Computational Economics (2023) 61:1207–1224
https://doi.org/10.1007/s10614-022-10242-w

Accurate and Efficient Finite Difference Method for the Black–Scholes Model with No Far-Field Boundary Conditions

Chaeyoung Lee[1] · **Soobin Kwak**[1] · **Youngjin Hwang**[1] · **Junseok Kim**[1]

Accepted: 31 January 2022 / Published online: 17 February 2022

Abstract
A fast and accurate explicit finite difference scheme for the Black–Scholes (BS) model with no far-field boundary conditions is proposed. The BS equation has been used to model the pricing of European options. The proposed numerical solution algorithm does not require far-field boundary conditions. Instead, the computational domain is progressively reduced one by one as the time iteration increases. A Saul'yev-type scheme for temporal discretization and non-uniform grids for the underlying asset variables are used. Because the scheme is stable, practically sufficiently large time steps can be applied. The main advantages of the proposed method are its speed, simplicity, and efficiency because it uses a stable explicit numerical scheme without using far-field boundary conditions. In particular, the proposed method is suitable for nonlinear boundary profiles such as power options because it does not require far-field boundary conditions. To validate the speed and efficiency of the proposed scheme, standard computational tests are performed. The computational test results confirmed the superior performance of the proposed method.

Keywords Pricing · Option pricing · Explicit algorithm · Black–Scholes equation

✉ Junseok Kim
cfdkim@korea.ac.kr

Chaeyoung Lee
chae1228@korea.ac.kr

[1] Department of Mathematics, Korea University, Seoul 02841, Republic of Korea

1 Introduction

Options are popular financial instruments that can be applied to both investment decisions and risk management. An option contract is a derivatives contract between a holder and an underwriter, which grants its holder the right to buy (call option) or sell (put option) an underlying asset at a strike price at a fixed maturity date (Lyu et al., 2021; Sosa-Correa et al., 2018). One of the option pricing models is the Black–Scholes (BS) equation (Black and Scholes 1973). Let $u(S, t)$ be the option value of the underlying asset S at time t. In this study, a fast and stable explicit numerical scheme is presented for the following BS equation without far-field boundary conditions:

$$\frac{\partial u(S,t)}{\partial t} = -\frac{1}{2}\sigma^2 S^2 \frac{\partial^2 u(S,t)}{\partial S^2} - rS\frac{\partial u(S,t)}{\partial S} + ru(S,t), \text{ for } S > 0, 0 \le t < T \qquad (1)$$

with the payoff function $u(S, T)$. Here, σ is the volatility, r is the interest rate, and T is the expiry date (Black and Scholes 1973). For example, for a standard European call option, the payoff function is given by $u(S,T) = \max(S-K, 0)$, where K is the strike price. A European call option is a financial contract between the buyer and seller. The buyer of the European call option has the right to buy an agreed quantity of the underlying asset from the seller of the option at the expiration date T for the strike price K. The buyer pays a premium $u(S, 0)$ at the current time for this right, and the value $u(S, 0)$ can be obtained by solving the BS Eq. (1).

The classical BS equation can be derived from the stochastic differential equations (Sauer, 2013). In general, the option value u depends on the strike price K, volatility σ, interest rate r, current underlying asset price S, and expiry time T. In this study, for simplicity, K, σ, r, and T are assumed as the given constant parameters. Note that our proposed numerical method can be applied to generalized BS equations with general volatility $\sigma(S, t)$ and interest rate $r(S, t)$.

Several studies have solved the BS equation using the finite difference method (FDM) (Abdi-Mazraeh et al., 2020; Koffi & Tambue, 2020; Golbabai et al., 2019). When solving Eq. (1) using the FDM, the solution is approximated on a truncated domain and the appropriate boundary conditions are necessary (Anwar & Andallah, 2018; Roul & Goura, 2020, 2020). For example, Dirichlet, linear, and partial differential equation (PDE) boundary conditions were used (Jeong et al. 2015; Koleva and Vulkov 2017; Roul and Goura 2020). Kangro and Nicolaides derived pointwise error bounds caused by the boundary conditions, and the error estimates can be applied to compute a priori a suitable position for the far-field boundary condition (Kangro and Nicolaides 2000). The linear boundary condition is one of the most frequently used boundary conditions and assumes that the second derivative of the derivative security value is zero. Rigorous studies on the stability of the linear boundary condition of the FDM for the BS equation have been conducted (Windcliff et al., 2004). Choi et al. (2015) developed a hybrid method using a payoff-consistent extrapolation based on the payoff function. In addition, Jeong et al. (2019) proposed a Monte Carlo boundary condition that applies a far-field boundary value computed from a Monte Carlo simulation (MCS).

However, if the payoff function is nonlinear, it is difficult to apply the above boundary conditions. A computational scheme was recently presented for the BS model without boundary conditions (Jeong et al., 2018). The explicit scheme demands a strict condition for stability (Kim & Lee, 2018), where the authors used an explicit Euler's method with a time-step constraint. Because of the time-step constraint, sufficiently small time steps have to be used for the stability of the method.

The main purpose of this study is to propose a practically stable numerical method for the BS equation without far-field boundary conditions based on a Saul'yev-type scheme to overcome the time-step restriction (Saul'Yev, 1964). It is simple to implement the governing equation because the proposed method is an explicit scheme. Furthermore, relatively large time steps can be used because of the stable numerical scheme, which alleviates the strict time-step constraint of the existing explicit scheme. A fast and stable explicit numerical method are also presented for two-dimensional BS equations.

The remainder of this paper is organized as follows. Section 2 gives the proposed computational method without far-field boundary conditions. Section 3 presents the results of the computational experiments. Section 5 provides concluding remarks. In the Appendix, the computer program codes are given for interested readers.

2 Numerical Solution

In this section, a detailed description of the numerical method is provided to find the approximate solutions of the BS equation with one or two underlying assets.

2.1 One-Dimensional Black–Scholes Equation

Let $\tau = T - t$ and $x = S$, Eq. (1) then becomes

$$\frac{\partial u}{\partial \tau} = \frac{1}{2}(\sigma x)^2 \frac{\partial^2 u}{\partial x^2} + rx\frac{\partial u}{\partial x} - ru, \text{ for } x > 0, 0 < \tau \leq T \tag{2}$$

with $u(x, 0)$ (Dubey et al. 2019; Farnoosh et al. 2016). The BS model is discretized on a non-uniform grid (Hanh and Thanh 2019; Lyu et al. 2021; Soleymani and Akgül 2019). The non-uniform mesh consists of $x_{i+1} = x_i + h_i$ for $i = 0, \ldots, N_x - 1$, where $x_0 = 0$ and h_i is the non-uniform step size and N_x is a positive integer, as shown in Fig. 1.

Let u_i^n be an approximation of the analytic solution $u(x_i, n\Delta\tau)$. Here, $\Delta\tau = T/N_\tau$ is a temporal step and N_τ is a positive integer. The Saul'yev-type explicit method (Saul'Yev, 1964) is applied to Eq. (2):

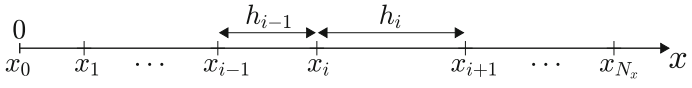

Fig. 1 Schematic of the computational discrete domain

$$\begin{aligned}\frac{u_i^{n+1}-u_i^n}{\Delta\tau} =& \sigma^2 x_i^2\left(\frac{(u_{i+1}^n-u_i^n)}{(h_{i-1}+h_i)h_i}-\frac{(u_i^{n+1}-u_{i-1}^{n+1})}{(h_{i-1}+h_i)h_{i-1}}\right)\\ &+rx_i\left(\frac{h_{i-1}(u_{i+1}^n-u_i^n)}{(h_{i-1}+h_i)h_i}+\frac{h_i(u_i^{n+1}-u_{i-1}^{n+1})}{h_{i-1}(h_{i-1}+h_i)}\right)-\frac{r}{2}(u_i^n+u_i^{n+1}).\end{aligned} \tag{3}$$

Eq. (3) can be rewritten in the explicit form as follows:

$$\begin{aligned}u_i^{n+1} =& \left[\frac{u_i^n}{\Delta\tau}+\sigma^2 x_i^2\left(\frac{(u_{i+1}^n-u_i^n)}{h_i(h_{i-1}+h_i)}+\frac{u_{i-1}^{n+1}}{h_{i-1}(h_{i-1}+h_i)}\right)\right.\\ &\left.+rx_i\left(\frac{h_{i-1}(u_{i+1}^n-u_i^n)}{h_i(h_{i-1}+h_i)}-\frac{h_i u_{i-1}^{n+1}}{h_{i-1}(h_{i-1}+h_i)}\right)-\frac{r}{2}u_i^n\right]\Big/ fac,\\ &\text{where } fac=\frac{1}{\Delta\tau}+\frac{\sigma^2x_i^2-rx_ih_i}{h_{i-1}(h_{i-1}+h_i)}+\frac{r}{2}.\end{aligned} \tag{4}$$

A technique is adopted in which one grid point is dropped from the far-field boundary point after one time step is updated. In option pricing problems, people are interested in the solutions neighboring the current underlying asset price to compute the option price and its Greeks. For example, let x_j be the current underlying asset price for some j. Assume that our goal is to compute $u_{j-1}^{N_\tau}$, $u_j^{N_\tau}$, and $u_{j+1}^{N_\tau}$. This is then defined as $N_x = N_\tau + j + 1$; thus, the numerical solutions after N_τ time step iterations, $u_{j-1}^{N_\tau}$, $u_j^{N_\tau}$, and $u_{j+1}^{N_\tau}$, can be obtained. For European call option pricing, the main algorithm is as follows: The left-end boundary point value is fixed as $u_0^n = 0$ for all n, and Eq. (4) is solved for $n = 0, \ldots, N_\tau - 1$ by dropping the far-field boundary grid point after each time step update. That is, in the first time-step update, Eq. (4) is solved for $i = 1, \ldots, N_x - 1$. In the second step, Eq. (4) is solved on $i = 1, \ldots, N_x - 2$. In the last time step, Eq. (4) is solved on $i = 1, \ldots, j + 1$. Therefore, the far-field boundary condition is not required because the mesh points are reduced one by one (Jeong et al. 2018). Figure 2 displays a graphical representation of the reducing grids.

Compared to the previous algorithm (Jeong et al., 2018), the proposed algorithm is much simpler and faster because the new method uses a stable finite difference scheme that allows larger time steps.

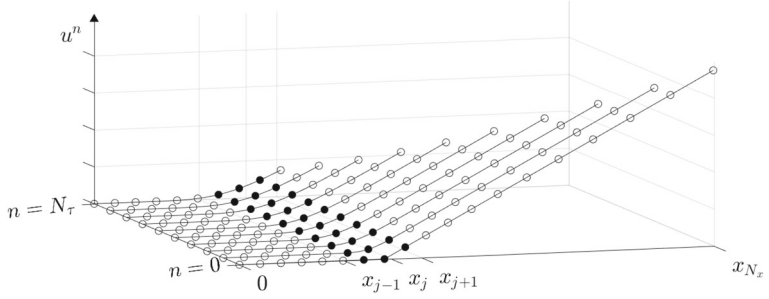

Fig. 2 Schematic illustration of the reducing grids

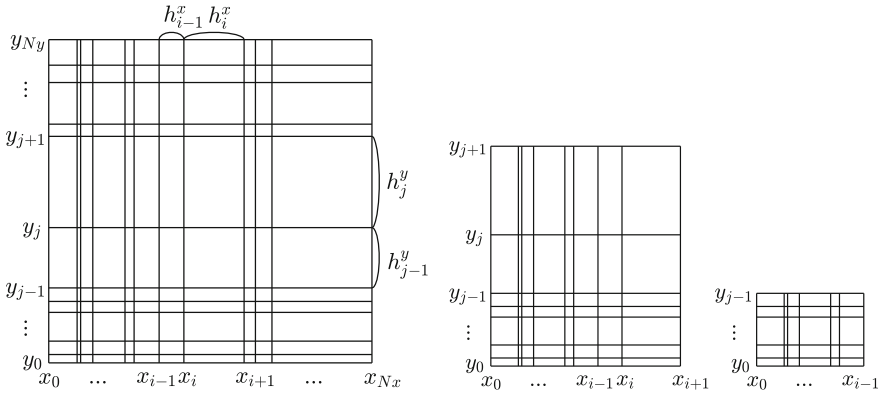

Fig. 3 Schematic of 2D computational discrete domains

2.2 Two-dimensional Black–Scholes Equation

Let us consider a two-dimensional (2D) version of the BS equation (Khodayari & Ranjbar, 2019):

$$\frac{\partial u}{\partial \tau} = \frac{1}{2}\sigma_x^2 x^2 \frac{\partial^2 u}{\partial x^2} + \rho\sigma_x\sigma_y xy \frac{\partial^2 u}{\partial xy} + \frac{1}{2}\sigma_y^2 y^2 \frac{\partial^2 u}{\partial y^2} + rx\frac{\partial u}{\partial x} + ry\frac{\partial u}{\partial y} - ru, \quad \text{for } x, y > 0, 0 < \tau \leq T \tag{5}$$

with an initial condition $u(x, y, 0)$ (Dubey et al., 2019; Farnoosh et al., 2016). The 2D BS model is discretized on a non-uniform grid (Hanh & Thanh, 2019; Lyu et al., 2021; Soleymani & Akgül, 2019). The non-uniform mesh consists of $x_{i+1} = x_i + h_i^x$ and $y_{j+1} = y_j + h_j^y$ for $i = 0, \ldots, N_x - 1$, $j = 0, \ldots, N_y - 1$, where $x_0 = 0, y_0 = 0$ and h_i^x, h_j^y are the non-uniform step sizes and N_x, N_y are positive integers, as shown in Fig. 3.

Let u_{ij}^n be an approximation of the analytic solution $u(x_i, y_j, n\Delta\tau)$. The Saul'yev-type explicit method (Saul'Yev, 1964) is applied to Eq. (5):

$$
\begin{aligned}
\frac{u_{ij}^{n+1}-u_{ij}^{n}}{\Delta\tau} =& \sigma_x^2 x_i^2 \left(\frac{(u_{i+1,j}^{n}-u_{ij}^{n})}{(h_{i-1}^{x}+h_i^{x})h_i^{x}} - \frac{(u_{ij}^{n+1}-u_{i-1,j}^{n+1})}{(h_{i-1}^{x}+h_i^{x})h_{i-1}^{x}} \right) \\
&+ \rho\sigma_x\sigma_y x_i y_j \left(\frac{u_{i+1,j+1}^{n}-u_{i+1,j}^{n}-u_{i,j+1}^{n}+u_{ij}^{n}}{h_i^{x}h_j^{y}} \right) \\
&+ \sigma_y^2 y_j^2 \left(\frac{(u_{i,j+1}^{n}-u_{ij}^{n})}{(h_{j-1}^{y}+h_j^{y})h_j^{y}} - \frac{(u_{ij}^{n+1}-u_{i,j-1}^{n+1})}{(h_{j-1}^{y}+h_j^{y})h_{j-1}^{y}} \right) \\
&+ r x_i \left(\frac{h_{i-1}^{x}(u_{i+1,j}^{n}-u_{ij}^{n})}{(h_{i-1}^{x}+h_i^{x})h_i^{x}} + \frac{h_i^{x}(u_{ij}^{n+1}-u_{i-1,j}^{n+1})}{h_{i-1}^{x}(h_{i-1}^{x}+h_i^{x})} \right) \\
&+ r y_j \left(\frac{h_{j-1}^{y}(u_{i,j+1}^{n}-u_{ij}^{n})}{(h_{j-1}^{y}+h_j^{y})h_j^{y}} + \frac{h_j^{y}(u_{ij}^{n+1}-u_{i,j-1}^{n+1})}{h_{j-1}^{y}(h_{j-1}^{y}+h_j^{y})} \right) \\
&- \frac{r}{2}(u_{ij}^{n}+u_{ij}^{n+1}).
\end{aligned}
\tag{6}
$$

Eq. (6) can be rewritten in the explicit form as follows:

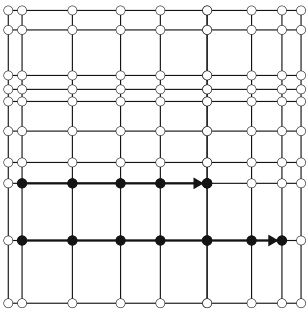

Fig. 4 Schematic of the ordering used in updating the numerical solutions

$$
\begin{aligned}
u_{ij}^{n+1} = \Bigg[& \frac{u_{ij}^n}{\Delta\tau} + \sigma_x^2 x_i^2 \left(\frac{(u_{i+1,j}^n - u_{ij}^n)}{h_i^x (h_{i-1}^x + h_i^x)} + \frac{u_{i-1,j}^{n+1}}{h_{i-1}^x (h_{i-1}^x + h_i^x)} \right) \\
& + \rho \sigma_x \sigma_y x_i y_j \left(\frac{u_{i+1,j+1}^n - u_{i+1,j}^n - u_{i,j+1}^n + u_{ij}^n}{h_i^x h_j^y} \right) \\
& + \sigma_y^2 y_j^2 \left(\frac{(u_{i,j+1}^n - u_{ij}^n)}{h_j^y (h_{j-1}^y + h_j^y)} + \frac{u_{i,j-1}^{n+1}}{h_{j-1}^y (h_{j-1}^y + h_j^y)} \right) \\
& + r x_i \left(\frac{h_{i-1}^x (u_{i+1,j}^n - u_{ij}^n)}{h_i^x (h_{i-1}^x + h_i^x)} - \frac{h_i^x u_{i-1,j}^{n+1}}{h_{i-1}^x (h_{i-1}^x + h_i^x)} \right) \\
& + r y_j \left(\frac{h_{j-1}^y (u_{i,j+1}^n - u_{ij}^n)}{h_j^y (h_{j-1}^y + h_j^y)} - \frac{h_j^y u_{i,j-1}^{n+1}}{h_{j-1}^y (h_{j-1}^y + h_j^y)} \right) - \frac{r}{2} u_{ij}^n \Bigg] \Bigg/ fac, \\
\text{where } fac = & \frac{1}{\Delta\tau} + \frac{\sigma_x^2 x_i^2 - r x_i h_i^x}{h_{i-1}^x (h_{i-1}^x + h_i^x)} + \frac{\sigma_y^2 y_j^2 - r y_j h_j^y}{h_{j-1}^y (h_{j-1}^y + h_j^y)} + \frac{r}{2}.
\end{aligned}
\tag{7}
$$

Eq. (7) can be calculated inside two nested for-loops, as shown in Fig. 4 for the direction of the loop.

Table 1 Relative errors of the one-asset European call option price at $x = 100$ with $r = 0.03$, $\sigma = 0.3$, and $T = 0.1$

N_τ	200	400	800	1600	3200
$h = 2$	5.5666e-03	5.4658e-03	5.4185e-03	5.3956e-03	5.3843e-03
$h = 1$	1.7604e-03	1.5308e-03	1.4286e-03	1.3806e-03	1.3574e-03
$h = 0.5$	1.3221e-03	7.6034e-04	5.2994e-04	4.2735e-04	3.7920e-04

The exact value is 3.929276040140451

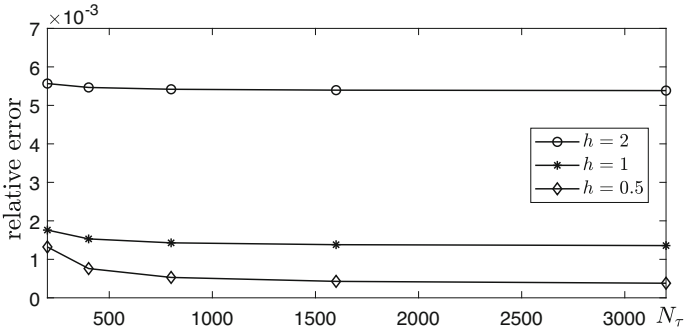

Fig. 5 Plot of relative errors with respect to the number of time steps (N_τ). Here, $h = 2, 1$, and 0.5 are used

Table 2 Relative errors and convergence rates at $x = 100$ with respect to the time-step sizes

N_τ	400	Rates	800	Rates	1600
Relative errors	2.0919e-03	1.07	9.9837e-04	0.97	5.1090e-04

Table 3 Relative errors and convergence rates at $x = 100$ with respect to the mesh grid sizes

h	2	Rates	1	Rates	0.5
Relative errors	2.8645e-02	2.05	6.9118e-03	1.98	1.7494e-03

3 Numerical Tests

To confirm the efficiency of the proposed method, computational tests, such as on the pricing of European call and powered options, are performed. All tests are performed on an Intel Core i7-10700 CPU at 2.90 GHz with 16 GB RAM using MATLAB R2020b software.

3.1 One-Asset Options

3.1.1 European Call Option

Let us consider a European call option as the first test. Let $u(x,0) = \max(x - K, 0)$ be the initial condition with a strike price of $K = 100$. Let us define the uniform grids with mesh grid sizes of $h = 1, 0.5, 0.25, 0.125$, that is, $\Omega_h = \{x_i | x_i = hi, \text{ for } i = 0, \ldots, 100/h + N_\tau\}$. Therefore, after N_τ time-step iterations, the numerical solution at $x = 100$ can be obtained. The time-step size $\Delta\tau = T/N_\tau$, $r = 0.03$, $\sigma = 0.3$, and $T = 0.1$ are used. The analytic solution of the European call option (Feng et al., 2020) is

$$u(x,\tau) = xN(d_1) - Ke^{-r\tau}N(d_2),$$
$$d_1 = \left(\ln(x/K) + \left(r + 0.5\sigma^2\right)\tau\right)/(\sigma\sqrt{\tau}), d_2 = d_1 - \sigma\sqrt{\tau},$$

where $N(d)$ denotes the cumulative distribution function (Black & Scholes, 1973; Ma et al., 2019):

$$N(d) = \frac{1}{\sqrt{2\pi}}\int_{-\infty}^{d} \exp(-0.5x^2)\mathrm{d}x.$$

Table 1 lists the relative errors of the European call option price at $x = 100$ with $r = 0.03$, $\sigma = 0.3$, and $T = 0.1$. Here, the relative error is defined as $|u(100,T) - u_{100/h}^{N_\tau}|/u(100,T)$.

Table 4 Relative errors of the powered option price at $x = 100$ with $r = 0.03$, $\sigma = 0.3$, and $T = 0.1$. The exact value is 51.08399700557311

N_τ	200	400	800	1600	3200
$h = 2$	1.1540e-02	5.8411e-03	2.9855e-03	1.5562e-03	8.4120e-04
$h = 1$	2.2865e-02	1.1480e-02	5.7633e-03	2.8991e-03	1.4656e-03
$h = 0.5$	4.5453e-02	2.2858e-02	1.1465e-02	5.7442e-03	2.8780e-03

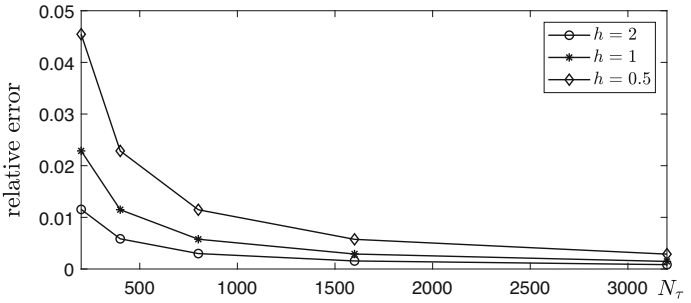

Fig. 6 Convergence of the relative errors of the computational results for a powered option with respect to the number of time steps N_τ. Here, $h = 2, 1$, and 0.5 are used

Figure 5 displays the relative errors of the computational results with various spatial and time step sizes. It can be seen that the convergence of the option prices as the time step size, $\Delta\tau = T/N_\tau$, is refined.

Convergence tests for the proposed scheme are conducted with respect to N_τ and h. The parameters not mentioned are set to the same values as in the previous test. Note that the relative error when using h and N_τ is called $e_h^{N_\tau}$. First, let us consider a test with a progressively smaller time-step size of $\Delta\tau$. The parameters $h = 1$ and $T = 0.5$ are fixed on the domain $\Omega_h = (0, 900.5)$. The convergence rate according to the time-step size is defined as $\log_2\left(e_h^{N_\tau}/e_h^{2N_\tau}\right)$. Table 2 lists the relative errors and convergence rates at $x = 100$ with increasingly larger N_τ. As expected from the temporal discretization of the BS equation, the first-order convergence rate is observed.

Next, a convergence test is conducted as the mesh grid size h becomes finer. The parameters $N_\tau = 800$ and $T = 0.02$ are fixed on the domain $\Omega_h = (0, 1702)$. The convergence rate according to the mesh grid size is defined as $\log_2\left(e_h^{N_\tau}/e_{h/2}^{N_\tau}\right)$. Table 3 lists the relative errors and convergence rates at $x = 100$ with an increasingly smaller h. It is seen that the second-order convergence rate of the underlying asset variable.

3.1.2 Powered Option

Next, let us consider the pricing of a powered option. The payoff is defined as $u(x, 0) = \max(x - K, 0)^p$, where p is the power (Haug, 2007). Here, $p = 2$ and

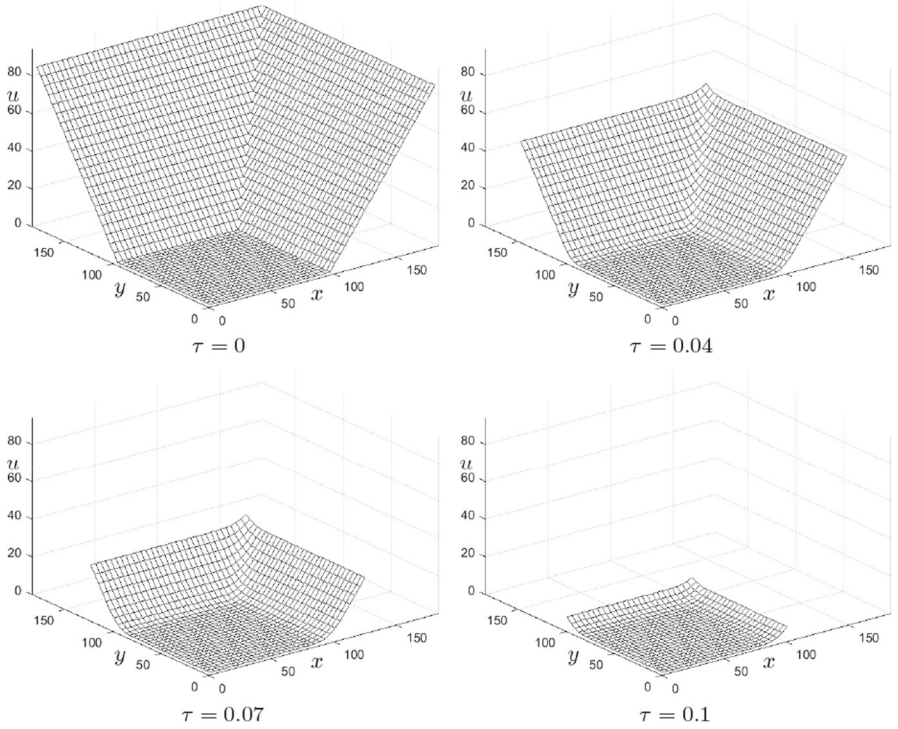

Fig. 7 Temporal evolution of numerical solutions and computational grids

Table 5 Relative errors in the two-asset European call option price at $x = 100$ and $y = 100$ with $r = 0.015$, $\sigma_x = 0.3$, $\sigma_y = 0.3$, $\rho = 0.3$, and $T = 0.1$. The exact value is 6.191151814151041

N_τ	100	200	400	800	1600
$h = 2$	1.3034e−02	1.2750e−02	1.2615e−02	1.2550e−02	1.2518e−02
$h = 1$	5.9192e−03	5.3653e−03	5.1199e−03	5.0051e−03	4.9497e−03
$h = 0.5$	4.2623e−03	2.9885e−03	2.4797e−03	2.2572e−03	2.1540e−03

$K = 100$ are used. Let us define the uniform grids with mesh grid sizes $h = 1, 0.5, 0.25$, that is, $\Omega_h = \{x_i | x_i = hi, \text{ for } i = 0, \ldots, 100/h + N_\tau\}$. Therefore, after N_τ time-step iterations, the numerical solution can be obtained at $x = 100$. The time-step size is $\Delta\tau = T/N_\tau$. Table 4 lists the relative errors of the powered option prices. The analytic solution of the powered option is

$$u(x,\tau) = \sum_{q=0}^{p} \frac{p!}{q!(p-q)!} x^{p-q}(-K)^q e^{(p-q-1)(r+0.5(p-q)\sigma^2)\tau} N(d_{p,q}), \tag{8}$$

where $d_{p,q} = [\ln(x/K) + (r + (p - q - 0.5)\sigma^2)\tau]/(\sigma\sqrt{\tau})$ (Zhang, 1998). From the table, it is observed that the relative errors converge as the time steps are reduced.

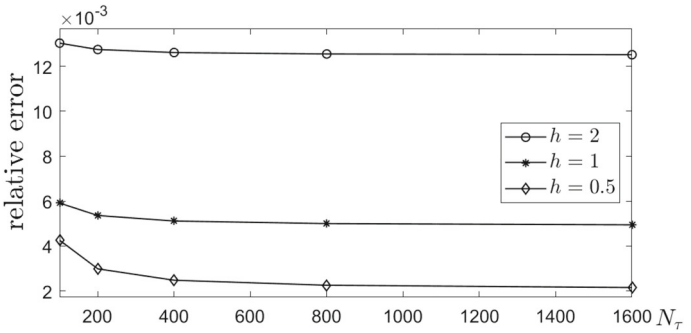

Fig. 8 Plot of the relative errors with respect to the number of time steps, N_τ. Here, $h = 2, 1$, and 0.5 are used

Figure 5 displays the convergence of the relative errors of the computational results with various time-step sizes.

3.2 Two-Asset European Call Option

Let $u(x, y, 0) = \max[\max(x, y) - K, 0]$ be the initial condition with a strike price of K. Figures 7(a)–(d) show the option prices at $\tau = 0, 0.04, 0.07$, and 0.1, respectively. Here, the following parameters are used: $\sigma_x = \sigma_y = 0.3, r = 0.015, \rho = 0.3$, $K = 100, d\tau = 5e\text{-}3$, and $h = h^x = h^y = 4$ on the computational domain $\Omega_h = (0, 184)^2$.

Let us define the uniform grids $\Omega_{h^x,h^y} = \Omega_{h^x} \times \Omega_{h^y}$ with mesh grid sizes $h = 2, 1, 0.5$, where $\Omega_{h^x} = \{x_i | x_i = hi, \text{ for } i = 0, \ldots, 100/h + N_\tau\}$ and $\Omega_{h^y} = \{y_i | y_i = hi, \text{ for } i = 0, \ldots, 100/h + N_\tau\}$. Therefore, after N_τ time-step iterations, the numerical solution at $x = 100$ and $y = 100$ can be obtained. A time-step size of $\Delta\tau = T/N_\tau$, $r = 0.015$, $\sigma_x = 0.3$, $\sigma_y = 0.3$, $\rho = 0.3$, and $T = 0.1$ are used. The analytic solution of the European call option (Haug, 2007; Heo et al., 2019) is as follows:

$$\begin{aligned} u(x, y, T) =& xM(d_1, d; \rho_1) + yM(d_2, -d + \sigma\sqrt{T}; \rho_2) \\ & - Xe^{-rT}[1 - M(-d_1 + \sigma_1\sqrt{T}, -d_2 + \sigma_2\sqrt{T}; \rho)], \end{aligned}$$

where $d = \frac{\ln(x/y) + 0.5\sigma^2 T}{\sigma\sqrt{T}}$, $d_1 = \frac{\ln(x/X) + (r + 0.5\sigma_1^2)T}{\sigma_1\sqrt{T}}$, $d_2 = \frac{\ln(y/X) + (r + 0.5\sigma_2^2)T}{\sigma_2\sqrt{T}}$, $\sigma = \sqrt{\sigma_1^2 + \sigma_2^2 - 2\rho\sigma_1\sigma_2}$, $\rho_1 = (\sigma_1 - \rho\sigma_2)/\sigma$, $\rho_2 = (\sigma_2 - \rho\sigma_1)/\sigma$, and $M(m, n; \rho)$ is the standard cumulative normal distribution function:

$$M(m, n; \rho) = \frac{1}{2\pi\sqrt{1 - \rho^2}} \int_{-\infty}^{m} \int_{-\infty}^{n} e^{-\frac{\xi^2 - 2\rho\xi\eta + \eta^2}{2(1-\rho^2)}} d\xi \, d\eta.$$

Table 5 lists the relative errors of the two-asset European call option price at $x = 100$ and $y = 100$ with $K = 100$, $r = 0.015$, $\sigma_x = 0.3$, $\sigma_y = 0.3$, $\rho = 0.3$, and $T = 0.1$. Here, the relative error is defined as follows:

Table 6 Comparison between the previous study and the proposed scheme

Scheme	Option price	Relative error	N_τ	CPU time (s)
Heo et al. (2019)	5.88205486745001	8.2320e−3	1489	98.5294
Proposed	5.87934843996972	8.6883e−3	360	2.5118

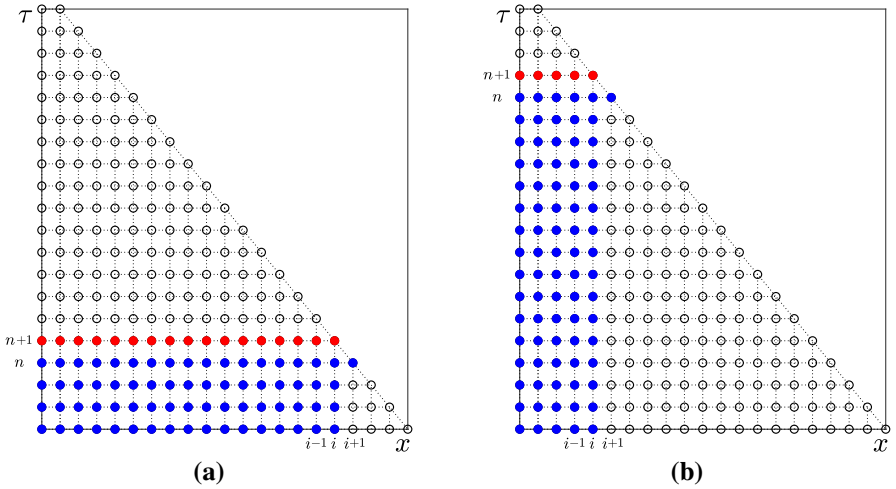

Fig. 9 Schematic diagram of the memory allocation used to find the solution of $u(x,\tau)$ at time $\tau = (n+1)\Delta\tau$ when applying the proposed method: **a** early and **b** later times

$$R_{err} = \left| u(100,T) - u^{N_\tau}_{100/h} \right| \Big/ u(100,T).$$

Figure 8 displays the relative errors shown in Table 5. From the results in the table and figure, it is confirmed that the relative errors converge as the time and space step sizes decrease.

3.3 Comparison Test

To verify the computational efficiency, our proposed scheme is compared with that of previous study (Heo et al. 2019). For the test, a two-asset European call option price is considered at $x = 100$ and $y = 100$ with $r = 0.03$, $\sigma_x = 0.3$, $\sigma_y = 0.3$, $\rho = 0.5$, $h = 1$, and $T = 0.1$. Here, the exact value is 5.93087774227051. Table 6 lists the numerical option prices, relative errors, N_τ, and CPU times. Here, the CPU times are averaged over 10 trials and represented in seconds. It is observed that the proposed scheme is approximately 40 times faster than the previous scheme with a similar accuracy.

The previous method has a severe time-step constraint, which results in a costly computation. By contrast, our proposed scheme requires relatively large time step sizes of $\Delta\tau$, which greatly reduces the computational costs. In financial markets, it is

significantly important to compute option prices quickly; hence, this approach may attract the attention of researchers and financial engineers.

4 Discussion

Owing to the outstanding memory effect of fractional derivatives, fractional order differential equations have been widely applied to better model the dynamical phenomena (Nikan and Avazzadeh 2021; Nikan et al. 2021). For this reason, the following time-fractional BS equation can be used to model the pricing of options with memory effect (Golbabai & Nikan, 2020):

$$\frac{\partial^{\alpha} u}{\partial \tau^{\alpha}} = \frac{1}{2}(\sigma x)^2 \frac{\partial^2 u}{\partial x^2} + rx\frac{\partial u}{\partial x} - ru, \text{ for } x > 0, 0 < \tau \leq T, \tag{9}$$

where $0 < \alpha < 1$ and the time-fractional term is defined as

$$\frac{\partial^{\alpha} u}{\partial \tau^{\alpha}}(x,\tau) = \frac{1}{\Gamma(1-\alpha)}\frac{d}{d\tau}\int_0^{\tau} \frac{u(x,T-\eta)-u(x,T)}{(\tau-\eta)^{\alpha}} d\eta. \tag{10}$$

here $\Gamma(z) = \int_0^{\infty} t^{z-1}e^{-t}dt$ is the gamma function. Equation (10) can be discretized as follows (Huang et al. 2019):

$$\frac{1}{(\Delta\tau)^{\alpha}\Gamma(2-\alpha)}\sum_{p=1}^{n+1}[(n+2-p)^{1-\alpha} - (n+1-p)^{1-\alpha}]\left(u_i^p - u_i^{p-1}\right). \tag{11}$$

Let us consider an application of the proposed method to give an economic rationale for the use of fractional derivatives. Figure 9a and b show schematic diagrams of the memory allocation and computational nodes to find the solution of $u(x,\tau)$ at time $\tau = (n+1)\Delta\tau$ using the proposed method at early and later times, respectively. The blue dots are nodes needed to compute the solutions of $u(x,\tau)$ at time $\tau = (n+1)\Delta\tau$, which are denoted by red dots. Therefore, the proposed method can save memory space and computational costs.

For the convenience of calculation, a constant volatility was used in this paper. The proposed method can be used to compute the generalized BS equation with a space- and time-dependent local volatility function. The classical BS Eq. (2) can be generalized to include the general volatility $\sigma(S,t)$ as follows (Kim et al., 2021; Kim & Kim, 2021):

$$\frac{\partial u(S,\tau)}{\partial \tau} = \frac{1}{2}(\sigma(S,\tau)S)^2\frac{\partial^2 u(S,\tau)}{\partial S^2} + rS\frac{\partial u(S,\tau)}{\partial S} - ru(S,\tau),$$

for $(S,\tau) \in \Omega \times (0,T)$. In future work, we will apply our proposed numerical method to the generalized BS equation to reconstruct a local volatility surface that practitioners in the financial industry are interested in.

5 Conclusions

In this study, the fast explicit numerical scheme was presented for the BS model with no far-field boundary conditions. The Saul'yev-type scheme for the temporal discretization and non-uniform grids for the underlying asset variable were used; therefore, practically sufficiently large time steps can be employed. Thus, the proposed scheme is fast and efficient. In particular, the proposed method is good for nonlinear boundary profiles such as power options because it does not require far-field boundary conditions. To confirm the speed and efficiency of the proposed scheme, various computational experiments were performed. The computational results confirmed the superior performance of the proposed method. Future research topics include the extension of the proposed method to high-dimensional BS equations (Ullah, 2020), nonlinear BS equations (Al-Zhour et al., 2019), high-order method for BS equations (Hu & Gan, 2018), fractional BS equations (Chen & Wang, 2020; Golbabai et al., 2019), and Hull–White PDE (Lee & Yang, 2020).

Appendix

The following codes are MATLAB scripts for the European call and powered options.

```
% One asset European call option code
clear all; format short e
r=0.03; sig=0.3; T=0.1; K=100.0;
for k0=1:3
for k1=1:5
  maxit=200*2^(k1-1); dt=T/maxit; a=2^(2-k0); x=[0:a:100+1*a+a*maxit];
  h=diff(x); Nx=length(x); u=max(x-K,0); u0=u;
  for k=1:maxit
  for i=2:Nx-k
  fac=1.0/dt+0.5*r+(sig*x(i))^2/(h(i-1)*(h(i-1)+h(i))) ...
      -r*x(i)*h(i)/(h(i-1)*(h(i-1)+h(i)));
  u(i)=(u(i)/dt+(sig*x(i))^2*((u(i+1)-u(i))/(h(i)*(h(i-1)+h(i))) ...
       +u(i-1)/(h(i-1)*(h(i-1)+h(i)))) ...
       +r*x(i)*(  h(i-1)*(u(i+1)-u(i))/(h(i)*(h(i-1)+h(i))) ...
       -h(i)*u(i-1)/(h(i-1)*(h(i-1)+h(i))))-0.5*r*u(i))/fac;
  end
  end
d1=(log(x/K)+(r+0.5*sig^2)*T)/(sig*sqrt(T)); d2=d1-sig*sqrt(T);
exu = x.* normcdf(d1) - K*exp(-r*T)*normcdf(d2); clf;
plot(x,u,'o',x,u0,'k*-',x,exu,'r-');axis([0 x(Nx-k) 0 20]);pause(.1);
R(k0,k1)=abs(interp1(x,u,100)-interp1(x,exu,100))/interp1(x,exu,100);
end
end
R
```

```
% One asset Powered option code
clear all; format short e
r=0.03; sig=0.3; T=0.1; K=100.0; pw=2;
for k0=1:3
for k1=1:5
  maxit=200*2^(k1-1); dt=T/maxit; a=2^(2-k0);
  x=[0 a:a:100+1*a+a*maxit]; h=diff(x); Nx=length(x);
  u=max(x-K,0).^pw; u0=u; exu=zeros(1,Nx); clf;
  for k=1:maxit
  for i=2:Nx-k
   fac=1.0/dt+0.5*r+(sig*x(i))^2/(h(i-1)*(h(i-1)+h(i))) ...
       -r*x(i)*h(i)/(h(i-1)*(h(i-1)+h(i)));
   u(i)=(u(i)/dt+(sig*x(i))^2*((u(i+1)-u(i))/(h(i)*(h(i-1)+h(i))) ...
        +u(i-1)/(h(i-1)*(h(i-1)+h(i)))) ...
        +r*x(i)*(   h(i-1)*(u(i+1)-u(i))/(h(i)*(h(i-1)+h(i))) ...
        -h(i)*u(i-1)/(h(i-1)*(h(i-1)+h(i))) ) -0.5*r*u(i))/fac;
  end
  end
  for i=1:Nx
    c=0; S=x(i);
    for j=0:pw
      d=(log(S/K)+(r+(pw-j-1/2)*sig^2)*T)/(sig*sqrt(T));
      c=c+factorial(pw)/(factorial(j)*factorial(pw-j))*S^(pw-j) ...
        *((-K)^j)*exp((pw-j-1)*(r+(pw-j)*(sig^2)/2)*T)*normcdf(d);
    end
    exu(i)=c;
  end
plot(x,u,'o',x,u0,'k*-',x,exu,'r-');axis([0 x(Nx-k) 0 50]);pause(.1)
R(k0,k1)=abs(interp1(x,u,100)-interp1(x,exu,100))/interp1(x,exu,100);
end
end
R
```

```
% Two asset European call option code
clear all; format short e
r=0.015; sigx=0.3; sigy=0.3; rho=0.3; T=0.1; K1=100.0; K2=100.0;
for k0=1:3
for k1=1:5
  maxit=100*2^(k1-1); dt=T/maxit; a=2^(2-k0);
  x=[0:a:100+1*a+a*maxit]; y=x; X=x;
  hx=diff(x); hy=hx; Nx=length(x); Ny=Nx; u=zeros(Nx,Ny);
  for i=1:Nx
  for j=1:Ny
    if (x(i)≥K1) && (y(j)≥K2)
      u(i,j)=100.0;
    else
      u(i,j)=0.0;
    end
  u(i,j)=max(max(x(i)-K1,0),y(j)-K2);
  end
  end
  u0=u;
  for k=1:maxit
    v=u;
    for i=2:Nx-k
    for j=2:Ny-k
      fac=1/dt+r/2+((sigx*x(i))^2-r*x(i)*hx(i))...
          /(hx(i-1)*(hx(i-1)+hx(i)))  ...
          +((sigy*y(j))^2-r*y(j)*hy(j))/(hy(j-1)*(hy(j-1)+hy(j)));
      u(i,j)=(u(i,j)/dt+(sigx*x(i))^2  ...
        *((u(i+1,j)-u(i,j))/(hx(i)*(hx(i-1)+hx(i)))  ...
        + u(i-1,j)/(hx(i-1)*(hx(i-1)+hx(i))))  ...
        + rho*sigx*sigy*x(i)*y(j)...
        *(v(i+1,j+1) - v(i+1,j) -v(i,j+1) +v(i,j))/(hx(i)*hy(j))  ...
        + (sigy*y(j))^2*((u(i,j+1)-u(i,j))/(hy(j)*(hy(j-1)+hy(j)))  ...
        + u(i,j-1)/(hy(j-1)*(hy(j-1)+hy(j))))  ...
        + r*x(i)*(hx(i-1)*(u(i+1,j)-u(i,j))/(hx(i)*(hx(i-1)+hx(i)))...
        - hx(i)*u(i-1,j)/(hx(i-1)*(hx(i-1)+hx(i))))  ...
        + r*y(j)*(hy(j-1)*(u(i,j+1)-u(i,j))/(hy(j)*(hy(j-1)+hy(j))))...
        - hy(j)*u(i,j-1)/(hy(j-1)*(hy(j-1)+hy(j))))  ...
        - (r/2)*u(i,j))/fac;
    end
    end
    if mod(k,30)==0
      clf; mesh(x(2:end-k),y(2:end-k),u(2:end-k,2:end-k));
      axis([0 x(end) 0 y(end) 0 x(end)])
    end
  end
sig=sqrt(sigx^2+sigy^2-2*rho*sigx*sigy);
rho1=(sigx-rho*sigy)/sig; rho2=(sigy-rho*sigx)/sig;
d=(log(x/y)+0.5*sig^2*T)/(sig*sqrt(T));
y1=(log(x/X)+0.5*sigx^2*T)/(sig*sqrt(T));
y2=(log(y/X)+0.5*sigy^2*T)/(sig*sqrt(T));
V=x*mvncdf([y1 d],[0 0],[1 rho1; rho1 1])  ...
  +y*mvncdf([y2 -d+sig*sqrt(T)],[0 0],[1 rho2; rho2 1])  ...
  -X*exp(-r*T)*(1-mvncdf([-y1+sigx*sqrt(T)  ...
  -y2+sigy*sqrt(T)],[0 0],[1 rho; rho 1]));
id = find(x==100);
NSol(k0,k1)=u(id,id);
R(k0,k1)=abs(NSol(k0,k1)-V(id))/V(id);
end
end
R
```

Acknowledgements The first author (C. Lee) was supported by the National Research Foundation(NRF), Republic of Korea, under project BK21 FOUR. The corresponding author (J.S. Kim) was supported by Korea University Grant. The authors thank the reviewers for their useful comments regarding this paper.

Declarations

Conflict of interest The author declares that there is no conflict of interest.

References

Abdi-Mazraeh, S., Khani, A., & Irandoust-Pakchin, S. (2020). Multiple shooting method for solving Black-Scholes equation. *Computational Economics, 56*(4), 723–746.

Al-Zhour, Z., Barfeie, M., Soleymani, F., & Tohidi, E. (2019). A computational method to price with transaction costs under the nonlinear Black-Scholes model. *Chaos, Solitons & Fractals, 127*, 291–301.

Anwar, M. N., & Andallah, L. S. (2018). A study on numerical solution of Black-Scholes model. *Journal of Mathematical Finance, 8*(2), 372–381.

Black, F., & Scholes, M. (1973). The pricing of options and corporate liabilities. *The Journal of Political Economy, 81*(3), 637–654.

Chen, W., & Wang, S. (2020). A 2nd-order ADI finite difference method for a 2D fractional Black-Scholes equation governing European two asset option pricing. *Mathematics and Computers in Simulation, 171*, 279–293.

Choi, Y., Jeong, D., Kim, J., Kim, Y. R., Lee, S., Seo, S., & Yoo, M. (2015). Robust and accurate method for the Black-Scholes equations with payoff-consistent extrapolation. *Communications of the Korean Mathematical Society, 30*(3), 297–311.

Dubey, V. P., Kumar, R., & Kumar, D. (2019). A reliable treatment of residual power series method for time-fractional Black-Scholes European option pricing equations. *Physica A: Statistical Mechanics and its Applications, 533,* 122040.

Farnoosh, R., Rezazadeh, H., Sobhani, A., & Beheshti, M. H. (2016). A numerical method for discrete single barrier option pricing with time-dependent parameters. *Computational Economics, 48*(1), 131–145.

Feng, C., Tan, J., Jiang, Z., & Chen, S. (2020). A generalized European option pricing model with risk management. *Physica A: Statistical Mechanics and its Applications, 545,* 123797.

Golbabai, A., Nikan, O., & Nikazad, T. (2019). Numerical analysis of time fractional Black-Scholes European option pricing model arising in financial market. *Computational and Applied Mathematics, 38*(4), 173.

Golbabai, A., & Nikan, O. (2020). A computational method based on the moving least-squares approach for pricing double barrier options in a time-fractional Black-Scholes model. *Computational Economics, 55*(1), 119–141.

Haug, E. G. (2007). *The complete guide to option pricing formulas*. McGraw-Hill Education.

Hanh, T. T. H., & Thanh, D. N. H. (2019). Finite-difference method for the Gamma equation on non-uniform grids. *Vietnam Journal of Science, Technology and Engineering, 61*(4), 3–8.

Heo, Y., Han, H., Jang, H., Choi, Y., & Kim, J. (2019). Finite difference method for the two-dimensional Black-Scholes equation with a hybrid boundary condition. *Journal of the Korean Society for Industrial and Applied Mathematics, 23*, 19–30.

Hu, J., & Gan, S. (2018). High order method for Black-Scholes PDE. *Computers & Mathematics with Applications, 75*(7), 2259–2270.

Huang, J., Cen, Z., & Zhao, J. (2019). An adaptive moving mesh method for a time-fractional Black-Scholes equation. *Advances in Difference Equations, 2019*, 516.

Jeong, D., Seo, S., Hwang, H., Lee, D., Choi, Y., & Kim, J. (2015). Accuracy, robustness, and efficiency of the linear boundary condition for the Black-Scholes equations. *Discrete Dynamics in Nature and Society, 2015*, 359028.

Jeong, D., Yoo, M., & Kim, J. (2018). Finite difference method for the Black-Scholes equation without boundary conditions. *Computational Economics, 51*(4), 961–972.

Jeong, D., Yoo, M., Yoo, C., & Kim, J. (2019). A hybrid Monte Carlo and finite difference method for option pricing. *Computational Economics, 53*(1), 111–124.

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

Khodayari, L., & Ranjbar, M. (2019). A computationally efficient numerical approach for multi-asset option pricing. *International Journal of Computer Mathematics, 96*(6), 1158–1168.

Kim, N., & Lee, Y. (2018). Estimation and prediction under local volatility jump-diffusion model. *Physica A: Statistical Mechanics and its Applications, 491*, 729–740.
Kim, S., Han, H., Jang, H., Jeong, D., Lee, C., Lee, W., & Kim, J. (2021). Reconstruction of the local volatility function using the Black-Scholes model. *Journal of Computational Science, 51*, 101341.
Kim, S., & Kim, J. (2021). Robust and accurate construction of the local volatility surface using the Black-Scholes equation, Chaos. *Solitons & Fractals, 150*, 111116.
Koffi, R. S., & Tambue, A. (2020). A fitted multi-point flux approximation method for pricing two options. *Computational Economics, 55*(2), 597–628.
Koleva, M. N., & Vulkov, L. G. (2017). Numerical solution of time-fractional Black-Scholes equation. *Computational and Applied Mathematics, 36*(4), 1699–1715.
Lyu, J., Park, E., Kim, S., Lee, W., Lee, C., Yoon, S., Park, J., & Kim, J. (2021). Optimal non-uniform finite difference grids for the Black-Scholes equations. *Mathematics and Computers in Simulation, 182*, 690–704.
Lee, Y., & Yang, K. (2020). Finite difference method for the Hull–White partial differential equations., *Mathematics, 8*(10), 1719.
Ma, C., Ma, Z., & Xiao, S. (2019). A closed-form pricing formula for vulnerable European options under stochastic yield spreads and interest rates. *Chaos, Solitons & Fractals, 123*, 59–68.
Nikan, O., Avazzadeh, Z., & Machado, J. T. (2021). A local stabilized approach for approximating the modified time-fractional diffusion problem arising in heat and mass transfer. *Journal of Advanced Research, 32*, 45–60.
Nikan, O., & Avazzadeh, Z. (2021). An improved localized radial basis-pseudospectral method for solving fractional reaction-subdiffusion problem. *Results in Physics, 23,* 104048.
Roul, P., & Goura, V. P.(2020). A new higher order compact finite difference method for generalised Black–Scholes partial differential equation: European call option, *Journal of Computational and Applied Mathematics, 363,* 464–484.
Roul, P., & Goura, V. P. (2020). A sixth order numerical method and its convergence for generalized Black-Scholes PDE. *Journal of Computational and Applied Mathematics, 377*, 112881.
Sauer, T. (2013). Computational solution of stochastic differential equations. *Wiley Interdisciplinary Reviews: Computational Statistics, 5*(5), 362–371.
Saul'Yev, V. K. (1964). *Integration of equations of parabolic type by the method of nets*. Pergamon.
Soleymani, F., & Akgül, A. (2019). Improved numerical solution of multi-asset option pricing problem: A localized RBF-FD approach. *Chaos, Solitons & Fractals, 119*, 298–309.
Sosa-Correa, W. O., Ramos, A. M., & Vasconcelos, G. L. (2018). Investigation of non-Gaussian effects in the Brazilian option market. *Physica A: Statistical Mechanics and its Applications, 496*, 525–539.
Ullah, M. Z. (2020). An RBF-FD sparse scheme to simulate high-dimensional Black-Scholes partial differential equations. *Computers & Mathematics with Applications, 79*(2), 426–439.
Windcliff, H., Forsyth, P. A., & Vetzal, K. R. (2004). Analysis of the stability of the linear boundary condition for the Black-Scholes equation. *Journal of Computational Finance, 8*, 65–92.
Zhang, P. G. (1998). *A guide to second generation options*. World Scientific.