Value U	Difference	Ratio	CPU Time	No. of Iterations (Average)
100	100	124.2518066			1.563e-001	1.620
200	200	124.0998017	-1.520e-001		5.156e-001	1.725
400	400	124.0416281	-5.817e-002	2.613	1.719e+000	1.780
800	800	124.0075138	-3.411e-002	1.705	6.547e+000	1.855
1600	1600	123.9876613	-1.985e-002	1.718	2.631e+001	1.938
3200	3200	123.9779560	-9.705e-003	2.046	1.123e+002	2.069

5.3.2 AFV Model

As in the previous subsection, we test the simple CB, semi-full CB and full CB for the AFV model. Table 5.28 lists the numerical results for the simple CB, Table 5.29 for the semi-full CB, and Table 5.30 for the full CB. Similar to the TF model, the convergence rates for the simple and semi-full CBs using the AFV model are quadratic, while it is linear for the full CB. The reasons are similar to those discussed in the previous subsection. It is interesting to note that the value of the CB given by the AFV model is larger than that given by the TF model. We believe that this is because of differences in the two models and is not due to numerical errors. As discussed in Chapter 2, by setting $\eta = 0$ and $r_c = p(1 - R)$, we can expect the results to be similar, but not exactly the same. The experiments support this conclusion.

Table 5.28: Numerical results for a simple CB (without coupon payments and without any call/put features) at $S = 100$ using the AFV model. We used $\lambda = 0.01$ and $tol = 1.0\text{e}^{-7}$.

Grid Size	Time-Steps	Value U	Difference	Ratio	CPU Time	No. of Iterations (Average)
100	100	106.3702061			1.590	1.590
200	200	106.3555190	-1.469e-002		1.490	1.490
400	400	106.3519793	-3.540e-003	4.149	1.468	1.468
800	800	106.3510784	-9.009e-004	3.929	2.000	2.000
1600	1600	106.3508553	-2.231e-004	4.039	1.848	1.848
3200	3200	106.3507992	-5.605e-005	3.980	1.981	1.981

Table 5.29: Numerical results for a semi-full CB (with coupon payments but without any call/put features) at $S = 100$ using the AFV model. We used $\lambda = 0.01$ and $tol = 1.0\text{e}^{-7}$.

Grid Size	Time-Steps	Value U	Difference	Ratio	CPU Time	No. of Iterations (Average)
100	100	137.7893187			1.406e-001	1.100
200	200	137.7833560	-5.963e-003		3.906e-001	1.100
400	400	137.7818094	-1.547e-003	3.855	1.438e+000	1.100
800	800	137.7814204	-3.891e-004	3.975	5.688e+000	1.100
1600	1600	137.7813223	-9.810e-005	3.966	2.242e+001	1.100
3200	3200	137.7812977	-2.457e-005	3.992	8.650e+001	1.100

Table 5.30: Numerical results for a full CB (with coupon payments and with call/put features) at $S = 100$ using the AFV model. We used $\lambda = 0.01$ and $tol = 1.0\text{e}^{-7}$.

Grid Size	Time-Steps	Value U	Difference	Ratio	CPU Time	No. of Iterations (Average)
100	100	124.9570542			1.563e-001	1.620
200	200	124.9300469	-2.701e-002		6.250e-001	1.735
400	400	124.9224035	-7.643e-003	3.533	1.781e+000	1.817
800	800	124.9196067	-2.797e-003	2.733	7.516e+000	1.936
1600	1600	124.9185711	-1.036e-003	2.701	2.814e+001	2.097
3200	3200	124.9181387	-4.325e-004	2.394	1.267e+002	2.307

Chapter 6

Conclusions and Future Work

Convertible bonds with credit risks are difficult to price because they are hybrids of debt and equity instruments. Additional features, such as callability and puttability, make the problem even more complicated. Currently, most models in the literature adopt a single-factor Black-Scholes analysis (i.e., they use the stock price as the underlying stochastic variable with the interest rate assumed to be deterministic) to build the pricing frameworks for convertible bonds. Such models generally result in a complex coupled system of PDEs with free boundary conditions that are often solved as linear complementarity problems (LCPs). Two such models were studied in this thesis: the TF model and the AFV model.

These two models differ in their splitting strategies, which result in different PDEs as well as different initial conditions and boundary conditions. This has a significant effect on convergence when solving the associated parabolic PDEs numerically. In the AFV model, the initial conditions are either a hockey-stick function or a continuous function, while in the TF model, they are either a hockey-stick function or a step function. The numerical experiments show that the step function degrades convergence rates dramatically: with Crank-Nicolson time-stepping, the convergence rate for the TF model is approximately linear only, rather than quadratic, as we would expect for a smooth problem. Moreover, oscillations around the critical point (e.g., the kink point for the hockey-stick function or the jump point for the step function) appear in the numerical solutions. Also the experiments show that the degradation in the convergence rate is less for the hockey-stick function than for the step function (the convergence rate is close to quadratic for the American put option considered in our examples), and oscillation problems caused by the nonsmooth initial conditions in the numerical solution computed by the Crank-Nicolson method are even more pronounced for the associated Delta and Gamma functions than for the CB price itself. We do not see this degradation in the convergence rate or spurious oscillations when solving similar smooth problems with the Crank-Nicolson method.

This thesis investigates several techniques to alleviate convergence and oscillation problems cased by nonsmooth initial conditions. Rannacher smoothing, which takes one or two fully implicit steps at the start of the numerical integration and then uses Crank-Nicolson steps thereafter, can smooth out the oscillations in the numerical solutions effectively and restore the convergence rate back to quadratic when the initial condition is a hockey-stick function, as is the case for vanilla European and American call and put options, and convertible bonds under the AFV model. However, if the initial condition is a step function, as is the case for the COCB for convertible bonds under the TF model, Rannacher smoothing cannot restore convergence back to quadratic. Similarly, the technique of controlling $\lambda = \frac{\Delta\tau}{h}$ or $\Gamma = \frac{\Delta\tau}{h^2}$ can effectively eliminate oscillations caused by the nonsmooth initial conditions, but cannot effectively restore convergence back to quadratic for the discontinuous initial functions. Thus, we use grid positioning to remedy convergence problems in such cases. We position the grid so that the discontinuity is midway between two grid points. As we refine the grid, we do so in a way that ensures that the discontinuity remains midway between two grid points. With this grid positioning technique, the errors can be dramatically reduced and the convergence rate restored back to quadratic. The experimental results for convertible bonds under the

TF model demonstrate that this grid positioning approach is highly successful without sacrificing any efficiency.

The thesis also discusses the effectiveness of linear boundary conditions to approximate the real boundary conditions at the truncated end of the domain. We test the at-the-money price of derivatives using linear boundary conditions and compare the numerical results to those using Dirichlet boundary conditions or the analytical solution for a European option and an American option. The numerical results are similar in all cases, which shows that our linear boundary condition is an appropriate choice. Thus we can safely apply such a linear boundary condition to convertible bonds.

Far-field approximation is another issue associated with the truncated end of the domain. When we replace an infinite domain by a finite one, we need to choose the truncation point to be large enough to maintain good accuracy in the region of interest, but not so large as to make the computation too costly. We discuss the far-field selection criterion and test our strategy on American options and convertible bonds.

In the numerical solution of convection-diffusion problems, it is very desirable that the matrices associated with the numerical method are M-matrices. To guarantee that the numerical solution is stable and free from oscillation, we adjust the spatial step size or replace the central difference scheme with a one-sided difference scheme to ensure the associated matrix is a M-matrix. This greatly improves the numerical solutions of convection-dominated problems.

The thesis also investigates difficulties associated with the penalty method. Oscillations may appear in the penalty matrices when solving linear systems using the discrete penalty method, especially when the grid spacing and time-steps approach zero. In such cases, the number of iterations becomes very large and it is impossible to guarantee a stable and convergent solution. We recommend a continuous penalty scheme instead to solve this oscillation problem. We tested the continuous penalty scheme on American options, but haven't applied it yet to convertible bonds due to the complexity of convertible bonds. In particular, to apply the continuous penalty scheme to convertible bonds, we need to prove that the continuous penalty terms we construct can guarantee the approximation satisfies the lower and upper free-boundary conditions. This is a subject for future research.

Adaptive spatial grids and time-stepping methods are discussed with the goal of developing more effective numerical methods. Our schemes perform well on simple free boundary problem, such as American put options. However, for the more complicated convertible bonds, the difficulties of locating the free boundary points make it hard to apply similar adaptive spatial grids and time-stepping schemes to convertible bonds. Adaptive time-stepping can improve the convergence rate to some extent, but is not that effective. We did not obtain good results for convertible bonds. Further study on adaptive spatial grids and adaptive time-stepping is needed.

Bibliography

[AFV02] E. Ayache, P. A. Forsyth, and K. R. Vetzal. Next generation models for convertible bonds with credit risk. *Wilmott Magazine*, pages 68–77, December 2002.

[AFV03] E. Ayache, P. A. Forsyth, and K. R. Vetzal. The valuation of convertible bonds with credit risk. *Journal of Derivatives*, 11(1):9–29, fall, 2003.

[BS73] F. Black and M. Scholes. The pricing of options and corporate liabilites. *Journal of Political Economy*, 81:637–659, May/June 1973.

[BS77] M. J. Brennan and E. S. Schwartz. Convertible bonds: Valuation and optimal strategies for call and conversion. *Journal of Finance*, 32:1699–1715, 1977.

[BS80] M. J. Brennan and E. S. Schwartz. Analyzing convertible bonds. *Journal of Financial and Quantitative Analysis*, 15:907–929, 1980.

[CN94] W. Cheung and I. Nelken. Costing the converts. *Risk*, 7:47–49, 1994.

[Cry71] C.W. Cryer. The solution of a quadratic programming problem using systematic overrelaxation. *SIAMJ*, 9:385–392, 1971.

[F$^+$00] P. Farrell et al. *Robust Computational Techniques for Boundary Layers.* Chapman and Hall/CRC Bota Raton, 2000.

[FV02] P. A. Forsyth and K. R. Vetzal. Quadratic convergence for valuing American options using a penalty method. *SIAM Journal on Scientific Computing*, 23(6):2095–2122, 2002.

[HP96] T. S. Y. Ho and D. M. Pfeffer. Convertible bonds: Model, value attribution, and analysis. *Financial Analysis Journal*, 52:35–44, Sep./Oct. 1996.

[Hul06] J. Hull. *Options, Futures, and Other Derivatives.* Pearson/Prentice Hall, Upper Saddle River, N.J., 6th edition edition, 2006.

[Ing77] J. Ingersoll. A contingent-claims valuation of convertible securities. *Journal of Financial Economics*, 4:289–322, 1977.

[Joh87] C. Johnson. *Numerical Solution of Partial Differential Equations by the Finite ELement Method.* Cambridge University Press, Cambridge, UK, 1987.

[KN00] R. Kangro and R. Nicolaides. Far field boundary conditions for Black-Scholes equations. *SIAM Journal on Numerical Analysis*, 38(4):1357–1368, 2000.

[Lee03] C. C. W. Leentvaar. Numerical solution of the Black-Scholes equation with a small number of grid points. Master thesis, Delft University of Technology, 2003. http://ta.twi.tudelft.nl/users/oosterlee/oosterlee/verslag20.pdf.

[Li05] X. W. Li. Pricing convertible bonds using partial differential equations. M.Sc. thesis, Department of Computer Science, University of Toronto, 2005. http://www.cs.toronto.edu/pub/reports/na/lucy-05-msc.pdf.

[Mer73] R. C. Merton. Theory of rational option pricing. *Bell Journal of Economics and Management Science*, 4:141–183, Spring 1973.

[Mo06] Q. K. Mo. Pricing convertible bonds with dividend protection subject to credit risk using a numerical PDE approach. M.Sc. thesis, Department of Computer Science, Univeristy of Toronto, 2006. http://www.cs.toronto.edu/pub/reports/na/ccc/moqk-06-msc.pdf.

[MS86] J. J. McConnell and E. S. Shwartz. Lyon tanning. *Journal of Finance*, 41:561–576, 1986.

[NST02] B. F. Nielsen, O. Skavhaug, and A. Tveito. Penalty and front-fixing methods for the numerical solution of American option problems. *The Journal of Computational Finance*, 5:69–97, 2002.

[Ran84] R. Rannacher. Finite element solution of diffusion problems with irregular data. *Numberische Mathematik*, 43:309–327, 1984.

[Shr04] S. E. Shreve. *Stochastic Calculus for Finance II: Continuous-Time Models.* Springer-Verlag, New York, 2004.

[Str04] J. C. Strikwerda. *Finite Difference Schemes and Partial Differential Equations.* Society for Industrial and Applied Mathematics, Philadelphia, 2nd edition edition, 2004.

[TF98] K. Tsiveriotis and C. Fernandes. Valuing convertible bonds with credit risk. *The Journal of Fixed Income*, 8(2):95–102, 1998.

[TR00] D. Tavella and C. Randall. *Princing Financial Instruments: The Finite Difference Method.* John Wiley & Sons, 2000.

[WFV04] H. Windcliff, P. A. Forsyth, and K. R. Vetzal. Analysis of the stability of the linear boundary condition for the Black-Scholes equation. *The Journal of Computational Finance*, 8:65–92, Fall, 2004.

[WHD95] P. Wilmott, S. Howison, and J. Dewynne. *The Mathematics of Financial Derivatives : A Student Introduction.* Cambridge University Press, Cambridge, 1995.

[You71] D. M. Young. *Iterative Solution of Large Linear System.* Academic Press, New York, 1971.