Pricing American Options Using LU Decomposition

Samuli Ikonen* Jari Toivanen*

November 12, 2005

Abstract

Numerical solution methods for pricing American options are considered. We propose a second-order accurate Runge-Kutta scheme for the time discretization of the Black-Scholes partial differential equation with an early exercise constraint. We reformulate the algorithm introduced by Brennan and Schwartz into a simple form using a LU decomposition and a modified backward substitution with a projection. In addition, we describe a direct solution method given by Elliott and Ockendon and we consider the similarity of these two direct algorithms. Numerical experiments demonstrate that the Runge-Kutta scheme produces smaller errors and less oscillations to numerical solutions than the Crank-Nicolson method. Experiments also show that the Brennan and Schwartz algorithm is much faster than the projected SOR method.

Keywords: American option, time discretization, linear complementarity problem, Brennan and Schwartz algorithm, direct method, LU decomposition.

1 Introduction

Financial options are widely used in a field of finance [17]. A valuation of option contracts has been topic of research in the last decades and there exist various type of mathematical models for the prices of different kinds of options. For the many of these models there are no analytical solution formulas available and hence, numerical valuation techniques need to be employed. One of the most famous model for the price of an American option is based on the Black-Scholes partial differential equation [1]. In general, a numerical solution of this option pricing model consists two tasks which are the discretization of the underlying partial differential equation and the solution of complementarity problems.

Surveys of valuation techniques for options are presented for example in [3], [11]. In [3], the speed of computing and the accuracy of computed option prices are compared among many existing valuation methods of American options. For instance, a binomial method, an accelerated binomial method, a trinomial method and an integral equation

*Department of Mathematical Information Technology, PO BOX 35 (Agora), 40014 University of Jyväskylä, Finland, Samuli.Ikonen@mit.jyu.fi, Jari.Toivanen@mit.jyu.fi

method are briefly explained and compared in that paper. Explicit and implicit finite difference methods and the binomial method are considered in [11]. The methods are compared numerically in the cases of American call and put options with and without dividends. Furthermore, several option models and solution techniques are studied in books [17], [28], [29], [31], and references therein.

There are classical methods for the space discretization of partial differential equations and those have been applied in the field of the option pricing. For example, a finite difference method is used in [2], [15], [16], [24], a finite element method is considered in [28], and a finite volume method in [10].

The Black-Scholes partial differential equation contains variable coefficients for the first-order and second-order spatial derivatives. The space discretization of this type of partial differential equation may lead to numerical difficulties. It turns out that the Black-Scholes partial differential equation can be transformed into a constant coefficient diffusion equation and after that the space discretization becomes easier; see [28], [31]. However, there exist lot of option pricing models where above mentioned transformation can not be done and therefore, the discretization of the convection-diffusion type Black-Scholes partial differential equation should be done. Hence, the choice of the space discretization scheme for the option pricing problem depends on the form of the partial differential equation which is in the case. We prefer a basic finite difference scheme because of its simplicity and furthermore, because it leads to a tridiagonal space discretization matrix which enables the efficient use of the LU decomposition.

The Black-Scholes partial differential equation is time dependent and that is why the numerical time integration needs to be considered. First-order accurate explicit and implicit Euler schemes are well known methods for the time discretization [30], [31]. Although, the implicit scheme is only first-order accurate, it has good stability properties. The Crank-Nicolson method is second-order accurate in time and it is widely used in financial problems [29]. Even though the scheme is unconditionally stable it can produce undesired oscillations to numerical solutions [10], [21], [26]. One way to improve the stability is that the time-stepping is started with a few implicit Euler time steps and then continued with the Crank-Nicolson method. This type of scheme was introduced by Rannacher and it damps undesired oscillations more efficiently than the Crank-Nicolson method does [27]. This scheme is applied for the option pricing problems in [26]. They apply the scheme for the digital call option problem and they showed that the Rannacher time-stepping leads to stable convergence rates while the Crank-Nicolson method does not. The numerical examples were performed with the original nonsmooth payoff function and with a specially smoothed payoff functions.

The second-order backward difference formula and the Runge-Kutta scheme, which are considered in this paper are L-stable second-order accurate time discretizations [4], [25]. In numerical experiments we show that these both schemes lead to more accurate numerical solution than the Crank-Nicolson method. In addition, convergence rates and oscillation problems are studied.

The pricing of an American option is a variational inequality problem [12], and moreover it can be formulated as a complementarity problem [15], [29]. A basic reference related to the complementarity problems is [6]. There are direct and iterative solution algorithms for solving the complementarity problems numerically. For example, the widely used projected SOR method is applied for option pricing in [31], a penalty method in [10], and a front-fixing method in [24]. Direct algorithms are applied in [2], [22] and an operator splitting method is used in [18]. The option pricing problem can be also formulated to the form of linear programming. Direct methods and the linear programming are considered in [7], [8]. In this paper, we study formerly known direct methods and furthermore, also the projected SOR method is applied in the numerical experiments.

The rest of this paper is divided into five sections. First, in Section 2, models for the American call and put options are briefly introduced. After that, the space and time discretizations of the Black-Scholes partial differential equation are considered. Two direct algorithms for the complementarity problem arising from the pricing of the American option are the subject of Section 4. The algorithm by Brennan and Schwartz was introduced in [2] and the algorithm by Elliott and Ockendon [9] is applied for the option pricing in [22]. In that section, we reformulate these algorithms into a simple form using the LU decomposition. Moreover, we consider the similarity of these algorithms. The numerical experiments are given in Section 5 and finally, Section 6 contains conclusions.

2 Models for American Options

We start by introducing mathematical models for American call and put options. The models considered in this paper are based on the Black-Scholes partial differential equation

$$\frac{\partial \phi}{\partial t} + \frac{1}{2}\sigma^2 x^2 \frac{\partial^2 \phi}{\partial x^2} + (r-d)x\frac{\partial \phi}{\partial x} - r\phi = 0, \tag{1}$$

where the function $\phi(x,t)$ is the price of the option for a given underlying asset value x at a given time t. Due to the early exercise possibility of American options an additional constraint

$$\phi(x,t) \geq g(x) \tag{2}$$

has to be introduced in order to avoid arbitrage possibilities. Here, g is the payoff function of the option contract. The price of the option is obtained by solving the partial differential equation with the previous constraint, boundary conditions and a final condition [29], [31].

It is well known [31] that there is a value $S_f(t)$ for all t which divides the domain $(0,\infty)$ into two subdomains $(0,S_f(t))$ and $(S_f(t),\infty)$ in such a way that in one of these subdomains the price of the option equals to the payoff function while in the other region it is higher than the payoff. The price of the option satisfies the Black-Scholes equation (1) in the subdomain where it is higher than the payoff. The function $S_f(t)$ is not known beforehand and it has to be found together with the price of the option. Hence, the option pricing problem is a free boundary problem. The stock value $S_f(t)$ indicates when the option should be exercised.

Option pricing problems considered in this paper are posed in an infinite region $[0,\infty) \times [0,T]$ with Dirichlet boundary conditions and a final condition. In order to solve these problems numerically, we reformulate problems in a truncated region $[0,X] \times [0,T]$. The truncation point X has to be sufficiently far in order to avoid excessive error due to the truncation. On the other hand unnecessarily large value of X increases computational cost. The choice of X is considered in [20], for example. Furthermore, we transform final value problems to a more familiar form of initial value problems.

2.1 American call option

An American call option gives a right to buy the underlying asset for the exercise price E. The value of this option is obtained by solving the linear complementarity problem

$$\begin{cases} \dfrac{\partial \phi}{\partial t} - \dfrac{1}{2}\sigma^2 x^2 \dfrac{\partial^2 \phi}{\partial x^2} - (r-d)x\dfrac{\partial \phi}{\partial x} + r\phi \geq 0, \\ \phi(x,t) \geq g(x), \\ \left(\dfrac{\partial \phi}{\partial t} - \dfrac{1}{2}\sigma^2 x^2 \dfrac{\partial^2 \phi}{\partial x^2} - (r-d)x\dfrac{\partial \phi}{\partial x} + r\phi\right)(\phi - g) = 0, \end{cases} \tag{3}$$

where $(x,t) \in [0,X] \times [0,T]$, with the boundary conditions

$$\phi(0,t) = 0 \qquad \text{and} \qquad \phi(X,t) = X - E, \tag{4}$$

and with the initial value $\phi(x,0) = g(x)$. Here, g is the payoff function

$$g(x) = \max(x - E, 0). \tag{5}$$

We denote the risk free interest rate by r, the volatility by σ and the constant dividend yield by d. In the domain $(0, S_f(t))$ at time t, the solution satisfies the Black-Scholes partial differential equation and the solution is equal to the payoff function in the domain $(S_f(t), X)$.

2.2 American put option

An American put option gives a right to sell the underlying asset for the exercise price E. The initial value problem related to the American put option is

$$\begin{cases} \dfrac{\partial \phi}{\partial t} - \dfrac{1}{2}\sigma^2 x^2 \dfrac{\partial^2 \phi}{\partial x^2} - (r-d)x\dfrac{\partial \phi}{\partial x} + r\phi \geq 0, \\ \phi(x,t) \geq g(x), \\ \left(\dfrac{\partial \phi}{\partial t} - \dfrac{1}{2}\sigma^2 x^2 \dfrac{\partial^2 \phi}{\partial x^2} - (r-d)x\dfrac{\partial \phi}{\partial x} + r\phi\right)(\phi - g) = 0, \end{cases} \tag{6}$$

where $(x,t) \in [0,X] \times [0,T]$, with the boundary conditions

$$\phi(0,t) = E \qquad \text{and} \qquad \phi(X,t) = 0, \tag{7}$$

and with the initial value $\phi(x,0) = g(x)$. The payoff function for the put option contract is

$$g(x) = \max(E - x, 0). \tag{8}$$

The solution ϕ is equal to the payoff function at time t in the domain $(0, S_f(t))$, while in the domain $(S_f(t), X)$, it satisfies the Black-Scholes partial differential equation.

3 Discretization of the Black-Scholes equation

In the following, we consider the discretization of the Black-Scholes partial differential equation

$$\frac{\partial \phi}{\partial t} - \frac{1}{2}\sigma^2 x^2 \frac{\partial^2 \phi}{\partial x^2} - (r-d)x\frac{\partial \phi}{\partial x} + r\phi = 0. \tag{9}$$

We apply a uniform grid for the computational domain $[0, X] \times [0, T]$ which is formed with a space step $\Delta x = X/(m+1)$ and a time step $\Delta t = T/n$. Moreover, we use the notation

$$\phi_i^{(j)} \approx \phi(x_i, t_j) = \phi(i\Delta x, j\Delta t), \tag{10}$$

where $i = 0, \ldots, m+1$ and $j = 0, \ldots, n$, for the numerical approximation of the solution. The efficiency of numerical solution can be improved by coordinate transformations or using nonuniform grids [5], [29]. For simplicity we do not consider such approaches here, but the proposed methods can be readily extended for these techniques.

3.1 Space discretization

The second-order accurate central finite differences

$$\frac{\partial \phi}{\partial x}(x_i, t) \approx \frac{\phi_{i+1}(t) - \phi_{i-1}(t)}{2\Delta x} \quad \text{and} \quad \frac{\partial^2 \phi}{\partial x^2}(x_i, t) \approx \frac{\phi_{i+1}(t) - 2\phi_i(t) + \phi_{i-1}(t)}{(\Delta x^2)} \tag{11}$$

are used for the approximation of the space derivatives of the Black-Scholes partial differential equation [23], [29]. These lead to the semi-discrete equation

$$\frac{\partial \phi}{\partial t} - \frac{1}{2}\left(\sigma^2 i^2 - (r-d)i\right)\phi_{i-1} + \left(\sigma^2 i^2 + r\right)\phi_i - \frac{1}{2}\left(\sigma^2 i^2 + (r-d)i\right)\phi_{i+1} = 0, \tag{12}$$

where $i = 1, \ldots, m$. This equation can be expressed more compactly in the form

$$\frac{\partial \boldsymbol{\phi}}{\partial t} + \boldsymbol{B}\boldsymbol{\phi} = 0, \tag{13}$$

where the matrix $\boldsymbol{B}$ is tridiagonal. With this discretization $\boldsymbol{B}$ has an M-matrix property if $\sigma^2 > r - d$. This property guarantees that the space discretization does not cause undesired oscillations into the numerical solution [32], [33].

3.2 Time discretization

Next, the time discretization of the semi-discrete equation (13) is discussed. We consider the Crank-Nicolson scheme, the second-order backward difference formula (BDF2) and a Runge-Kutta scheme. In the option pricing problems, the stability of the time discretization scheme is an important property because of the nonsmooth initial data. The stability of time discretization schemes is considered in [4], [21]. It is demonstrated that in certain type of problems the Crank-Nicolson method performs poorly while L-stable methods are more accurate.

The Crank-Nicolson time discretization scheme can be interpreted to be the average of the implicit and explicit Euler schemes. The scheme

$$\frac{1}{\Delta t}\left(\boldsymbol{\phi}^{(j+1)} - \boldsymbol{\phi}^{(j)}\right) + \frac{1}{2}\boldsymbol{B}\left(\boldsymbol{\phi}^{(j+1)} + \boldsymbol{\phi}^{(j)}\right) = 0, \tag{14}$$

is second-order accurate and unconditionally stable [29]. Despite this stability property, it has been noticed, for example in [14], [21], [26], that certain types of initial value can lead to numerical solutions with oscillations. One possible way to prevent the oscillation problem is to start the time-stepping by a finite number of implicit steps and then continue it with the Crank-Nicolson scheme [26], [27].

The BDF2 formula is more stable than the Crank-Nicolson scheme [25]. The scheme is L-stable [14], [25], and due to this high frequency oscillations are correctly damped out in time. This second-order accurate scheme is of the form

$$\frac{1}{\Delta t}\left(-\frac{4}{3}\boldsymbol{\phi}^{(j)} + \boldsymbol{\phi}^{(j+1)} + \frac{1}{3}\boldsymbol{\phi}^{(j-1)}\right) + \frac{2}{3}\boldsymbol{B}\boldsymbol{\phi}^{(j+1)} = 0, \tag{15}$$

where $\boldsymbol{\phi}^{(j+1)}$ is computed by using data from time steps j and $j-1$. The numerical integration has to be started with some other discretization method because this scheme needs data from two previous time steps. For example, we compute the vector $\boldsymbol{\phi}^{(1)}$ using the implicit Euler scheme.

The third time discretization which we applied in numerical experiments is an L-stable Runge-Kutta scheme [4]. The scheme is two-step method and it can be written in the form

$$\left\{ \begin{aligned} (\boldsymbol{I} + \theta\Delta t\boldsymbol{B})\overline{\boldsymbol{\phi}}^{(j+1)} &= (\boldsymbol{I} - (1-\theta)\Delta t\boldsymbol{B})\boldsymbol{\phi}^{(j)}, \\ (\boldsymbol{I} + \theta\Delta t\boldsymbol{B})\boldsymbol{\phi}^{(j+1)} &= (\boldsymbol{I} - \tfrac{1}{2}\Delta t\boldsymbol{B})\boldsymbol{\phi}^{(j)} - (\tfrac{1}{2} - \theta)\Delta t\boldsymbol{B}\overline{\boldsymbol{\phi}}^{(j+1)}, \end{aligned} \right. \tag{16}$$

where the parameter is chosen to be

$$\theta = 1 - 1/\sqrt{2}. \tag{17}$$

This scheme contains two fractional steps and hence, two systems of linear equations need to be solved at each time step while the Crank-Nicolson method and the BDF2 formula requires only one solution. In addition, it is worth noticing that this is a special case of a three-stage operator-splitting scheme in [13] when θ is chosen as in (17) and the second operator in the splitting is chosen as the zero operator.

The space and time discretizations of the partial differential equation lead to the system of linear equations which needs be solved at each time step. For example, in the case of the BDF2 formula the system is of the form

$$\left(\frac{1}{\Delta t}\boldsymbol{I} + \frac{2}{3}\boldsymbol{B}\right)\boldsymbol{\phi}^{(j+1)} = \frac{4}{3\Delta t}\boldsymbol{I}\boldsymbol{\phi}^{(j)} - \frac{1}{3\Delta t}\boldsymbol{I}\boldsymbol{\phi}^{(j-1)}, \tag{18}$$

where the matrix $\boldsymbol{B}$ arises from the space discretization. Furthermore, by using notations for the discretization matrix and for the right hand side vector, this equation can be written in the form

$$\boldsymbol{A}\boldsymbol{\phi}^{(j+1)} = \boldsymbol{b}^{(j)}, \tag{19}$$

where $j = 1, \ldots, n-1$. In the next section, the elements of the tridiagonal discretization matrix are indexed in the following way

$$\boldsymbol{A} = \begin{pmatrix} a_{11} & a_{12} & & & \\ a_{21} & a_{22} & a_{32} & & \\ & & \ddots & \ddots \ddots & \\ & & & & a_{m-1m} \\ & & & a_{mm-1} & a_{mm} \end{pmatrix}. \tag{20}$$

4 Direct algorithms for the complementarity problem

In this section, we consider two direct algorithms for the complementarity problems arising from the pricing of American options. The aim of this section is to show that the Brennan and Schwartz algorithm can be reformulate into a simple form using a LU decomposition. In this method, the early exercise constraint of the American option contract is taken into account in a modified backward substitution and, hence, the solution algorithm differs only slightly from the solution method of a system of linear equations with a tridiagonal matrix. In addition, we show that the algorithm presented by Elliott and Ockendon can be also reformulated into that same simple form.

In the following, we restrict to study the solution of the American call option problem. The pricing of put options by using the reformulated algorithm is straightforward because only the numbering of the nodes should be reversed. Moreover, in order to compare the Brennan and Schwartz algorithm with the algorithm given by Elliott and Ockendon, we transform the linear complementarity problem into a standard for where the solution is required to be non negative.

4.1 Brennan and Schwartz algorithm with a LU decomposition

Brennan and Schwartz introduced the algorithm for the American put option problem in [2]. The solution algorithm is based on a Gaussian elimination where the early exercise constraint of the option contract is handled in a simple manner. This algorithm is analysed in [19]. In this section, we reformulate this algorithm in such a way that it can be use for the American call option problem (3) and, furthermore, we give the algorithm in a form where a LU decomposition of the discretization matrix can be used.

The space and time discretizations of the complementarity problems (3) and (6) lead to a sequence of stationary complementarity problems. Hence, at each time step j, $j = 0, \ldots, n-1$, the problem

$$\begin{cases} \boldsymbol{A}\boldsymbol{\phi}^{(j+1)} \geq \boldsymbol{b}^{(j)}, \\ \boldsymbol{\phi}^{(j+1)} \geq \boldsymbol{g}, \\ \left(\boldsymbol{A}\boldsymbol{\phi}^{(j+1)} - \boldsymbol{b}^{(j)}\right)^T \left(\boldsymbol{\phi}^{(j+1)} - \boldsymbol{g}\right) = 0, \end{cases} \tag{21}$$

needs to be solved. Here, the matrix $\boldsymbol{A}$ arises from the discretization of the partial differential equation and the vector $\boldsymbol{g}$ refers the early exercise constraint. In the following, we consider how the Brennan and Schwartz algorithm is applied at every time step to solve the problem (21). As we observe, once a LU decomposition has been made, it suffices only to make a forward and a modified backward substitution at each time step.

First, we describe the Brennan and Schwartz algorithm briefly for the American call option. The algorithm is the following: The Gaussian elimination transforms each row of the system of linear equations $\boldsymbol{A}\boldsymbol{\phi} = \boldsymbol{b}$ to be the form

$$p_i \phi_i + s_i \phi_{i+1} = k_i, \tag{22}$$

for $i = 1, \ldots, m$, where the coefficients can be chosen to be

$$p_1 = a_{11}, \quad s_1 = a_{12}, \quad k_1 = b_1,$$
$$p_i = a_{ii} - \frac{a_{ii-1}}{p_{i-1}}\, a_{i-1i}, \quad s_i = a_{ii+1}, \quad k_i = b_i - \frac{a_{ii-1}}{p_{i-1}}\, k_{i-1},$$
$$p_m = a_{mm} - \frac{a_{mm-1}}{p_{m-1}}\, a_{m-1m} \quad \text{and} \quad k_m = b_m - \frac{p_{m-1}}{a_{mm-1}}\, k_{m-1}.$$

These coefficients are formed starting from the first row of the system of linear equations while in the original algorithm the elimination was started from the last row. The reason is that we consider call options and the article [2] considers put options. The solution $\boldsymbol{\phi}$ for the problem (21) is obtained by using the equations (22) and the payoff function of the call option. Start by solving ϕ_m from the equation (22) when $i = m$; if ϕ_m is less than the payoff value then set $\phi_m = g_m$. Then proceed to solve ϕ_{m-1}, etc. For details, see the original article [2].

The Brennan and Schwartz algorithm can be reformulated to the form where the LU decomposition is applied. In a basic form, the solution of the system of linear equations using the LU decomposition consists of forward and backward substitutions:

$$\boldsymbol{L}\boldsymbol{y} = \boldsymbol{b} \quad \text{and} \quad \boldsymbol{U}\boldsymbol{x} = \boldsymbol{y}, \tag{23}$$

where $\boldsymbol{L}$ is a lower triangular matrix and $\boldsymbol{U}$ is an upper triangular matrix. Next, we give the Brennan and Schwartz algorithm in a form where only the previous backward substitution is modified.

In the following LU decomposition, the lower and the upper triangular matrices are formed in such a way that elements of $\boldsymbol{U}$ corresponds to p_i and s_i in the Brennan and Schwartz algorithm. The similarity can be seen by noticing:

$$l_{ii-1} = \frac{a_{ii-1}}{p_{i-1}}, \quad l_{ii} = 1, \quad u_{ii} = p_i, \quad u_{ii+1} = s_i, \quad \text{and} \quad y_i = k_i. \tag{24}$$

Reformulation for the Brennan and Schwartz algorithm using the LU decomposition of the discretization matrix reads

Algorithm 1: Brennan and Schwartz algorithm with the LU decomposition

$u_{11} = a_{11}$
$u_{12} = a_{12}$
do $i = 2, \ldots, m-1$
    $l_{ii-1} = a_{ii-1}/u_{i-1i-1}$
    $u_{ii} = a_{ii} - l_{ii-1}\ u_{i-1i}$
    $u_{ii+1} = a_{ii+1}$
end do
$l_{mm-1} = a_{mm-1}/u_{m-1m-1}$
$u_{mm} = a_{mm} - l_{mm-1}\ u_{m-1m}$

$y_1 = b_1$
do $i = 2, \ldots, m$
    $y_i = b_i - l_{ii-1}\ y_{i-1}$
end do

$\phi_m = y_m/u_{mm}$
$\phi_m = \max(\phi_m, g_m)$
do $i = m-1, \ldots, 1$
    $\phi_i = (y_i - u_{ii+1}\ \phi_{i+1})/u_{ii}$
    $\phi_i = \max(\phi_i, g_i)$
end do

It is obvious that the use of the max-function in the backward substitution is possible because of the form of the solution of the option pricing problem. This algorithm is for the time step j of the problem (21) and so, the right hand side vector $\boldsymbol{b}^{(j)}$ should be updated at each time step.

4.2 Brennan and Schwartz algorithm for a transformed problem

In order to compare the Brennan and Schwartz algorithm with another direct algorithm, we transform the call option problem to the form where the early exercise constraint equals to zero function instead of the payoff function. This transformation is made by using the following substitutions for the problem (21):

$$\boldsymbol{z}^{(j+1)} = \boldsymbol{\phi}^{(j+1)} - \boldsymbol{g} \quad \text{and} \quad \boldsymbol{v}^{(j)} = \boldsymbol{b}^{(j)} - \boldsymbol{A}\boldsymbol{g}. \tag{25}$$

With these notations, the complementarity problems can be reformulated into a form:

$$\begin{cases} \boldsymbol{A}\boldsymbol{z}^{(j+1)} \geq \boldsymbol{v}^{(j)}, \\ \boldsymbol{z}^{(j+1)} \geq 0, \\ \left(\boldsymbol{A}\boldsymbol{z}^{(j+1)} - \boldsymbol{v}^{(j)}\right)^T \boldsymbol{z}^{(j+1)} = 0, \end{cases} \tag{26}$$

where $j = 0, \ldots, n-1$.

In the following, we formulate the Brennan and Schwartz algorithm using another LU decomposition which has ones on the diagonal of $\boldsymbol{U}$. Again a projection is added into the backward substitution. The Brennan and Schwartz algorithm for the transformed problem (26) with the alternative LU decomposition reads

Algorithm 2: Brennan and Schwartz algorithm for the transformed problem

$\bar{l}_{11} = a_{11}$
$\bar{u}_{12} = a_{12}/\bar{l}_{11}$
do $i = 2, \ldots, m-1$
    $\bar{l}_{ii-1} = a_{ii-1}$
    $\bar{l}_{ii} = a_{ii} - \bar{l}_{ii-1}\ \bar{u}_{i-1i}$
    $\bar{u}_{ii+1} = a_{ii+1}/\bar{l}_{ii}$
end do
$\bar{l}_{mm-1} = a_{mm-1}$
$\bar{l}_{mm} = a_{mm} - \bar{l}_{mm-1}\ \bar{u}_{m-1m}$

$\bar{y}_1 = v_1/\bar{l}_{11}$
do $i = 2, \ldots, m$
    $\bar{y}_i = (v_i - \bar{l}_{ii-1}\ \bar{y}_{i-1})/\bar{l}_{ii}$
end do

$z_m = \bar{y}_m$
$z_m = \max(z_m, 0)$
do $i = m-1, \ldots, 1$
    $z_i = \bar{y}_i - \bar{u}_{ii+1}\ z_{i+1}$
    $z_i = \max(z_i, 0)$
end do

The solution for the original problem (21) is obtained by $\boldsymbol{\phi}^{(j+1)} = \boldsymbol{z}^{(j+1)} + \boldsymbol{g}$.

4.3 Elliott-Ockendon algorithm with the LU decomposition

The second algorithm which we consider is presented by Elliott and Ockendon. They give the algorithm for the linear complementarity problem in [9] and this direct algorithm is applied for the American option pricing problems in [22]. According to [15], for example, the complementarity problems can be written in the form:

$$\boldsymbol{w}^{(j+1)} = \boldsymbol{A}\boldsymbol{z}^{(j+1)} - \boldsymbol{v}^{(j)}, \quad \boldsymbol{w}^{(j+1)} \geq 0, \quad \boldsymbol{z}^{(j+1)} \geq 0 \;\text{ and }\; (\boldsymbol{w}^{(j+1)})^T \boldsymbol{z}^{(j+1)} = 0, \qquad (27)$$

where $j = 0, \ldots n-1$. This form of the complementarity problem is also considered in [9] and [22]. They assume that the discretization matrix $\boldsymbol{A}$ is an M-matrix. Furthermore, it is assumed that the vector $\boldsymbol{v}^{(j)}$ has an specified form; see details in [9]. In the following, the solution method is again considered only at one time step, and so the time step index j is omitted.

The Elliott-Ockendon algorithm in the reference [9] reads

Algorithm 3: Elliott-Ockendon algorithm

decompose $\quad i = 1, \quad \alpha_1 = a_{11}, \quad f_1 = v_1/a_{11}$

step: $\quad i \to i + 1$

$\gamma_{i-1} = a_{i-1i}/\alpha_{i-1}$

$\alpha_i = a_{ii} - a_{ii-1}\ \gamma_{i-1}$

$f_i = \big(v_i - a_{ii-1}\ f_{i-1}\big)/\alpha_i$

if $\quad f_i > v_{i+1}/a_{i+1i}$ **then go to step**

backsolve $\quad k = i$

$z_k = f_k$

$i = k - 1\ (-1)\ 1$

$z_i = f_i - \gamma_i\ z_{i+1}$

Some parts of this algorithm are similar to the LU decomposition. That is, γ_i and α_i are elements of the LU decomposition and f_i is the intermediate vector after the forward substitution. In the Elliott-Ockendon algorithm, only those elements are computed which are needed.

Next, we reformulate this Elliott-Ockendon algorithm in such a way that the LU decomposition is formed fully and the early exercise constraint is taken into account in a modified backward substitution. The elements of this LU decomposition correspond to those of the Elliott-Ockendon algorithm in the following way:

$$\bar{l}_{ii} = \alpha_i, \quad \bar{u}_{ii+1} = \gamma_i, \quad \text{and} \quad \bar{y}_i = f_i. \qquad (28)$$

Taking advantage of the LU decomposition, the Elliott-Ockendon algorithm can be written in the form:

Algorithm 4: Elliott-Ockendon algorithm with the LU decomposition

$\bar{l}_{11} = a_{11}$
$\bar{u}_{12} = a_{12}/\bar{l}_{11}$
do $i = 2, \ldots, m-1$
    $\bar{l}_{ii-1} = a_{ii-1}$
    $\bar{l}_{ii} = a_{ii} - \bar{l}_{ii-1}\ \bar{u}_{i-1i}$
    $\bar{u}_{ii+1} = a_{ii+1}/\bar{l}_{ii}$
end do
$\bar{l}_{mm-1} = a_{mm-1}$
$\bar{l}_{mm} = a_{mm} - \bar{l}_{mm-1}\ \bar{u}_{m-1m}$

$\bar{y}_1 = v_1/\bar{l}_{11}$
$i = 1$
do while$\big(\bar{y}_i > v_{i+1}/a_{i+1i}\big)$
    $i = i + 1$
    $\bar{y}_i = \big(v_i - a_{ii-1}\ \bar{y}_{i-1}\big)/\bar{l}_{ii}$
end do

$k = i$
do $i = m, \ldots, k+1$
    $z_i = 0$
end do

$z_k = \bar{y}_k$
do $i = k-1, \ldots, 1$
    $z_i = \bar{y}_i - \bar{u}_{ii+1}\ z_{i+1}$
end do

Again, this algorithm should be applied at every time step and the solution for the original option pricing problem is obtained by by $\boldsymbol{\phi}^{(j+1)} = \boldsymbol{z}^{(j+1)} + \boldsymbol{g}$.

4.4 Comparison of two direct algorithms

At each time step the solution of the linear complementarity problem satisfies the Black-Scholes partial differential equation in one region and in the other region it equals the early exercise constraint. Thus, at the time step j there is one discretization node k_j which divides the x-axis for these two separate regions. Next, we study how this node k_j can be found during the forward substitution. We show that the solution satisfies the partial differential equation at the nodes where the intermediate value after the forward substitution is positive and the solution equals to the early exercise constraint (here zero function) at the nodes where the intermediate value after the forward substitution is negative. This kind of condition can be found by studying both direct algorithms and this is what is done in the following.

As mentioned earlier, we assume that the solution has a specified form. More precisely, it is assumed that the solution $\boldsymbol{z}^{(j)}$ has the form

$$z_i^{(j)} > 0,\ i = 0, \ldots, k_j, \quad \text{and} \quad z_i^{(j)} = 0,\ i = k_j + 1, \ldots, m + 1, \tag{29}$$

at each time step j.

First, we study the Elliott-Ockendon algorithm and especially Algorithm 4. The condition in the **do while** loop can be reformulate using the following equivalence relations:

$$\begin{aligned} \bar{y}_{i+1} = (v_{i+1} - a_{i+1i}\ \bar{y}_i)/\bar{l}_{ii} > 0 \\ \iff \quad v_{i+1} - a_{i+1i}\ \bar{y}_i > 0 \\ \iff \quad v_{i+1} > a_{i+1i}\ \bar{y}_i \\ \iff \quad \bar{y}_i > v_{i+1}/a_{i+1i}, \end{aligned}$$

where a_{i+1i} is negative due to the M-matrix property. Also the positivity of $\bar{l}_{ii}$ follows easily from this property. We assume that k_j is the smallest index for which it holds that $\bar{y}_{k_j+1} < 0$. According to the Elliott-Ockendon algorithm and equivalences above, we notice that $z_{i+1} = 0$ for $i = k_j, \ldots, m$ and furthermore, the rest of the solution can be computed using the backward substitution which is $z_i = \bar{y}_i - \bar{u}_{ii+1}\ z_{i+1}$ for $i = k_j, \ldots, 1$.

This leads to the conclusion that also the Elliott-Ockendon algorithm can be reformulated into the form where the forward substitution is carried out for all components and the early exercise constraint is taken into account in the modified backward substitution. In fact, this modified backward substitution is similar to the last loop of Algorithm 2.

Similarly, in the Brennan and Schwartz algorithm the node k_j can be found by studying the forward substitution. First, the last loop of Algorithm 2 can be written in the form:

```
do while (ȳ_i − ū_{ii+1} z_{i+1} < 0)
    z_i = 0
    i = i − 1
end do

k = i
do i = k, 1
    z_i = ȳ_i − ū_{ii+1} z_{i+1}
end do
```

From this and the assumption on the specific form of the solution, it follows that the solution z_i is zero when $\bar{y}_i$ is negative. Hence, also in this Brennan and Schwartz algorithm we can find the node k_j by studying the sing of $\bar{y}_i$ in the forward substitution.

5 Numerical Experiments

In this section, we present numerical experiments with the methods discussed in the previous sections. We show that in the option pricing the BDF2 formula and the Runge-Kutta scheme are more efficient time discretizations than the widely used Crank-Nicolson method. Furthermore, we compare required CPU times between the Brennan and Schwartz algorithm and the projected SOR method with different values of m and n. Finally, we compare the delta functions of the option prices computed using the Crank-Nicolson method and the Runge-Kutta method.

In order to report maximum errors for numerical solutions, we computed reference solutions for the American call and put option problems numerically. These reference solutions were computed with a dense grid defined by $(m, n) = (131072, 4096)$, the central finite difference scheme, the Runge-Kutta scheme, and the Brennan and Schwartz algorithm. In the following examples the computational domain is $[0, 50] \times [0, 1]$. Numerical experiments were performed on an HP J5600 workstation.

5.1 Example I

First, we compare the convergence rates of the implicit Euler scheme, the Crank-Nicolson method, the BDF2 formula and the Runge-Kutta scheme. Also the stability of these schemes is studied numerically. The spatial derivatives were approximated by the central finite differences described in Section 3.1 and the complementarity problems at each time step were solved with the Brennan and Schwartz algorithm. In this numerical experiment the American call option problem (3) is described by the parameter values $\sigma = 0.6$, $r = 0.25$, $E = 10$, $d = 0.2$ and $T = 1$.

For parabolic partial differential equations the Crank-Nicolson method, the BDF2 formula and the Runge-Kutta scheme are second-order accurate in time. Due to the nonsmooth initial function, the convergence rates of these three methods are reduced. The L-stable BDF2 formula and Runge-Kutta scheme are more accurate than only unconditionally stable Crank-Nicolson method. This can be seen from Tables 1 and 2 where we report the maximum errors of numerical solutions. We have also computed ratios between the error with n and the error with $n/2$. This estimates the convergence rate of the time discretization schemes. With moderate time steps the convergence rate and accuracy of the Runge-Kutta scheme are higher than with other schemes. Especially the convergence of the Crank-Nicolson method is slow with larger time steps. The first-order accurate implicit Euler scheme converges as expected. It is never as accurate as the BDF2 formula and the Runge-Kutta scheme, but for large time steps it is more accurate than the Crank-Nicolson method.

In the case of the BDF2 formula and the Runge-Kutta scheme the numerical solutions become more accurate when the number of space step m is increased from 2047 to 8191; see Tables 1 and 2. While this is not true for the Crank-Nicolson method. Due to the poor stability properties of the Crank-Nicolson method the accuracy of the solution decreases when the number of space steps is increased. In Figure 1, we have plotted error functions in the case of the Crank-Nicolson method and the Runge-Kutta scheme for various number of time steps, $n = 8, 16, 32$, and for a fixed number of space

	Implicit-Euler		Crank-Nicolson		BDF2		Runge-Kutta	
n	error	ratio	error	ratio	error	ratio	error	ratio
2	1.905E-1		2.739E-1		1.111E-1		2.307E-2	
4	1.015E-1	1.88	1.403E-1	1.95	3.084E-2	3.60	3.680E-3	6.27
8	5.305E-2	1.91	6.929E-2	2.02	7.136E-3	4.32	7.892E-4	4.66
16	2.732E-2	1.94	3.265E-2	2.12	2.056E-3	3.47	1.830E-4	4.31
32	1.392E-2	1.96	1.422E-2	2.30	6.381E-4	3.22	5.099E-5	3.59
64	7.044E-3	1.98	5.237E-3	2.71	1.944E-4	3.28	1.606E-5	3.17
128	3.547E-3	1.99	1.271E-3	4.12	5.602E-5	3.47	3.776E-6	4.25
256	1.780E-3	1.99	9.369E-5	13.57	1.524E-5	3.68	2.105E-6	1.79
512	8.920E-4	2.00	2.182E-6	42.93	4.100E-6	3.72	2.105E-6	1.00

Table 1: The maximum errors and the time convergence rates when $m = 2047$.

	Implicit-Euler		Crank-Nicolson		BDF2		Runge-Kutta	
n	error	ratio	error	ratio	error	ratio	error	ratio
2	1.905E-1		2.773E-1		1.111E-1		2.307E-2	
4	1.016E-1	1.88	1.438E-1	1.93	3.084E-2	3.60	3.681E-3	6.27
8	5.305E-2	1.91	7.280E-2	1.97	7.136E-3	4.32	7.904E-4	4.66
16	2.732E-2	1.94	3.610E-2	2.02	2.056E-3	3.47	1.841E-4	4.29
32	1.392E-2	1.96	1.751E-2	2.06	6.382E-4	3.22	5.107E-5	3.60
64	7.044E-3	1.98	8.184E-3	2.14	1.944E-4	3.28	1.473E-5	3.47
128	3.547E-3	1.99	3.551E-3	2.30	5.600E-5	3.47	3.920E-6	3.76
256	1.780E-3	1.99	1.306E-3	2.72	1.519E-5	3.69	9.892E-7	3.96
512	8.919E-4	2.00	3.161E-4	4.13	3.966E-6	3.83	2.382E-7	4.15

Table 2: The maximum errors and the time convergence rates when $m = 8191$.

step, $m = 2047$. Undesired oscillations are clearly visible in the errors for the Crank-Nicolson method and the errors are significantly smaller for the Runge-Kutta scheme.

5.2 Example II

Next, we compare required CPU times when the Brennan and Schwartz algorithm and the projected SOR method are applied for the American call option problem. The discretization was performed with the central finite difference schemes and the Runge-Kutta scheme. Furthermore, the same parameter values were used as in the first numerical experiment.

The stopping criterion for the iterative projected SOR method was chosen in such a way that the maximum error was at the most 10 percent larger than with the direct Brennan and Schwartz algorithm. The overrelaxation parameter ω in the projected SOR was optimized so that iterations would converge as fast as possible. The values of ω are given for the different values of m and n in Table 3.

The results of this example are reported in Table 3. For the Brennan and Schwartz

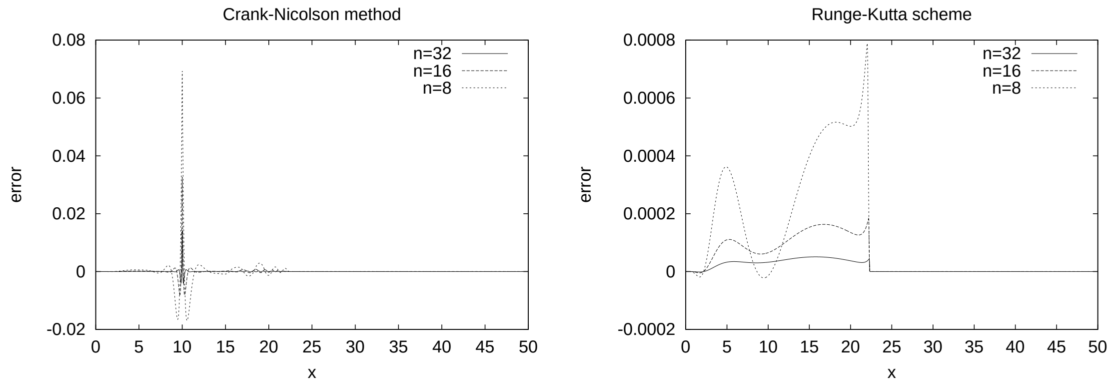

Figure 1: The error functions of the numerical solutions in the cases of the Crank-Nicolson method (left) and the Runge-Kutta scheme (right) when $m = 2047$ and n varies.

		PSOR				B-S	
m	n	error	CPU	ω	iter ave	error	CPU
127	16	2.684E-4	0.026	1.5	14.063	2.641E-4	0.006
255	32	1.393E-4	0.146	1.6	20.219	1.341E-4	0.022
511	64	2.271E-5	1.191	1.71	44.047	2.144E-5	0.089
1023	128	1.221E-5	6.235	1.79	55.133	1.190E-5	0.353
2047	256	2.182E-6	35.230	1.88	83.223	2.105E-6	1.392
4095	512	5.705E-7	311.672	1.90	184.586	5.765E-7	5.586

Table 3: The maximum errors and the CPU times in the cases of the projected SOR method and the Brennan and Schwartz (B-S) algorithm.

algorithm the required CPU time increases linearly with respect to the number of unknowns. The CPU time for every (m, n) pair are several times smaller than the corresponding time for the projected SOR method. The CPU times for the projected SOR method grow approximately like $(mn)^{1.35}$.

5.3 Example III

In the last numerical experiment, we study the influence of the time discretization schemes to the delta function, which is the derivative of the option price with respect to x. In Figure 2, we have plotted the delta functions for the American call and put options in the cases of the Crank-Nicolson method and the Runge-Kutta scheme. The Runge-Kutta scheme leads to smooth delta functions while an undesired oscillation arising from the use of the Crank-Nicolson method can be clearly seen. This oscillation is focused near to the exercise price E, which is where the space derivative of the initial function is discontinuous. The parameter values were same as in the first experiment and the gird defined by $(m, n) = (2047, 32)$ was used. In order to demonstrate the oscillation the number of time step n was relatively small.

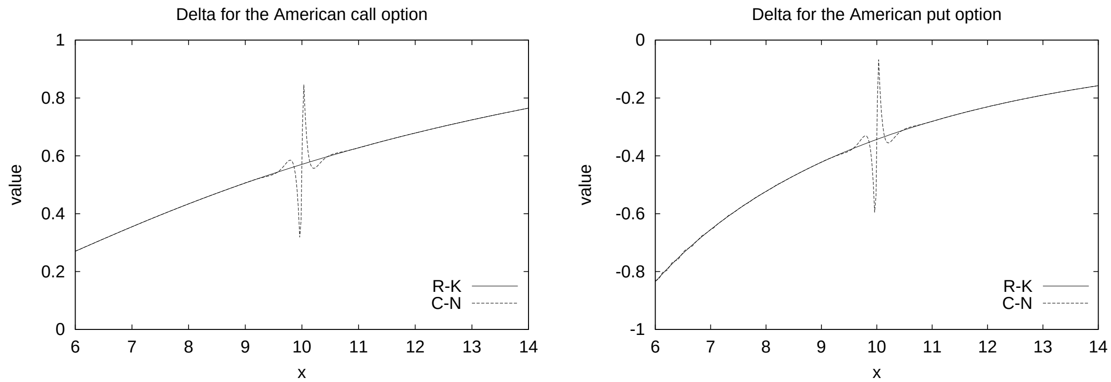

Figure 2: The delta functions in the cases of the American call option (left) and put option (right) when the Crank-Nicolson method and the Runge-Kutta scheme are applied, $(m, n) = (2047, 32)$.

6 Conclusions

In this paper, we have considered finite difference methods for pricing American options. We have studied time discretization schemes for the Black-Scholes partial differential equation and introduced a reformulation of the Brennan and Schwartz algorithm.

We applied the BDF2 formula and the Runge-Kutta scheme for the time discretization and in the numerical experiments these were found out to be more accurate than the Crank-Nicolson method. The discontinuous first derivative of the payoff function decreases convergence rate. We demonstrated that the convergence rates of the L-stable BDF2 formula and Runge-Kutta scheme are better than the convergence rate of the Crank-Nicolson method. In addition, we showed that the Runge-Kutta scheme produces much less oscillations than only unconditionally stable Crank-Nicolson method.

In the option pricing the complementarity problems need to be solved at each time step. We reformulated the Brennan and Schwartz algorithm in a way that the LU decomposition can be used. Furthermore, we studied the Elliott-Ockendon algorithm and we showed that also this algorithm can be formulated into a form where the LU decomposition is employed. In these reformulations the early exercise constraint is handled in a modified backward substitution using a projection.

Furthermore, in the numerical experiments we compared the required CPU times between the direct Brennan and Schwartz algorithm and the iterative projected SOR method. It was observed that the Brennan and Schwartz algorithm was several times faster on coarser grids and tens of times faster on finer grids. Besides, it is difficult to find an optimal stopping criterion and overrelaxation parameter for the projected SOR method while the Brennan and Schwartz algorithm is parameter free and, thus, much easier to use.

Acknowledgements

The research was supported by the University of Jyväskylä and by the Academy of Finland, grant #53588. The authors would like to thank Dr. Janne Martikainen and Dr. Raino A. E. Mäkinen for helpful comments.

References

[1] F. Black and M. Scholes, *The pricing of options and corporate liabilities*, Journal of Political Economy, 81 (1973), pp. 637–654.

[2] M. J. Brennan and E. S. Schwartz, *The valuation of American put options*, Journal of Finance, 32 (1977), pp. 449–462.

[3] M. Broadie and J. Detemple, *American option valuation: New bounds, approximations, and a comparison of existing methods*, Review of Financial Studies, 9 (1996), pp. 1211–1250.

[4] J. R. Cash, *Two new finite difference schemes for parabolic equations*, SIAM Journal on Numerical Analysis, 21 (1984), pp. 433–446.

[5] N. Clarke and K. Parrott, *Multigrid for American option pricing with stochastic volatility*, Applied Mathematical Finance, 6 (1999), pp. 177–195.

[6] R. W. Cottle, J.-S. Pang, and R. E. Stone, *The linear complementarity problem*, Computer Science and Scientific Computing, Academic Press Inc., Boston, MA, 1992.

[7] M. A. H. Dempster and J. P. Hutton, *Fast numerical valuation of American, exotic and complex options*, Applied Mathematical Finance, 4 (1997), pp. 1–20.

[8] ——, *Pricing American stock options by linear programming*, Mathematical Finance, 9 (1999), pp. 229–254.

[9] C. M. Elliott and J. R. Ockendon, *Weak and variational methods for moving boundary problems*, vol. 59 of Research Notes in Mathematics, Pitman, Boston, Mass., 1982.

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

[11] R. Geske and K. Shastri, *Valuation by approximation: A comparison of alternative option valuation techniques*, Journal of Financial Quantitative Analysis, 20 (1985), pp. 45–71.

[12] R. Glowinski, *Numerical methods for nonlinear variational problems*, Springer Series in Computational Physics, Springer-Verlag, New York, 1984.

[13] R. Glowinski, *Splitting methods for the numerical solution of the incompressible Navier-Stokes equations*, in Vistas in applied mathematics, Optimization Software, New York, 1986, pp. 57–95.

[14] E. Hairer and G. Wanner, *Solving ordinary differential equations. II: Stiff and differential-algebraic problems*, vol. 14 of Springer Series in Computational Mathematics, Springer-Verlag, Berlin, second ed., 1996.

[15] J. Huang and J.-S. Pang, *Option pricing and linear complementarity*, Journal of Computational Finance, 2 (1998), pp. 31–60.

[16] J. Hull and A. White, *Valuing derivatives securities using the explicit finite difference method*, Journal of Financial and Quantitative Analysis, 25 (1990), pp. 87–100.

[17] J. C. Hull, *Options, futures, and other derivatives*, Prentice-Hall, 3 ed., 1997.

[18] S. Ikonen and J. Toivanen, *Operator splitting methods for American option pricing*, Applied Mathematical Letters. To appear.

[19] P. Jaillet, D. Lamberton, and B. Lapeyre, *Variational inequalities and the pricing of American options*, Acta Applicandae Mathematicae, 21 (1990), pp. 263–289.

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

[21] J. D. Lawson and J. L. Morris, *The extrapolation of first order methods for parabolic partial differential equations. I*, SIAM Journal on Numerical Analysis, 15 (1978), pp. 1212–1224.

[22] B. J. McCartin and S. M. Labadie, *Accurate and efficient pricing of vanilla stock options via the Crandall-Douglas scheme*, Applied Mathematics and Computation, 143 (2003), pp. 39–60.

[23] K. W. Morton, *Numerical solution of convection-diffusion problems*, vol. 12 of Applied Mathematics and Mathematical Computation, Chapman & Hall, London, 1996.

[24] B. F. Nielsen, O. Skavhaug, and A. Tveito, *Penalty and front-fixing methods for the numerical solution of american option problems*, Journal of Computational Finance, 5 (2002), pp. 69–97.

[25] C. W. Oosterlee, *On multigrid for linear complementarity problems with application to American-style options*, Electronic Transactions on Numerical Analysis, 15 (2003), pp. 165–185.

[26] D. M. Pooley, K. R. Vetzal, and P. A. Forsyth, *Convergence remedies for non-smooth payoffs in option pricing*, Journal of Computational Finance, 6 (2003), pp. 25–40.

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

[28] R. Seydel, *Tools for computational finance*, Universitext, Springer-Verlag, Berlin, 2002.

[29] D. Tavella and C. Randall, *Pricing financial instrument*, John Wiley & Sons, 2000.

[30] A. Tveito and R. Winther, *Introduction to partial differential equations*, Springer-Verlag, New York, 1998.

[31] P. Wilmott, J. Dewynne, and S. Howison, *Option pricing: mathematical models and computation*, Oxford Financial Press, Oxford, 1993.

[32] G. Windisch, *M-matrices in numerical analysis*, vol. 115, BSB B. G. Teubner Verlagsgesellschaft, Leipzig, 1989.

[33] R. Zvan, P. Forsyth, and K. Vetzal, *Negative coefficients in two factor option pricing models*, Journal of Computational Finance, 7 (2003), pp. 37–73.