symmetry

MDPI

Article

Iterative Speedup by Utilizing Symmetric Data in Pricing Options with Two Risky Assets

Dohyun Pak [1], Changkyu Han [2] and Won-Tak Hong [1,*]

[1] Department of Mathematics & Finance, Gachon University, 1342 Seongnamdaero, Sujeong-gu, Seongnam-si, Gyeonggi-do 13120, Korea; dhmpak@gachon.ac.kr

[2] Department of Risk Management, Kiwoom Securities Co., Ltd., 18 Yeouinaru-ro 4(sa)-gil, Yeongdeungpo-gu, Seoul 07331, Korea; ckhan@kiwoom.com

* Correspondence: wontak@gachon.ac.kr; Tel.: +82-31-750-5387

Academic Editors: Doo-Soon Park and Shu-Ching Chen

Received: 29 September 2016; Accepted: 13 January 2017; Published: 21 January 2017

Abstract: The Crank–Nicolson method can be used to solve the Black–Scholes partial differential equation in one-dimension when both accuracy and stability is of concern. In multi-dimensions, however, discretizing the computational grid with a Crank–Nicolson scheme requires significantly large storage compared to the widely adopted Operator Splitting Method (OSM). We found that symmetrizing the system of equations resulting from the Crank–Nicolson discretization help us to use the standard pre-conditioner for the iterative matrix solver and reduces the number of iterations to get an accurate option values. In addition, the number of iterations that is required to solve the preconditioned system, resulting from the proposed iterative Crank–Nicolson scheme, does not grow with the size of the system. Thus, we can effectively reduce the order of complexity in multidimensional option pricing. The numerical results are compared to the one with implicit Operator Splitting Method (OSM) to show the effectiveness.

Keywords: Black–Scholes equation; Operator Splitting Method (OSM); Crank–Nicolson; iterative solver

1. Introduction

The multidimensional Black–Scholes equation is often used to model options written on multiple assets. One of the traditional methods both in practice and research for discretizing multidimensional Black–Scholes equations is the Operator Splitting Method (OSM) [1–3]. To solve multidimensional Black–Scholes equations, the OSM solves one-dimensional Black–Scholes equations in turn. Thus, it is possible to use a highly efficient tridiagonal matrix solver as in one dimension [2]. The OSM converges at first order in time and second order in space if we discretize multidimensional Black–Scholes equations with an implicit central difference method. In one-dimensional cases, in which the option is written on a single asset, the order of convergence in time can be improved to the second order without too many difficulties if one replaces a time integration scheme with Crank–Nicolson.

In multi-dimensions, however, replacing the time integration scheme is not straightforward as in one-dimension. In general, multiple assets are correlated, and thus the multidimensional Black–Scholes equation has corresponding cross partial derivative terms, which do not appear in one-dimensional Black–Scholes equations. Sometimes, these partial derivatives are not calculated and are assumed to be known, which is easy to implement but leads to inaccuracy under high correlation and large volatilities. If an implicit scheme along with OSM is applied to discretize the mixed partial derivatives appearing in the equation, the resulting system becomes no longer tridiagonal and the Thomas algorithm is not applicable. Therefore, another matrix solver or more advanced multidimensional modeling with radial basis functions [4,5] have to be used at the cost of computation time. In practice, however,

the Thomas algorithm is indispensable because of its highly efficient nature. Therefore, one avoids fully implicit discretization of multidimensional Black–Scholes equations by replacing the mixed partial derivative terms with known values so that other partial derivative terms can be discretized implicitly. In this way, practitioners could use the OSM with the most efficient matrix solver, the Thomas algorithm, but end up with only first order convergence in time.

Having second order convergence both in space and time is important if highly accurate option values are of concern. A second order convergence in time is helpful for the practitioners if an option has complex payoff structures or parameter adjustment is needed before the maturity. In addition, Greeks (Δ, Γ, etc.) are usually calculated in the post-processing stage using the computed option values, and more significant digits on the option price will improve the accuracy of Greeks. However, in practice, the time to obtain one more significant digit on the option price grows exponentially with the first order convergence speed in time. Therefore, it would be natural to try second order schemes such as Crank–Nicolson, BDF-2 [6,7], etc. Each of the second order schemes has its own advantages and disadvantages. We use the Crank–Nicolson scheme, which has second-order convergence in space and time.

A straightforward Crank–Nicolson discretization of the multidimensional Black–Scholes equation produces a system that makes the direct solver unattractive in terms of the computational effort. However, a simple modification to symmetrize the system helps to solve the system more efficiently with an iterative solver. We found that a standard preconditioner for the iterative solver significantly reduces the number of iterations for those problems that we tested after symmetrization. Finally, we compare the computational complexity of the iterative Crank–Nicolson method and OSM.

2. Nomenclature

We use bold uppercase letters to represent matrix $\mathbf{M}$, bold lowercase letters to denote vector $\mathbf{v}$, superscript' to denote transpose, and superscript (n) to indicate n-th time period. All vectors that are used in this paper are assumed to be a column vector.

Let the price of the derivative $V(t,x,y)$, where x and y are two different asset prices, and $V(t,x,y)$ is the solution of the following two-dimensional Black–Scholes partial differential equation [8,9]:

$$\frac{\partial V}{\partial t} + \frac{1}{2}\sigma_1^2 x^2 \frac{\partial^2 V}{\partial x^2} + \frac{1}{2}\sigma_2^2 y^2 \frac{\partial^2 V}{\partial y^2} + \rho\sigma_1\sigma_2 xy \frac{\partial^2 V}{\partial x \partial y} + rx\frac{\partial V}{\partial x} + ry\frac{\partial V}{\partial y} = rV, \tag{1}$$

where the domain is defined by $\{(t,x,y) \mid t \in (0,T\,],\ x \geq 0\,,\ y \geq 0\}$. In Equation (1), σ_i is the volatility of the i-th asset , ρ is the correlation coefficient, and r is the risk-free interest rate. The maturity of the option V is denoted by subscript T. We express the final payoff of the option V in the following form:

$$V_T = V_T(x,y) = V(T,x,y) \quad , \forall x, y \in [0,\infty). \tag{2}$$

Big O notation is used to measure the growth rate of algorithm in terms of input size. For example, a function $f(N) = O(g(N))$ means that there exist positive numbers C and N_0 such that $f(N) \leq C \times g(N)$ for all $N > N_0$.

3. Implicit OSM

Operator Splitting Method (OSM) finds the solution of multi-dimensional version of Equation (1) by splitting the differential operator so that the multi-dimensional problem becomes several one-dimensional problems [2]. For brevity, let us consider two-dimensional version of OSM. Equation (1) can be rewritten as follows:

$$\partial_t V + \mathcal{L}_x V + \mathcal{L}_y V = 0, \tag{3}$$

where

$$\begin{aligned}\mathcal{L}_x V &= \frac{1}{2}\sigma_1^2 x^2 \frac{\partial^2 V}{\partial x^2} + \frac{1}{2}\rho\sigma_1\sigma_2 xy \frac{\partial^2 V}{\partial x \partial y} + rx\frac{\partial V}{\partial x} - \frac{1}{2}r,\\ \mathcal{L}_y V &= \frac{1}{2}\sigma_2^2 y^2 \frac{\partial^2 V}{\partial y^2} + \frac{1}{2}\rho\sigma_1\sigma_2 xy \frac{\partial^2 V}{\partial x \partial y} + ry\frac{\partial V}{\partial y} - \frac{1}{2}r.\end{aligned} \tag{4}$$

Given the final condition V_T, we find the solution at previous time, $T - \Delta t$, in two steps:

Step (1) Find the solution at $T - \frac{1}{2}\Delta t$, by discretizing the equation $\partial_t V + \mathcal{L}_x V = 0$ with the given V_T:

$$\begin{aligned}\frac{V_{i,j}^{(n+\frac{1}{2})}}{\Delta t} &+ \sigma_1^2 x_i^2 \frac{V_{i-1,j}^{(n+\frac{1}{2})} - 2V_{i,j}^{(n+\frac{1}{2})} + V_{i+1,j}^{(n+\frac{1}{2})}}{2\Delta x^2}\\ &+ rx\frac{V_{i+1,j}^{(n+\frac{1}{2})} - V_{i-1,j}^{(n+\frac{1}{2})}}{2\Delta x} - \frac{1}{2}rV_{i,j}^{(n+\frac{1}{2})}\\ &= -\rho\sigma_1\sigma_2 x_i y_j \frac{V_{i+1,j+1}^{(n+1)} + V_{i-1,j-1}^{(n+1)} - V_{i+1,j-1}^{(n+1)} - V_{i-1,j+1}^{(n+1)}}{8\Delta x\Delta y}\\ &+ \frac{V_{i,j}^{(n+1)}}{\Delta t}.\end{aligned} \tag{5}$$

Step (2) Find the solution at $T - \Delta t$, by discretizing the equation $\partial_t V + \mathcal{L}_y V = 0$ with the solution found in Step (1) as the given condition:

$$\begin{aligned}\frac{V_{i,j}^{(n)}}{\Delta t} &+ \sigma_2^2 y_j^2 \frac{V_{i,j-1}^{(n)} - 2V_{i,j}^{(n)} + V_{i,j+1}^{(n)}}{2\Delta y^2}\\ &+ ry_j\frac{V_{i,j+1}^{(n)} - V_{i,j-1}^{(n)}}{2\Delta y} - \frac{1}{2}rV_{i,j}^{(n)}\\ &= -\rho\sigma_1\sigma_2 x_i y_j \frac{V_{i+1,j+1}^{(n+\frac{1}{2})} + V_{i-1,j-1}^{(n+\frac{1}{2})} - V_{i+1,j-1}^{(n+\frac{1}{2})} - V_{i-1,j+1}^{(n+\frac{1}{2})}}{8\Delta x\Delta y}\\ &+ \frac{V_{i,j}^{(n+\frac{1}{2})}}{\Delta t}.\end{aligned} \tag{6}$$

Depending on the size of the time step Δt, we need to repeat the above Steps (1) and (2) to find the option price at $t = 0$. In the current presentation, we have only two steps to obtain solution at $T - \Delta t$ from time T because we have two operators in Equation (3). For a general m-dimensional problem, we need m steps to obtain a solution at $T - \Delta t$. In each Step, we have to solve the following linear system where $\mathbf{M}^{(n)}$ and $\mathbf{v}^{(n+1)}$ are known:

$$\mathbf{M}^{(n)}\mathbf{v}^{(n)} = \mathbf{v}^{(n+1)}. \tag{7}$$

The above discretization given in Equations (5) and (6) are not a *full* implicit discretization because of the partial derivative terms. Note that the mixed partial derivative terms are assumed to be known so that other partial derivative terms can be discretized implicitly. Thus, we should call it *semi* implicit discretization to be more precise, but, for the rest of this study, we will call the discretization given in Equation (7) an implicit discretization. This implicit discretization is unconditionally stable and has truncation error, $O(\Delta x^2, \Delta t)$. In addition, the matrix $\mathbf{M}^{(n)}$ in Equation (7) is tridiagonal (depending on the boundary condition, the matrix $\mathbf{M}^{(n)}$ may not be exactly tridigonal; however, it could be converted to tridiagonal [10]). Thus, the system can be effectively solved by Thomas Algorithm [11], which is the main feature of implicit OSM.

4. Iterative Crank–Nicolson Method

We propose an iterative Crank–Nicolson finite difference discretization of Equation (1) on a general non-uniform grid. Instead of solving a smaller size one-dimensional problem repeatedly, as in the Operator Splitting Method (OSM), we propose to fully discretize the two asset Black–Scholes equation with a Crank–Nicolson scheme. The discretization can be solved efficiently by a GMRES (Generalized Minimal Residual Method) [12,13] solver with preconditioning after symmetrization.

In the following, we use x_i and y_i for the price of the first and second asset on a (i,j)-th finite difference stencil. We define $h_i = x_{i+1} - x_i$ and $k_j = y_{j+1} - y_j$. We approximate the differential operator in the Equation (1) as follows:

$$\frac{\partial V}{\partial t}\Big|_{x_i,y_j} = \frac{V_{i,j}^{n+1} - V_{i,j}^{n}}{\Delta t}, \tag{8}$$

$$\begin{aligned}\frac{\partial V}{\partial x}\Big|_{x_i,y_j} &= \frac{-h_i\left(V_{i-1,j}^{n+1} + V_{i-1,j}^{n}\right)}{2h_{i-1}(h_{i-1}+h_i)} + \frac{(h_i - h_{i-1})\left(V_{i,j}^{n+1} + V_{i,j}^{n}\right)}{2h_{i-1}h_i} \\ &+ \frac{h_{i-1}\left(V_{i+1,j}^{n+1} + V_{i+1,j}^{n}\right)}{2h_i(h_{i-1}+h_i)},\end{aligned} \tag{9}$$

$$\begin{aligned}\frac{\partial V}{\partial y}\Big|_{x_i,y_j} &= -\frac{k_j\left(V_{i,j-1}^{n+1} + V_{i,j-1}^{n}\right)}{2k_{j-1}(k_{j-1}+k_j)} + \frac{(k_j - k_{j-1})\left(V_{i,j}^{n+1} + V_{i,j}^{n}\right)}{2k_{j-1}k_j} \\ &+ \frac{k_{j-1}\left(V_{i,j+1}^{n+1} + V_{i,j+1}^{n}\right)}{2k_j(k_{j-1}+k_j)},\end{aligned} \tag{10}$$

$$\frac{\partial^2 V}{\partial x^2}\Big|_{x_i,y_j} = \frac{V_{i-1,j}^{n+1} + V_{i-1,j}^{n}}{h_{i-1}(h_{i-1}+h_i)} - \frac{V_{i,j}^{n+1} + V_{i,j}^{n}}{h_{i-1}h_i} + \frac{V_{i+1,j}^{n+1} + V_{i+1,j}^{n}}{h_i(h_{i-1}+h_i)}, \tag{11}$$

$$\frac{\partial^2 V}{\partial y^2}\Big|_{x_i,y_j} = \frac{V_{i,j-1}^{n+1} + V_{i,j-1}^{n}}{k_{j-1}(k_{j-1}+k_j)} - \frac{V_{i,j}^{n+1} + V_{i,j}^{n}}{k_{j-1}k_j} + \frac{V_{i,j+1}^{n+1} + V_{i,j+1}^{n}}{k_j(k_{j-1}+k_j)}, \tag{12}$$

$$\begin{aligned}\frac{\partial^2 V}{\partial x \partial y}\Big|_{x_i,y_j} &= \frac{V_{i+1,j+1}^{n+1} - V_{i+1,j-1}^{n+1} - V_{i-1,j+1}^{n+1} + V_{i-1,j-1}^{n+1}}{2(h_{i-1}+h_i)(k_{j-1}+k_j)} \\ &+ \frac{V_{i+1,j+1}^{n} - V_{i+1,j-1}^{n} - V_{i-1,j+1}^{n} + V_{i-1,j-1}^{n}}{2(h_{i-1}+h_i)(k_{j-1}+k_j)},\end{aligned} \tag{13}$$

$$V|_{x_i,y_j} = \frac{V_{i,j}^{n+1} + V_{i,j}^{n}}{2}. \tag{14}$$

Applying Equations (8)−(14) to Equation (1), we obtain the following Crank–Nicolson discretization of two-dimensional Black–Scholes equation:

$$\begin{aligned}&{}^{1}\mathcal{L}_{i,j}V_{i-1,j-1}^{n} + {}^{2}\mathcal{L}_{i,j}V_{i-1,j}^{n} + {}^{3}\mathcal{L}_{i,j}V_{i-1,j+1}^{n} + {}^{4}\mathcal{L}_{i,j}V_{i,j-1}^{n} \\ &+{}^{5}\mathcal{L}_{i,j}V_{i,j}^{n} + {}^{6}\mathcal{L}_{i,j}V_{i,j+1}^{n} + {}^{7}\mathcal{L}_{i,j}V_{i+1,j-1}^{n} + {}^{8}\mathcal{L}_{i,j}V_{i+1,j}^{n} + {}^{9}\mathcal{L}_{i,j}V_{i+1,j+1}^{n} \\ &= {}^{1}\mathcal{R}_{i,j}V_{i-1,j-1}^{n+1} + {}^{2}\mathcal{R}_{i,j}V_{i-1,j}^{n+1} + {}^{3}\mathcal{R}_{i,j}V_{i-1,j+1}^{n+1} + {}^{4}\mathcal{R}_{i,j}V_{i,j-1}^{n+1} \\ &+{}^{5}\mathcal{R}_{i,j}V_{i,j}^{n+1} + {}^{6}\mathcal{R}_{i,j}V_{i,j+1}^{n+1} + {}^{7}\mathcal{R}_{i,j}V_{i+1,j-1}^{n+1} + {}^{8}\mathcal{R}_{i,j}V_{i+1,j}^{n+1} + {}^{9}\mathcal{R}_{i,j}V_{i+1,j+1}^{n+1},\end{aligned} \tag{15}$$

where

$$
\begin{aligned}
{}^{1}\mathcal{L}_{i,j} &= -\frac{\rho\sigma_1\sigma_2 x_i y_j}{(h_{i-1}+h_i)(k_{j-1}+k_j)}\frac{\Delta t}{2}, \\
{}^{2}\mathcal{L}_{i,j} &= \left(\frac{r x_i h_i}{h_{i-1}(h_{i-1}+h_i)} - \frac{\sigma_1^2 x_i^2}{h_{i-1}(h_{i-1}+h_i)}\right)\frac{\Delta t}{2}, \\
{}^{3}\mathcal{L}_{i,j} &= \frac{\rho\sigma_1\sigma_2 x_i y_j}{(h_{i-1}+h_i)(k_{j-1}+k_j)}\frac{\Delta t}{2}, \\
{}^{4}\mathcal{L}_{i,j} &= \left(\frac{r y_j k_j}{k_{j-1}(k_{j-1}+k_j)} - \frac{\sigma_2^2 y_j^2}{k_{j-1}(k_{j-1}+k_j)}\right)\frac{\Delta t}{2}, \\
{}^{5}\mathcal{L}_{i,j} &= \left(-\frac{r x_i(h_i-h_{i-1})}{h_{i-1}h_i} - \frac{r y_j(k_j-k_{j-1})}{k_{j-1}k_j} + \frac{\sigma_1^2 x_i^2}{h_{i-1}h_i} + \frac{\sigma_2^2 y_j^2}{k_{j-1}k_j}\right)\frac{\Delta t}{2} \\
&\quad + \frac{r\Delta t}{2} + 1, \\
{}^{6}\mathcal{L}_{i,j} &= \left(-\frac{r y_j k_{j-1}}{k_j(k_{j-1}+k_j)} - \frac{\sigma_2^2 y_j^2}{k_j(k_{j-1}+k_j)}\right)\frac{\Delta t}{2}, \\
{}^{7}\mathcal{L}_{i,j} &= \frac{\rho\sigma_1\sigma_2 x_i y_j}{(h_{i-1}+h_i)(k_{j-1}+k_j)}\frac{\Delta t}{2}, \\
{}^{8}\mathcal{L}_{i,j} &= \left(-\frac{r x_i h_{i-1}}{h_i(h_{i-1}+h_i)} - \frac{\sigma_1^2 x_i^2}{h_i(h_{i-1}+h_i)}\right)\frac{\Delta t}{2}, \\
{}^{9}\mathcal{L}_{i,j} &= -\frac{\rho\sigma_1\sigma_2 x_i y_j}{(h_{i-1}+h_i)(k_{j-1}+k_j)}\frac{\Delta t}{2},
\end{aligned}
$$

and

$$
\begin{aligned}
{}^{1}\mathcal{R}_{i,j} &= \frac{\rho\sigma_1\sigma_2 x_i y_j}{(h_{i-1}+h_i)(k_{j-1}+k_j)}\frac{\Delta t}{2}, \\
{}^{2}\mathcal{R}_{i,j} &= \left(-\frac{r x_i h_i}{h_{i-1}(h_{i-1}+h_i)} + \frac{\sigma_1^2 x_i^2}{h_{i-1}(h_{i-1}+h_i)}\right)\frac{\Delta t}{2}, \\
{}^{3}\mathcal{R}_{i,j} &= -\frac{\rho\sigma_1\sigma_2 x_i y_j}{(h_{i-1}+h_i)(k_{j-1}+k_j)}\frac{\Delta t}{2}, \\
{}^{4}\mathcal{R}_{i,j} &= \left(-\frac{r y_j k_j}{k_{j-1}(k_{j-1}+k_j)} + \frac{\sigma_2^2 y_j^2}{k_{j-1}(k_{j-1}+k_j)}\right)\frac{\Delta t}{2}, \\
{}^{5}\mathcal{R}_{i,j} &= \left(\frac{r x_i(h_i-h_{i-1})}{h_{i-1}h_i} + \frac{r y_j(k_j-k_{j-1})}{k_{j-1}k_j} - \frac{\sigma_1^2 x_i^2}{h_{i-1}h_i} - \frac{\sigma_2^2 y_j^2}{k_{j-1}k_j}\right)\frac{\Delta t}{2} \\
&\quad - \frac{r\Delta t}{2} + 1, \\
{}^{6}\mathcal{R}_{i,j} &= \left(\frac{r y_j k_{j-1}}{k_j(k_{j-1}+k_j)} + \frac{\sigma_2^2 y_j^2}{k_j(k_{j-1}+k_j)}\right)\frac{\Delta t}{2}, \\
{}^{7}\mathcal{R}_{i,j} &= -\frac{\rho\sigma_1\sigma_2 x_i y_j}{(h_{i-1}+h_i)(k_{j-1}+k_j)}\frac{\Delta t}{2}, \\
{}^{8}\mathcal{R}_{i,j} &= \left(\frac{r x_i h_{i-1}}{h_i(h_{i-1}+h_i)} + \frac{\sigma_1^2 x_i^2}{h_i(h_{i-1}+h_i)}\right)\frac{\Delta t}{2}, \\
{}^{9}\mathcal{R}_{i,j} &= \frac{\rho\sigma_1\sigma_2 x_i y_j}{(h_{i-1}+h_i)(k_{j-1}+k_j)}\frac{\Delta t}{2}.
\end{aligned}
$$

The symmetrized iterative Crank–Nicolson method for the two asset Black–Scholes equation is described as follows.

Step (1) Get a linear system by discretizing Equation (1) with the Crank–Nicolson scheme. The following equation is matrix-vector form of the Equation (15) :

$$\boldsymbol{\mathcal{L}}^{(n)}\mathbf{v}^{(n)} = \boldsymbol{\mathcal{R}}^{(n+1)}\mathbf{v}^{(n+1)}, \tag{16}$$

where

$$\begin{aligned}\mathbf{v}^{(\mathbf{n})} &= (V^n_{i-1,j-1}, V^n_{i-1,j}, V^n_{i-1,j+1}, V^n_{i,j-1}, V^n_{i,j}, \\ &\quad V^n_{i,j+1}, V^n_{i+1,j-1}, V^n_{i+1,j}, V^n_{i+1,j+1})', \end{aligned} \tag{17}$$

$$\begin{aligned}\mathbf{v}^{(\mathbf{n+1})} &= (V^{n+1}_{i-1,j-1}, V^{n+1}_{i-1,j}, V^{n+1}_{i-1,j+1}, V^{n+1}_{i,j-1}, V^{n+1}_{i,j}, \\ &\quad V^{n+1}_{i,j+1}, V^{n+1}_{i+1,j-1}, V^{n+1}_{i+1,j}, V^{n+1}_{i+1,j+1})'. \end{aligned} \tag{18}$$

Note that the data, $\boldsymbol{\mathcal{L}}^{(n)}$ and $\boldsymbol{\mathcal{R}}^{(n)}$, grows quadratically in terms of the grid points. In addition, the system in Equation (16) is non-symmetric.

Step (2) Apply appropriate boundary conditions to the $\boldsymbol{\mathcal{L}}^{(n)}$ and $\boldsymbol{\mathcal{R}}^{(n+1)}$. Depending on the option type, the boundary condition is either given as a linear boundary condition on the truncated interface or an essential boundary condition where the price of the option is zero. We denote the boundary condition imposed system as follows:

$$\boldsymbol{\mathcal{L}}_{\mathrm{bc}}{}^{(n)}\mathbf{v}^{(n)} = \boldsymbol{\mathcal{R}}_{\mathrm{bc}}{}^{(n+1)}\mathbf{v}^{(n+1)}. \tag{19}$$

Step (3) Symmetrize the system given in Equation (19) as follows:

$$\left(\boldsymbol{\mathcal{L}}_{\mathrm{bc}}{}^{(n)}\right)'\boldsymbol{\mathcal{L}}_{\mathrm{bc}}{}^{(n)}\mathbf{v}^{(n)} = \left(\boldsymbol{\mathcal{L}}_{\mathrm{bc}}{}^{(n)}\right)'\boldsymbol{\mathcal{R}}_{\mathrm{bc}}{}^{(n+1)}\mathbf{v}^{(n+1)}, \tag{20}$$

where $\left(\boldsymbol{\mathcal{L}}_{\mathrm{bc}}{}^{(n)}\right)'$ is the transpose of $\boldsymbol{\mathcal{L}}_{\mathrm{bc}}{}^{(n)}$.

Step (4) Create preconditioned matrix $\mathbf{P}$ with $\left(\boldsymbol{\mathcal{L}}_{\mathrm{bc}}{}^{(n)}\right)'\boldsymbol{\mathcal{L}}_{\mathrm{bc}}{}^{(n)}$ using incomplete LU factorization (where LU stands for lower and upper triangular matrix). The choice of the preconditioner is more important than the choice of the Krylov iterative method such as GMRES [14,15]. The effectiveness of preconditioner $\mathbf{P}$ created by incomplete LU factorization is measured by how well $\mathbf{P}^{-1}$ approximates the $\left((\boldsymbol{\mathcal{L}}_{\mathrm{bc}}{}^{(n)})'\boldsymbol{\mathcal{L}}_{\mathrm{bc}}{}^{(n)}\right)^{-1}$.

Step (5) Solve Equation (20) repetitively using GMRES with the preconditioner $\mathbf{P}$ and the previous solution vector $\mathbf{v}^{(n+1)}$ as an initial guess until we find the option price $\mathbf{v}^{(0)}$. Use the final condition $\mathbf{v}^{(N)} = V_T = V(T, x_i, y_j)$ to start the iteration. We use the following split preconditioning with the incomplete LU factors, $\mathbf{P}_L^{-1}$ and $\mathbf{P}_R^{-1}$:

$$\mathbf{P}_L^{-1}\left(\boldsymbol{\mathcal{L}}_{\mathrm{bc}}{}^{(n)}\right)'\boldsymbol{\mathcal{L}}_{\mathrm{bc}}{}^{(n)}\mathbf{P}_R^{-1}\mathbf{u}^{(n)} = \mathbf{P}_L^{-1}\left(\boldsymbol{\mathcal{L}}_{\mathrm{bc}}{}^{(n)}\right)'\boldsymbol{\mathcal{R}}_{\mathrm{bc}}{}^{(n+1)}\mathbf{v}^{(n+1)}, \tag{21}$$

$$\mathbf{P}_R^{-1}\mathbf{u}^{(n)} = \mathbf{v}^{(n)}. \tag{22}$$

The previous solution vector, $\mathbf{v}^{(n+1)}$, is used as an initial guess of $\mathbf{v}^{(n)}$ to solve the Equations (21) and (22).

Thus far, we have explained the general procedure of the iterative Crank–Nicolson method for two asset Black–Scholes equations. The idea can be extended to the three asset Black–Scholes equation:

$$\begin{aligned}&\frac{\partial V}{\partial t}+\frac{1}{2}\sigma_1^2x^2\frac{\partial^2V}{\partial x^2}+\frac{1}{2}\sigma_2^2y^2\frac{\partial^2V}{\partial y^2}+\frac{1}{2}\sigma_3^2z^2\frac{\partial^2V}{\partial z^2}\\&\quad+\rho_{12}\sigma_1\sigma_2xy\frac{\partial^2V}{\partial x\partial y}+\rho_{23}\sigma_2\sigma_3yz\frac{\partial^2V}{\partial y\partial z}+\rho_{13}\sigma_1\sigma_3xz\frac{\partial^2V}{\partial x\partial z}+rx\frac{\partial V}{\partial x}+ry\frac{\partial V}{\partial y}+rz\frac{\partial V}{\partial z}=rV,\end{aligned}\tag{23}$$

where $\{(t,x,y,z)\mid t\in(0,T\,],\ x\geq 0\,,\ y\geq 0\,,\ z\geq 0\}$. The difference with two asset case is that we have four additional terms to discretize. The discretization is essentially the same with different indices. Thus, we obtain equations that are similar to Equation (15) but have 27 terms on each side instead of nine terms. In vector notation, the procedure given above for Steps (1) to (5) is the same for the three asset case.

The oscillations in the solution due to non-smooth initial data are a well-known drawback of the Crank–Nicolson method. Thus, if the final condition of the given option has non-smoothness, the computational grid can be prepared so that the option strike price agrees to one of the midpoints in the grid or the $\mathbf{v}^{(n+1)}$ in Equation (16) can be replaced with $\tilde{\mathbf{v}}^{(n+1)}$ by a simple moving average—for example, on a uniform grid, we can use the following equation:

$$\tilde{\mathbf{v}}_{i,j}^{(n+1)}=\frac{V_{i-1,j}^{n+1}+V_{i,j-1}^{n+1}+4V_{i,j}^{n+1}+V_{i+1,j}^{n+1}+V_{i,j+1}^{n+1}}{8}.\tag{24}$$

5. Computational Perspective of the Iterative Crank–Nicolson Method

We will demonstrate the iterative Crank–Nicolson method with some European style options with two assets. In the following, we study computational cost and order of convergence of the iterative Crank-Nicolson method.

5.1. Some Numerical Examples

All numerical computations in this section are performed on the finite domain $[0,300]\times[0,300]$. The relative error (%) in the maximum norm is calculated by the following equation:

$$\text{Rel.err}(\%)=\frac{||\mathbf{u}_{\text{app.}}-\mathbf{u}_{\text{ref.}}||_\infty}{||\mathbf{u}_{\text{ref.}}||_\infty}\times 100(\%),\tag{25}$$

where $\mathbf{u}_{\text{app.}}$ is the computed numerical solution and $\mathbf{u}_{\text{ref.}}$ is the reference solution. We present numerical results with the following three different final payoffs V_T and reference solution V_{ref} [16,17]. M is the bivariate normal distribution and K is the strike price.

The payoffs in Equations (26), (28) and (30) are carefully chosen so that (1) V_T in Equation (26) is symmetric and discontinuous; (2) V_T in Equation (28) is non-symmetric and discontinuous; and (3) V_T in Equation (30) is symmetric and continuous:

(1) Cash or Nothing

$$\begin{aligned}V(T,x,y)&=100\text{ if }\min(x,y)\geq K,&(26)\\V_{\text{ref}}&=Ke^{-rT}M(d_1,d_2;\rho),&(27)\\d_1&=\frac{\ln(\frac{x}{K})+(r-\frac{1}{2}\sigma_1^2)T}{\sigma_1\sqrt{T}},\\d_2&=\frac{\ln(\frac{y}{K})+(r-\frac{1}{2}\sigma_2^2)T}{\sigma_2\sqrt{T}},\end{aligned}$$

with parameters: $\rho=0.5,\ \ r=0.02,\ \ K=75,\ \ \sigma_1=0.15,\sigma_2=0.2,T=1.0$. Figure 1a,b shows V_T and V_{ref}, respectively.

(2) Call

$$\begin{aligned} V(T,x,y) &= \max(y-K_2,0) \text{ if } x \geq K_1, && (28)\\ V_{\text{ref}} &= yM(d_2+\sigma_2\sqrt{T}, d_1+\rho\sigma_2\sqrt{T};\rho), && (29)\\ &- K_2e^{-rT}M(d_2,d_1;\rho),\\ d_1 &= \frac{\ln(\frac{x}{K_1})+(r-\frac{1}{2}\sigma_1^2)T}{\sigma_1\sqrt{T}},\\ d_2 &= \frac{\ln(\frac{y}{K_2})+(r-\frac{1}{2}\sigma_2^2)T}{\sigma_2\sqrt{T}}, \end{aligned}$$

with parameters: $\rho = 0.5,\ r = 0.02,\ K_1 = 75,\ K_2 = 85,\ \sigma_1 = 0.15, \sigma_2 = 0.2, T = 1.0$. Figure 1c shows V_T and Figure 1d shows V_{ref}.

(3) Basket

$$V(T,x,y) = \max(x+y-K,0). \quad (30)$$

V_{ref} is approximated with the formula given in [18] with parameters: $\rho = 0.5$, $r = 0.02$, $K = 150$, $\sigma_1 = 0.15$, $\sigma_2 = 0.2$, $T = 1.0$. Figure 1e shows V_T and Figure 1f shows V_{ref}.

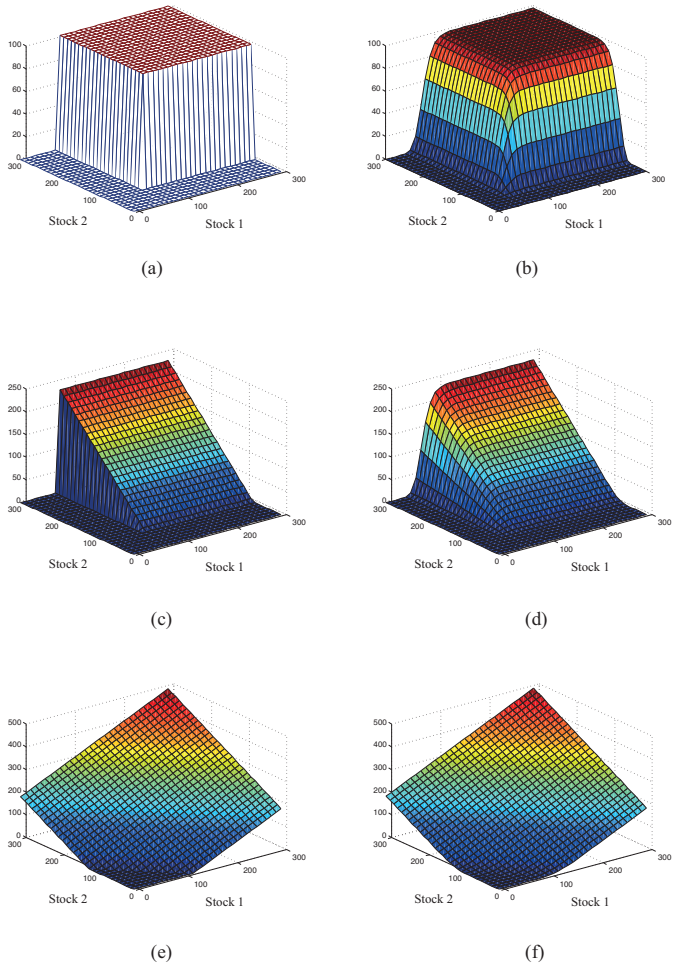

Figure 1. The surface of option payoff V_T and V_{ref} : (**a**) cash or nothing V_T; (**b**) cash or nothing V_{ref}; (**c**) two asset call V_T; (**d**) two asset call V_{ref}; (**e**) basket call V_T; and (**f**) basket call V_{ref}.

We use sparse storage for all matrices to hold the data throughout the test. In addition, the drop tolerance is set to 10^{-7} in the incomplete LU factorization to create preconditioner and 10^{-8} is used for GMRES stopping tolerance. We observed that the solution is reached within two iterations for each time step in all examples that we considered. These tolerances show dependency on the grid size for the problems that we considered and could be optimized, but we did not investigate the effect, as these parameters gave us an accurate numerical solution. Before we proceed to test the iterative Crank–Nicolson method for different payoffs, we show the symmetrization effect, Equation (20). The comparison between non-symmetrized system, given in Equation (19), and the symmetrized system is shown in Figure 2. The cash or nothing payoff is used with those parameters given in Equation (27). The slope for both symmetric and non-symmetric case is approximately constant, which suggests that the number of iteration per time steps and the number of iteration per spatical discretization does not grow as grid refinement.

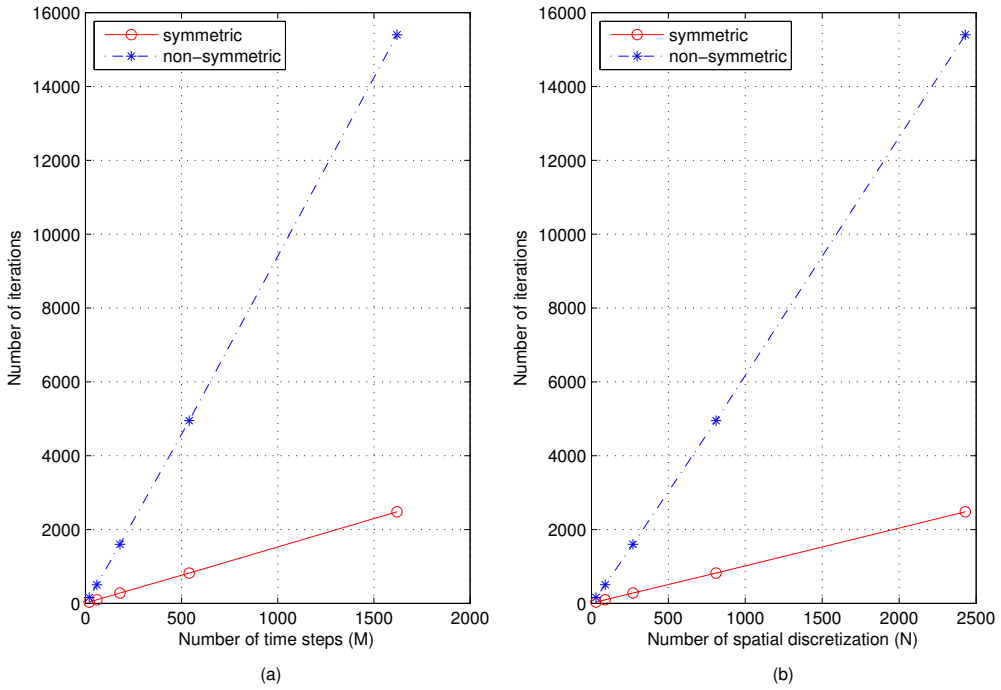

Figure 2. The effect of symmetrization: (**a**) number of time steps (M) versus total iterations; and (**b**) number of spatial discretization versus total iterations.

Computed solution with the iterative Crank–Nicolson method and its errors for three different options are shown in Figure 3. We can observe there is no oscillation in the computed solution with non-smooth payoffs. The result of numerical tests is summarized in Table 1. The ratio column shows the dropping rate of relative error for both methods. Note that the number of time steps required for OSM is significantly larger to maintain the same level of accuracy compared to the iterative Crank–Nicolson method. For a coarse grid, note that the time required to obtain the same level of accuracy for OSM is shorter than the iterative Crank–Nicolson. In the beginning, when the grid size is still coarse, the iterative Crank–Nicolson scheme takes more time to precondition the system than to actually solve it. However, the iterative Crank–Nicolson method reached the accuracy level faster than OSM as the grid becomes finer. Table 1 suggests that the iterative Crank–Nicolson method becomes more favorable in time if the accuracy is of concern.

We further compare both methods while keeping the $\Delta x/\Delta t$ ratio constant. The results are summarized in Table 2. We see that the iterative Crank–Nicolson method needs about half of the time that is required for OSM to reach the same level of relative error. However, the memory consumption of the iterative Crank–Nicolson is significantly larger than OSM.

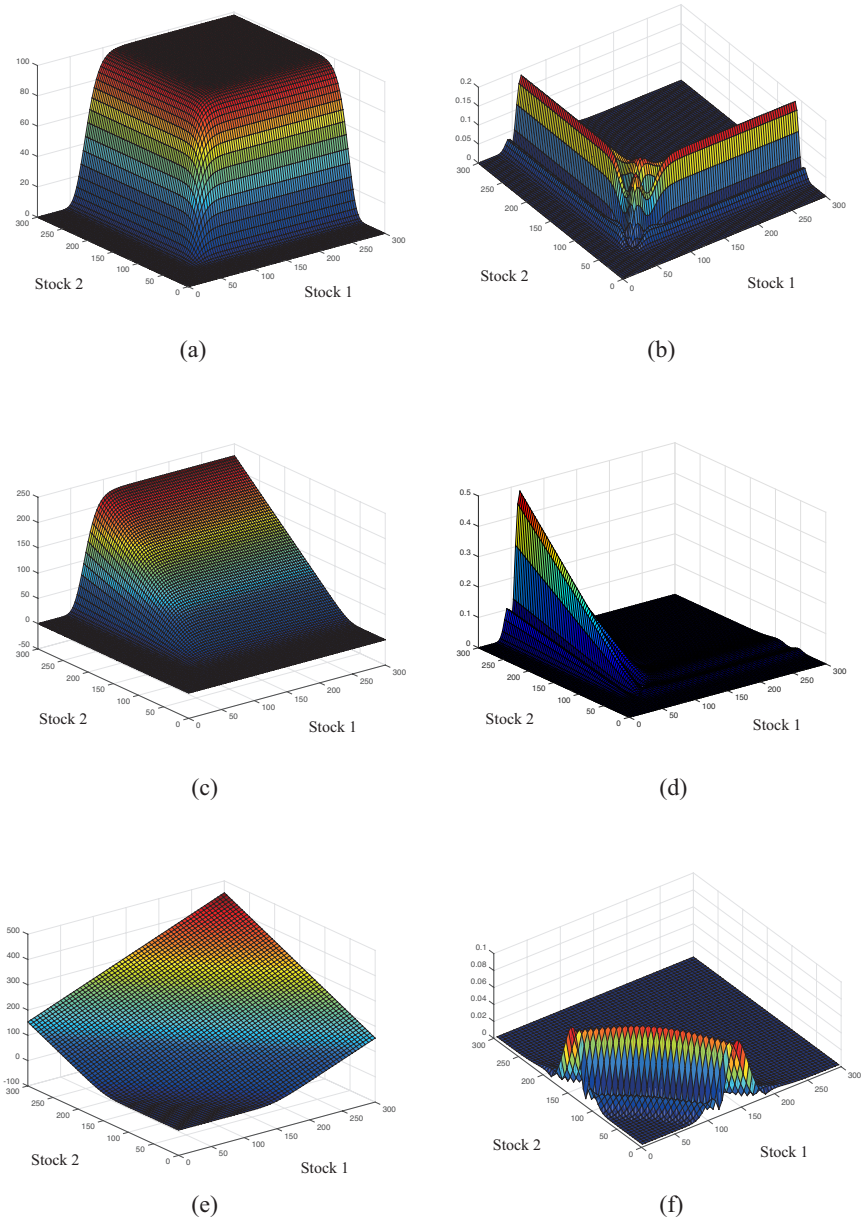

Figure 3. The iterative Crank–Nicolson solution and absolute error: (**a**) computed cash or nothing option values ($\Delta x = 1/90, \Delta t = 1/60$); (**b**) errors in cash or nothing options; (**c**) computed two asset call option values ($\Delta x = 1/90, \Delta t = 1/60$); (**d**) errors in two asset call options; (**e**) computed basket call option values ($\Delta x = 1/60, \Delta t = 1/60$); and (**f**) errors in basket call options.

Table 1. Computational time comparison between iterative Crank-Nicolson and Operator Splitting Method (OSM) to reach targeted accuracy while maintaining convergence rate.

Option Type	Numerical Method	Δx	Δt	Rel. Err (%)	Ratio	Time (s)
(1) Cash or Nothing	Iterative Crank–Nicolson	1/30	1/20	3.5	-	0.1
		1/90	1/60	0.34	10	1.5
		1/270	1/180	0.037	9.2	47
		1/810	1/540	0.0044	8.3	2000
	OSM	1/30	1/20	4.0	-	0.08
		1/90	1/180	0.39	10	0.8
		1/270	1/1620	0.042	9.3	5.4
		1/810	1/14580	0.0046	9.0	4200
(2) Call	Iterative Crank–Nicolson	1/30	1/20	1.6	-	0.1
		1/90	1/60	0.21	7.6	1.2
		1/270	1/180	0.024	8.8	43
		1/810	1/540	0.0026	9.4	1900
	OSM	1/30	1/20	1.9	-	0.03
		1/90	1/180	0.25	7.4	0.8
		1/270	1/1620	0.028	9.2	54
		1/810	1/14580	0.0030	9.0	4300
(3) Basket	Iterative Crank–Nicolson	1/30	1/30	0.71	-	0.1
		1/60	1/60	0.17	4.2	0.5
		1/120	1/120	0.042	4.0	4
		1/240	1/240	0.011	4.0	35
		1/480	1/480	0.0026	4.0	340
	OSM	1/30	1/30	0.73	-	0.03
		1/60	1/120	0.17	4.2	0.3
		1/120	1/480	0.043	4.0	3.6
		1/240	1/1920	0.011	4.0	6
		1/480	1/7680	0.0026	4.0	780

Table 2. CPU time, Memory, and Relative Error comparison with the use of identical Δx and Δt for both methods. The memory and CPU time are accumulated for the entire simulation.

Option Type	Numerical Method	Δx	Δt	Rel. Err. (%)	Memory (Mb)	Time (s)
(1) Cash or Nothing	Iterative Crank–Nicolson	1/30	1/20	1.68	18	0.8
		1/90	1/60	0.19	345	15
		1/270	1/180	0.02	8430	346
	OSM	1/30	1/20	2.15	5	1.3
		1/90	1/60	0.37	30	30
		1/270	1/180	0.09	252	771
(2) Call	Iterative Crank–Nicolson	1/30	1/20	1.6	18	0.8
		1/90	1/60	0.21	345	13
		1/270	1/180	0.02	8477	283
	OSM	1/30	1/20	1.9	4	1.2
		1/90	1/60	0.33	15	29
		1/270	1/180	0.06	159	756
(3) Basket	Iterative Crank–Nicolson	1/30	1/30	0.47	1.8	0.9
		1/60	1/60	0.12	24	5.8
		1/120	1/120	0.03	993	33
	OSM	1/30	1/30	0.49	2	1.6
		1/60	1/60	0.12	8.6	12
		1/120	1/120	0.04	45	96.9

5.2. Computational Cost

We compare the computational complexity of the iterative Crank–Nicolson method and implicit OSM. Throughout this section, we use N and M for the number of discretization steps for space and time, respectively. Both methods have an order of complexity $O(MN^2)$ and storage requirement $O(N^2)$, while the former has a second order convergence rate both in space and time and the latter has a second order convergence rate in space and a first order convergence rate in time.

Theorem 1. *The implicit OSM for the two-dimensional Black–Scholes equation has an order of computational complexity* $O(MN^2)$.

Proof. Suppose we partition the space and time domain with $(N-1)$ and $(M-1)$ intervals. For each time slice, we have to solve $2N$ (N times for each x- and y-direction) tridiagonal systems with the Thomas algorithm, which requires $O(N)$ computational cost. In other words, $2N^2$ operations are needed to solve the system of equations for a fixed time period, and we have a total of M time steps. As a consequence, the total computational cost to solve Equation (3) becomes $O(MN^2)$. □

With $O(MN^2)$ computational complexity, the implicit OSM achieves second order convergence in space but first order convergence in time. The computational demand for implicit OSM increases to $O(M^2N^2)$, if we increase the number of time steps to M^2 to obtain an overall second order convergence rate in error defined in Equation (25). Figure 4a–c shows the growth rate of computational complexity measured in time for implicit OSM, where M was chosen to be the same as N. The slope in Figure 4 supports the fact that implicit OSM, which has a second order convergence rate in terms of maximum norm, has computational complexity $O(M^2N^2) = O(N^4)$.

Theorem 2. *The iterative Crank–Nicolson method for the two-dimensional Black–Scholes equation has the computational complexity* $O(MN^2)$.

Proof. Let us assume that we partition the space and time domain with $(N-1)$ and $(M-1)$ intervals. Then, we have to solve the system given in Equation (16) of size N^2, for which the preconditioning and GMRES solver in Equations (21) and (22) require $O(N^2)$ computation for each time slice. Therefore, total computational cost to solve Equation (1), using the implicit Crank–Nicolson method is $O(MN^2)$. □

The growth rate shown in Figure 4d–f supports the iterative Crank–Nicolson method having computational complexity $O(MN^2) = O(N^3)$ when M was chosen to be the same as N.

Thus far, we have seen how the computational complexity grows for the iterative Crank–Nicolson method. Memory consumption is another factor that one should consider for the actual computation.

Theorem 3. *The memory requirement of the fully discretized Crank–Nicolson method is* $O(N^4)$ *for the two-dimensional Black–Scholes equation.*

Proof. The fully discretized Crank–Nicolson method, Equation (16), requires holding $\mathbf{P}_L$, $\mathbf{P}_R$, $\boldsymbol{\mathcal{L}}^{(n)}$, $\boldsymbol{\mathcal{R}}^{(n+1)}$ in Equations (21) and (22) for a fixed (n). The exact dimension of these matrices is $N^2 \times N^2$, requiring approximately N^4 amount of space when fully populated. □

However, $\mathbf{P}$, $\boldsymbol{\mathcal{L}}^{(n)}$, $\boldsymbol{\mathcal{R}}^{(n+1)}$ are all sparse matrices so that we can relax the storage requirement to N^2 by storing only the non-zeros. The actual number of non-zeros are on the order of N^2, not N^4. The ratio of non-zeros in the matrix, $\mathbf{P}_L$, $\mathbf{P}_R$, $\boldsymbol{\mathcal{L}}^{(n)}$, $\boldsymbol{\mathcal{R}}^{(n+1)}$, are similar. To see the growth of non-zeros in these matrices in terms of N, see, for example, Figure 5. The non-zero terms in sparse matrix (a) $\boldsymbol{\mathcal{L}}^{(n)}$, (b) $\boldsymbol{\mathcal{R}}^{(n+1)}$, and (c) $\mathbf{P}$ grow quadratically in number of grid points. The dotted line shown in Figure 5 has slope 2 and shows us that the actual storage requirement can be reduced to the order of N^2 from N^4.

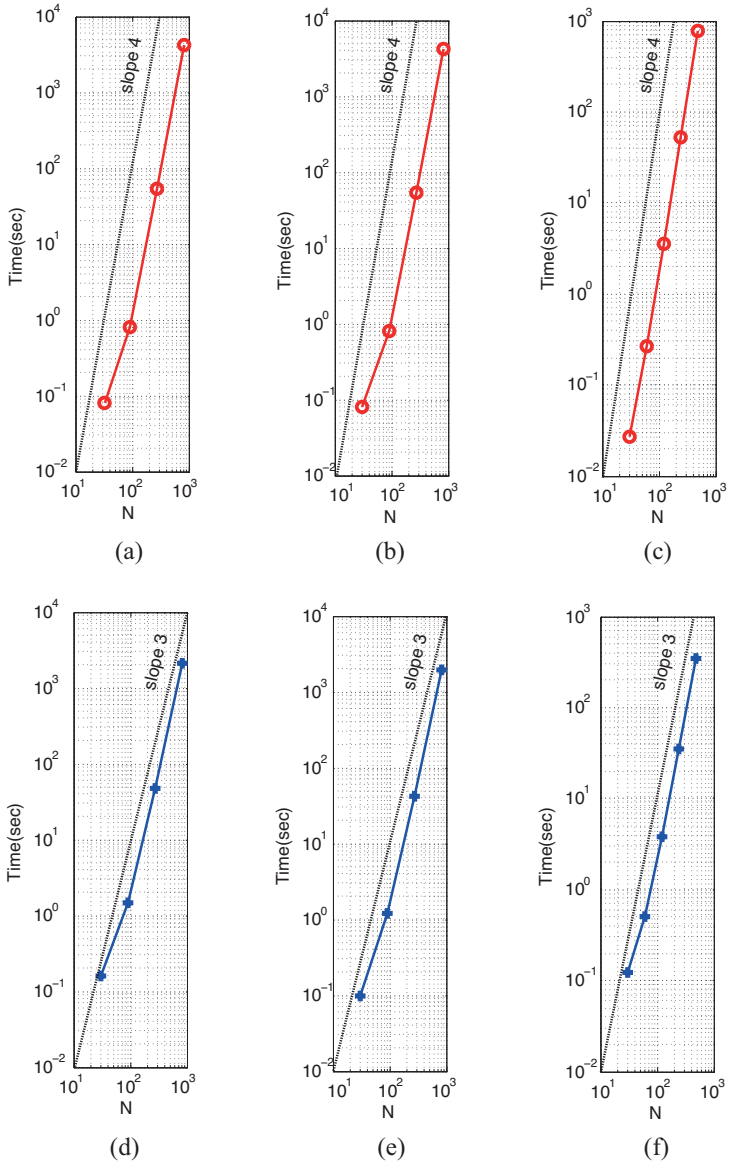

Figure 4. The order of complexity measured in time for implicit OSM with a different payoff: (**a**) cash or nothing; (**b**) two asset call; (**c**) basket call. The order of complexity measured in time for the iterative Crank–Nicolson method with a different payoff: (**d**) cash or nothing; (**e**) two asset call; and (**f**) basket call. The dotted line shows the growth rate.

Theorem 4. *The memory requirement of the implicit OSM is* $O(N^2)$ *for the two-dimensional Black–Scholes equation.*

Proof. To solve the tridiagonal system given in Equation (7), we only need $4N$ spaces in memory. However, to do this, we have to hold the data $\mathbf{v}^{(n+1)}$. Since the entire computational domain is a grid of size $N \times N$, N^2 amount of storage is needed in memory to hold $\mathbf{v}^{(n+1)}$ throughout the time period, Δt, to obtain $\mathbf{v}^{(n)}$ in Equation (7). Therefore, the amortized cost is on the order of N^2. □

Before closing this section, we summarize the computational complexity and memory requirement for the implicit OSM and iterative Crank–Nicolson method. The former is first order, and the latter is second order method in time. Both have the computational complexity of $O(MN^2)$. However, implicit OSM has $O(N^2)$ memory requirement and iterative Crank–Nicolson has $O(N^4)$ memory need for the two-dimensional Black–Scholes equation. Therefore, the growth of the data can not be compared for

large N but, after symmetrizing and using a sparse storage structure, we can reduce the storage need to $O\left(N^2\right)$ for the iterative Crank–Nicolson method, which is worth the effort.

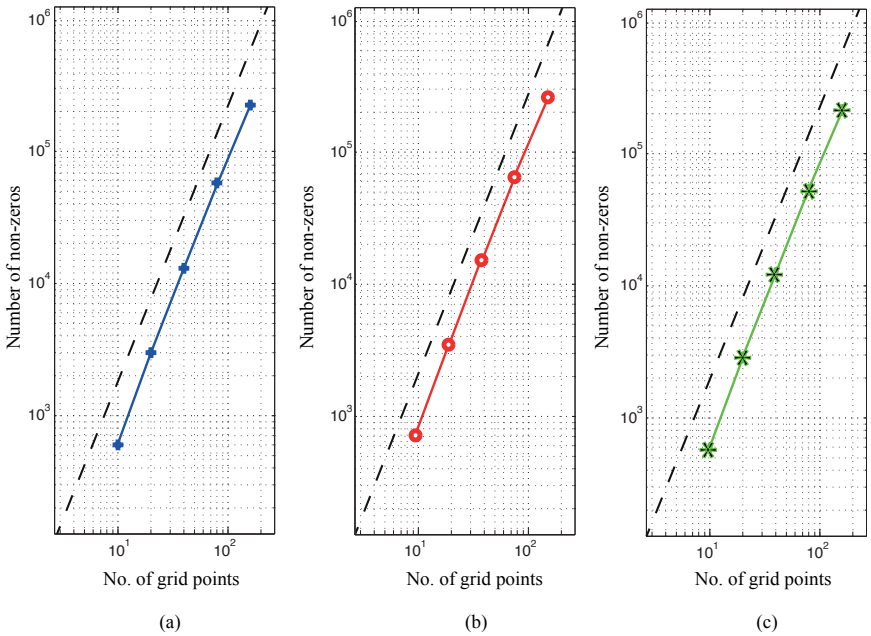

Figure 5. Number of non-zeros in the matrices, given in Equation (16), versus N, the number of grid points: (**a**) $\boldsymbol{\mathcal{L}}^{(n)}$; (**b**) $\boldsymbol{\mathcal{R}}^{(n+1)}$; and (**c**) $\mathbf{P}$. The quadratic growth rate is shown with the dotted line.

5.3. Order of Convergence in Space and Time

The numerical solution obtained by implicit OSM has a second order convergence rate in space and a first order convergence rate in time because the truncation error of the finite difference discretization given in Section 3 are all in second order and first order in time (see, for example, Figure 6a–c). The relative error in maximum norm for implicit OSM drops with slope one, which means it is first order in time.

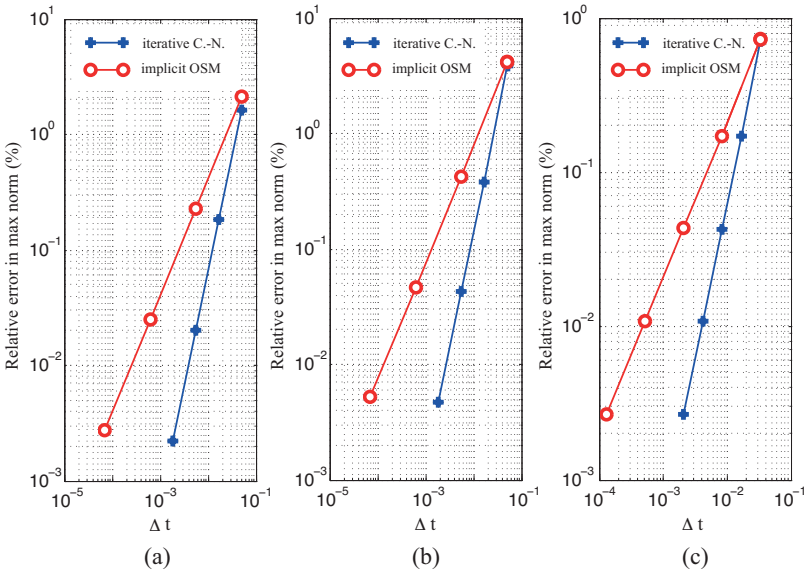

Figure 6. The relative error measured in maximum norm versus the time step size for: (**a**) cash or nothing option; (**b**) two asset call option; and (**c**) basket call option.

On the other hand, the iterative Crank–Nicolson method has a second order convergence rate both in space and time. The slope of the relative error curve, shown in Figure 6a–c, support the proposed method that has a second order convergence rate in time for all three options that we have considered while the implicit OSM converges first order in time. The second order of convergence can be proved by calculating the truncation error. The truncation error of iterative Crank–Nicolson discretization with non-uniform grid size is indeed second order in space and time (see Appendix in [19]).

6. Conclusions

A second order method is essential for pricing options when highly accurate option price is needed in time. The second order convergence even in the simplest case, a European style, is of great importance in practice when the option has complex payoff structures. For example, a multidimensional Black–Scholes equation has to be solved many times in a row to price and hedge an equity-linked security. A three-dimensional extension of the current study would be an interesting and important future work. A straightforward implementation of the multi-dimensional Crank–Nicolson scheme could be thought inefficient. However, with a simple symmetrization and a standard preconditioner, we have found that the order of complexity to solve the system is about the same for the fully discretized iterative Crank–Nicolson scheme and (*semi*-)implicit Operator Splitting Method. In other words, the order of complexity for the first and second order method turned out to be the same. However, note that the second order method, the iterative Crank–Nicolson method, needs more storage compared to the first order method, OSM, but reaches the same level of error in significantly less time. It is interesting to observe the trade-off between storage and computational time in the context of option pricing.

Acknowledgments: The authors would like to thank the anonymous reviewers for their valuable comments and suggestions to improve the quality of this paper. The research for this paper was financially supported by Gachon University, Grant No. GCU-2015-0177.

Author Contributions: Won-Tak Hong conceived and designed the study; Changyu Han implemented computer code and performed numerical analysis; Dohyun Pak analyzed the data and prepared the manuscript. All authors contributed to writing and revising the manuscript.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Daoud, Y.; Özis, T. The Operator Splitting Method for Black–Scholes Equation. *Appl. Math.* **2011**, *2*, 771–778.
2. Duffy, D.J. *Finite Difference Methods in Financial Engineering: A Partial Differential Approach*; John Wiley and Sons: New York, NY, USA, 2006.
3. Jeong, D.; Kim, J. A comparison study of ADI and operator splitting methods on option pricing models. *J. Comput. Appl. Math.* **2013**, *247*, 162–171.
4. Cavoretto, R. A numerical algorithm for multidimensional modeling of scattered data points. *Comput. Appl. Math.* **2015**, *34*, 65–80.
5. Shcherbakov, V.; Larsson, E. Radial basis function partition of unity methods for pricing vanilla basket options. *Comput. Math. Appl.* **2016**, *71*, 185–200.
6. Oosterlee, C.W.; Leentvaar, C.C.; Huang, X. *Accurate American Option Pricing by Grid Stretching and High Order Finite Differences*; Technical Report; Delft University of Technology: Delft, The Netherlands, 2005.
7. Le Floc'h, F. TR-BDF2 for Stable American Option Pricing. *J. Comput. Finance* **2014**, *17*, 31–56.
8. Hull, J.C. *Options, Futures and Other Derivatives*, 9th ed.; Prentice Hall: Upper Saddle River, NJ, USA, 2014.
9. Wilmott, P. *Paul Wilmott on Quantitative Finance*, 2nd ed.; Wiley: Hoboken, NJ, USA, 2006.
10. Zhu, Y.L.; Wu, X.; Chern, I. *Derivative Securities and Difference Methods*; Springer: Berlin, Germany, 2004.
11. Morton, K.W.; Mayers, D. *Numerical Solution of Partial Differential Equations*, 2nd ed.; Cambridge University Press: Cambridge, UK, 2005.
12. Saad, Y. *Iterative Methods for Sparse Linear Systems*, 2nd ed.; SIAM: Philadelphia, PA, USA, 2003.
13. Saad, Y.; Schultz, M.H. GMRES: A generalized minimal residual algorithm for solving nonsymmetric linear systems. *SIAM J. Sci. Stat. Comput.* **1986**, *7*, 856–869.

14. Mittal, R.; Al-Kurdi, A. An efficient method for constructing an ILU preconditioner for solving large sparse nonsymmetric linear systems by the GMRES method. *Comput. Math. Appl.* **2003**, *45*, 1757–1772.
15. Trefethen, L.N.; Iii, D.B. *Numerical Linear Algebra*; SIAM: Philadelphia, PA, USA, 1997.
16. Krekel, M.; de Kock, J.; Korn, R.; Man, T.K. *An Analysis of Pricing Methods for Baskets Options*; Wilmott: Soissons, France, 2004; pp. 82–89.
17. Haug, E.G. *The Complete Guide to Option Pricing Formulas*; McGraw-Hill Education: New York, NY, USA, 2006.
18. Ju, N. Pricing Asian and Basket options via taylor expansion. *J. Comput. Financ.* **2002**, *5*, 79–103, doi:10.21314/JCF.2002.088.
19. Han, C. Finite Difference Method with GMRES Solver for the 2D Black–Scholes Equation. Master's Thesis, Korea University, Seoul, Korea, 2014.