Advances in Applied Mathematics and Mechanics
Adv. Appl. Math. Mech., Vol. **1**, No. 6, pp. 845-861

DOI: 10.4208/aamm.09-m09S06
December 2009

Fourth Order Compact Boundary Value Method for Option Pricing with Jumps

Spike T. Lee[1] and Hai-Wei Sun[1,*]

[1] *Department of Mathematics, University of Macau, Macao, China*

Received 26 February 2009; Accepted (in revised version) 28 August 2009
Available online 18 November 2009

Abstract. We consider pricing options in a jump-diffusion model which requires solving a partial integro-differential equation. Discretizing the spatial direction with a fourth order compact scheme leads to a linear system of ordinary differential equations. For the temporal direction, we utilize the favorable boundary value methods owing to their advantageous stability properties. In addition, the resulting large sparse system can be solved rapidly by the GMRES method with a circulant Strang-type preconditioner. Numerical results demonstrate the high order accuracy of our scheme and the efficiency of the preconditioned GMRES method.

AMS subject classifications: 65T50, 65M06, 65F10, 91B28
Key words: Partial integro-differential equation, fourth order compact scheme, boundary value method, preconditioning, Toeplitz matrix.

1 Introduction

One of the most influential financial models is the jump-diffusion model presented by Merton [19] in 1976. In Merton's model, the asset return follows a standard Brownian process impelled by a compound Poisson process with normally distributed jumps. Under this assumption, the value of a contingent claim satisfies a partial integro-differential equation (PIDE). A PIDE usually comprises a differential operator and a non-local integral term. Numerical methods for solving PIDEs have already been widely studied [1, 2, 10–12, 15, 24]. However, the commonly used central difference discretization is only second order accurate in the spatial direction. Recently the authors of this paper proposed to apply a fourth order compact (FOC) finite difference scheme with local mesh refinement strategy to attain fourth order convergence in the spatial direction [18]. The approach in [18] is to firstly discretize the temporal direction of the PIDE by an implicit-explicit (IMEX) scheme. Then the semi-discretized equation

*Corresponding author.
URL: `http://www.fst.umac.mo/en/staff/fsthws.html`
Email: ma76522@umac.mo (S. T. Lee), HSun@umac.mo (H. W. Sun)

at each time step can be naturally approximated by the FOC scheme. Though convenient to use, the IMEX scheme is only first order accurate. Therefore one has to employ the extrapolation strategy to reach high order accuracy for the time direction [12], and thus the operation cost depends highly on the number of extrapolation stages.

The unwanted workload of the above treatment pushes us to seek an alternative path. In 2003, Sun and Zhang [23] combined the boundary value method (BVM) with FOC scheme to solve one-dimensional heat equations. The main concept in [23] is as follows. After discretizing the derivatives of the spatial variable by FOC scheme, they utilize a high order BVM to approximate the semi-discretized linear system of ordinary differential equations (ODEs). The BVMs are a class of numerical methods based on the linear multistep formula (LMF) for solving initial value problems (IVPs) of ODEs [5,6]. In particular, the unconditional stability properties of BVMs make them preferable over other initial value methods (IVMs) [6]. Furthermore, one can obtain a high order BVM by implementing the LMF properly [5], i.e., there is no need to use any extrapolation strategy.

In this paper, we consider solving a PIDE by extending Sun-Zhang's idea. Unlike [18], we first carry out a three point FOC discretization for the spatial direction, and then a high order BVM is employed for the semi-discretized ODE system. This combination, known as the FOCBVM, retains high order accuracy and remarkable stability property at the same time. Unfortunately, we remark that the initial condition is always non-smooth in option pricing theory. In the spatial direction, a specific local mesh refinement strategy [18, 25] is required to ease the impact of the non-smooth payoff function and restore fourth order convergence of the FOC scheme. In the temporal direction, as stated in [13, 20, 24], a numerical correction process should be executed beforehand or numerical oscillations are likely to spoil the desired convergence. For a high order BVM, we can retrieve the expected result by replacing the approximations of the beginning time steps with the second order backward difference formula (BDF2). The BDF2 has been previously used by Almendral and Oosterlee [1] to achieve second order convergence in time, and they showed that the linear system at each time step can be swiftly solved by an iterative method based on the regular splitting of matrices.

Nevertheless, a direct solver for FOCBVM may not be a wise move because of the incredibly huge size of the resulting system. In fact, this is a major challenge for solving systems of LMF-based ODE codes. In 2000, Bertaccini [3] proposed to use the Krylov subspace method with block-circulant preconditioners to solve such systems. From then on, many circulant preconditioners are designed to pair with the GMRES method for the same purpose [8,16,17]. In this paper, we follow [8] 's idea to speed up calculation by constructing a Strang-type circulant preconditioner. We will see from the numerical results that the preconditioned GMRES method works very well.

The rest of the paper is organized as follows. In Section 2 we apply the FOC scheme with local mesh refinement strategy to discretize the spatial direction of the PIDE. In Section 3 we introduce the BVM implementation and the BDF2 startup procedure. The preconditioning technique is discussed in Section 4. In Section 5 we illustrate the

numerical results. Concluding remarks are given in Section 6.

2 FOC scheme and local mesh refinement

First we briefly introduce how to price a European call option for a single underlying asset in Merton's jump-diffusion model [19], where jumps are normally distributed with mean δ and variation γ. The option value $v(x,t)$ with logarithmic price x and backward time t satisfies a forward PIDE on $(-\infty,+\infty)\times[0,T]$:

$$v_t = \frac{\sigma^2}{2}v_{xx} + (r-\lambda\kappa-\frac{\sigma^2}{2})v_x - (r+\lambda)v + \lambda\int_{-\infty}^{\infty} v(x+z,t)f(z)dz, \tag{2.1}$$

where T is the maturity time, σ is the stock return volatility, r is the risk-free interest rate, λ is the arrival intensity of a Poisson process,

$$\kappa = e^{(\delta+\gamma^2/2)} - 1,$$

is the expectation of the impulse function and f is the probability density function of the Gaussian distribution given by

$$f(z) = \frac{e^{\frac{-(z-\delta)^2}{2\gamma^2}}}{\sqrt{2\pi}\gamma}.$$

For an European call option, the initial condition is

$$v(x,0) = \max(Ke^x - K, 0), \tag{2.2}$$

where K is the strike price. Boundary conditions are

$$v(x,t) \approx \begin{cases} 0, & \text{as } x \to -\infty, \\ Ke^x - Ke^{-rt}, & \text{as } x \to +\infty. \end{cases} \tag{2.3}$$

See [1, 2, 10–12, 15, 18, 19, 24] for details. Our aim is to determine the option value $v(x,T)$.

2.1 FOC scheme

Here we will apply an FOC scheme to discretize the spatial variable of (2.1), see [18, 22, 23] for details. The coefficients of (2.1) are transformed into constants only for simplicity and boundary conditions can be imposed when a finite computational domain is used [15]. Hence we first truncate the infinite x-domain $(-\infty,\infty)$ to $[x_{\min}, x_{\max}]$. Let h_x be a uniform mesh size, then the computational spatial domain is denoted by

$$\{x_l \in \mathbb{R} : x_l = x_{\min} + lh_x,\ l = 0,1,2,\ldots,m+1\} \triangleq \Omega_m \cup \{x_{\min}, x_{\max}\},$$

where $x_0 = x_{\min}$, $x_{m+1} = x_{\max}$ and Ω_m denotes the set of inner grid points. Note that $x_{\min}$, $x_{\max}$ are selected appropriately such that $x^* = 0 \in \Omega_m$.

Rewrite the PIDE (2.1) as

$$v_t = av_{xx} + bv_x - (r+\lambda)v + \lambda(v * f), \tag{2.4}$$

where

$$a = \frac{1}{2}\sigma^2, \qquad b = r - \lambda\kappa - \frac{\sigma^2}{2}, \tag{2.5}$$

and the convolution integral

$$(v * f)(x,t) = \int_{-\infty}^{\infty} v(x+z,t)f(z)dz = \int_{-\infty}^{\infty} v(y,t)f(y-x)dy.$$

Let $v_l(t)$ and $(v * f)_l(t)$ be the discrete approximations to $v(x_l,t)$ and $(v * f)(x_l,t)$ respectively. A three point stencil of FOC difference scheme for (2.4) can be written in matrix form:

$$[B\mathbf{v}(t) + \mathbf{c}(t)]' = A\mathbf{v}(t) + \mathbf{d}_1(t) + \lambda B(\mathbf{v} * f)(t) + \mathbf{d}_2(t), \tag{2.6}$$

where the entries of tridiagonal matrices B, $A \in \mathbb{R}^{m\times m}$ are

$$A(k_1,k_2) = \begin{cases} \zeta_1 - (r+\lambda)\eta_1, & \text{if } k_1 = k_2 + 1, \\ -\zeta_1 - \zeta_2 - (r+\lambda)(1-\eta_1-\eta_2), & \text{if } k_1 = k_2, \\ \zeta_2 - (r+\lambda)\eta_2, & \text{if } k_1 = k_2 - 1, \\ 0, & \text{otherwise,} \end{cases}$$

$$B(k_1,k_2) = \begin{cases} \eta_1, & \text{if } k_1 = k_2 + 1, \\ 1-\eta_1-\eta_2, & \text{if } k_1 = k_2, \\ \eta_2, & \text{if } k_1 = k_2 - 1, \\ 0, & \text{otherwise,} \end{cases}$$

and

$$\begin{cases} \eta_1 = \dfrac{1}{12} - \dfrac{bh_x}{24a}, & \eta_2 = \dfrac{1}{12} + \dfrac{bh_x}{24a}, \\ \zeta_1 = \dfrac{b^2}{12a} + \dfrac{a}{h_x^2} - \dfrac{b}{2h_x}, & \zeta_2 = \dfrac{b^2}{12a} + \dfrac{a}{h_x^2} + \dfrac{b}{2h_x}. \end{cases} \tag{2.7}$$

All the vectors are given by

$$\mathbf{v}(t) = \begin{bmatrix} v_1(t) \\ v_2(t) \\ \vdots \\ v_m(t) \end{bmatrix}, \ (\mathbf{v} * f)(t) = \begin{bmatrix} (v*f)_1(t) \\ (v*f)_2(t) \\ \vdots \\ (v*f)_m(t) \end{bmatrix}, \ \mathbf{c}(t) = \eta_2 \begin{bmatrix} 0 \\ 0 \\ \vdots \\ Ke^{x_{\max}} - Ke^{-rt} \end{bmatrix},$$

$$\mathbf{d}_1(t) = [\zeta_2 - (r+\lambda)\eta_2] \begin{bmatrix} 0 \\ 0 \\ \vdots \\ Ke^{x_{\max}} - Ke^{-rt} \end{bmatrix}, \quad \mathbf{d}_2(t) = \lambda\eta_2 \begin{bmatrix} 0 \\ 0 \\ \vdots \\ (v*f)_{\max}(t) \end{bmatrix}.$$

2.2 Evaluation of the convolution integral

To obtain $(\mathbf{v} * f)(t)$ and $\mathbf{d}_2(t)$, we have to evaluate the following convolution integral:

$$(v * f)(x,t) = \int_{-\infty}^{\infty} v(y,t) f(y-x) dy.$$

For the non-local y-domain, we apply the splitting method which is used in [1, 11]. The concept of this method is to divide the y -domain into $[x_1, x_m]$ and $[x_1, x_m]^c = (-\infty, x_1] \cup [x_m, \infty)$. It follows that

$$\begin{aligned}(v * f)(x,t) &= \int_{x_1}^{x_m} v(y,t) f(y-x) dy + \int_{[x_1,x_m]^c} v(y,t) f(y-x) dy \\ &\triangleq v_{loc}(x,t) + v_c(x,t).\end{aligned}$$

Hence we have

$$(\mathbf{v} * f)(t) = \mathbf{v}_{loc}(t) + \mathbf{v}_c(t),$$

where

$$\begin{aligned}&\mathbf{v}_{loc}(t) = [v_{loc}(x_1,t), v_{loc}(x_2,t), \ldots, v_{loc}(x_m,t)]^{\mathsf{T}}, \\ \text{and} \quad &\mathbf{v}_c(t) = [v_c(x_1,t), v_c(x_2,t), \ldots, v_c(x_m,t)]^{\mathsf{T}}.\end{aligned}$$

To evaluate the first local integral, we discretize the integral domain by the uniform mesh $\{x_1, x_2, \ldots, x_m\}$ and write its discrete form by using fourth order accurate composite Simpson's rule:

$$\begin{aligned}v_{loc}(x,t) =& \frac{h_x}{3}\Big[v(x_1,t) f(x_1-x) + 2\sum_{k=1}^{\frac{m-3}{2}} v(x_{2k+1},t) f(x_{2k+1}-x) \\ &+ 4\sum_{k=1}^{\frac{m-1}{2}} v(x_{2k},t) f(x_{2k}-x) + v(x_m,t) f(x_m-x)\Big] + \mathcal{O}(h_x^4). \qquad (2.8)\end{aligned}$$

Substituting the spatial nodes $x_1, x_2, \ldots, x_m$ into (2.8), we can write $\mathbf{v}_{loc}(t)$ in a matrix form $T_f \mathbf{v}(t)$, where

$$T_f = \frac{h_x}{3}\begin{bmatrix} f(0) & 4f(x_2-x_1) & 2f(x_3-x_1) & \cdots & f(x_m-x_1) \\ f(x_1-x_2) & 4f(0) & 2f(x_3-x_2) & \cdots & f(x_m-x_2) \\ f(x_1-x_3) & 4f(x_2-x_3) & 2f(0) & \cdots & f(x_m-x_3) \\ \vdots & \vdots & \vdots & & \vdots \\ f(x_1-x_m) & 4f(x_2-x_m) & 2f(x_3-x_m) & \cdots & f(0) \end{bmatrix},$$

is an m-by-m matrix. We now reduce the local integral to a matrix-vector product $T_f \mathbf{v}(t)$. Note that T_f is restored to an m-by-m Toeplitz matrix when the coefficients $[1, 4, 2, \ldots, 4, 2, 1]^{\mathsf{T}}$ of the composite Simpson's formula are transferred to the vector $\mathbf{v}(t)$. If a matrix-vector multiplication $T_f \mathbf{v}(t)$ is required, we can carry out a fast Fourier transform (FFT) for rapid evaluation [7, 9].

For the second integral $\mathbf{v}_c(t)$, we have already known from (2.3) that $v(y,t)$ can be approximated by asymptotic boundary conditions. Hence direct calculation gives

$$\begin{aligned} v_c(x,t) &\approx \int_{x_m}^{\infty} (Ke^y - Ke^{-rt}) f(y-x)dy \\ &= Ke^{x+\delta+\gamma^2/2}\mathbb{N}\left(\frac{x-x_m+\delta+\gamma^2}{\gamma}\right) - \mathbb{K}e^{-rt}\mathbb{N}\left(\frac{x-x_m+\delta}{\gamma}\right), \end{aligned} \tag{2.9}$$

where $\mathbb{N}$ is the standard normal cumulative distribution function. After all, we obtain

$$(\mathbf{v} * f)(t) = \mathbf{v}_{loc}(t) + \mathbf{v}_c(t) = T_f\mathbf{v}(t) + \mathbf{v}_c(t).$$

Moreover, the integral on the boundary point $x_{\max}$ is derived in the same way:

$$\mathbf{d}_2(t) = \lambda\eta_2 \left[T_{\max}\mathbf{v}(t) + \mathbf{d}_3(t)\right],$$

where

$$T_{\max} = \frac{h_x}{3}\begin{bmatrix} 0 & 0 & 0 & \cdots & 0 \\ 0 & 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & & \vdots \\ f(x_1-x_{\max}) & 4f(x_2-x_{\max}) & 2f(x_3-x_{\max}) & \cdots & f(x_m-x_{\max}) \end{bmatrix},$$

and $\mathbf{d}_3(t) = [0,0,\ldots,v_c(x_{\max},t)]^{\intercal}$. Consequently, the ODE system (2.6) is simplified as

$$[B_x\mathbf{v}(t) + \mathbf{c}(t)]' = A_x\mathbf{v}(t) + \mathbf{d}(t), \tag{2.10}$$

where

$$\begin{aligned} &B_x = B, \qquad A_x = A + \lambda B_x T_f + \lambda\eta_2 T_{\max}, \\ &\mathbf{d}(t) = \mathbf{d}_1(t) + \lambda B\mathbf{v}_c(t) + \lambda\eta_2\mathbf{d}_3(t). \end{aligned}$$

2.3 Local mesh refinement

However, it is shown in [18] that the FOC scheme on a uniform grid only achieved second order accuracy. It is because the financial payoff function of a European call option (2.2) is non-smooth around x^*=0. Thus grid points concentration near x^* is necessary. In this paper, we prefer the local mesh refinement strategy [18,25] because it is easily adapted to the three-point stencil of FOC scheme. Assume that the spatial direction is first discretized by a uniform mesh containing x^* with initial mesh size h_x. After applying the local mesh refinement strategy in [18], there are $4\lceil -\log_2 h_x\rceil$ new nodes all together, where the operator $\lceil x\rceil$ rounds x to the nearest integer greater than or equal to x. Let Ω_{new} be the set of new grid points and $n = m + 4\lceil -\log_2 h_x\rceil$. Note that $4\lceil -\log_2 h_x\rceil$ is a number of order $\mathcal{O}(\log m)$, thus n is close to m. Then the refined computational domain is the following union:

$$\{x_{\min}, x_1, \ldots, x_m, x_{\max}\} \cup \Omega_{new} \triangleq \Omega_n \cup \{x_{\min}, x_{\max}\},$$

where $\Omega_n=\Omega_m \cup \Omega_{new}$ denotes the set of inner grid points in the refined domain. After refining the grid nodes, we need to move from Ω_m to Ω_n and modify all the operator matrices. For the differential operators, the original tridiagonal matrices B, $A \in \mathbb{R}^{m\times m}$ are now tridiagonal-like matrices $\widehat{B}$, $\widehat{A} \in \mathbb{R}^{n\times n}$. Note that $\widehat{B}$ and $\widehat{A}$ remain sparse as the three-point stencil of FOC scheme only produces three non-zero entries on each row. For the integral operator, we keep the splitting method as before and compute the second non-local integral directly by (2.9). The local integral is similarly approximated by the fourth order composite Simpson's rule and its corresponding operator is an n-by-n matrix $\widehat{T}_f$. Note that $T_f \in \mathbb{R}^{m\times m}$ is a submatrix of $\widehat{T}_f \in \mathbb{R}^{n\times n}$ and $n-m$ is merely a number of order $\mathcal{O}(\log m)$. Therefore T_f takes an overwhelmingly large part of $\widehat{T}_f$ and we can make good use of the Toeplitz-like structure of T_f. The semi-discretized ODE system (2.10) is then altered to

$$\left[\widehat{B}_x\widehat{\mathbf{v}}(t)+\widehat{\mathbf{c}}(t)\right]' = \widehat{A}_x\widehat{\mathbf{v}}(t)+\widehat{\mathbf{d}}(t), \tag{2.11}$$

where $\widehat{B}_x=\widehat{B}$, $\widehat{A}_x$ is a combination of $\widehat{A}$, $\lambda\widehat{B}_x\widehat{T}_f$ and a certain rank-1 matrix. The definitions of all the vectors $\widehat{\mathbf{v}}(t)$, $\widehat{\mathbf{c}}(t)$, $\widehat{\mathbf{d}}(t)$ of length n are analogous to those of $\mathbf{v}(t)$, $\mathbf{c}(t)$, $\mathbf{d}(t)$ of length m.

3 BVMs for ODE system and BDF2 startup

3.1 BVMs and their matrix forms

After the discretization of spatial direction, the semi-discretized system (2.11) is an IVP of ODEs with initial condition (2.2). That is why we turn to the BVMs, which are numerical methods based on the LMF for solving ODEs. For simplicity, we first consider solving the following general IVP:

$$\begin{cases} u'(t) = g(t,u), \quad t\in[0,T], \\ u(0) = u_0, \end{cases} \tag{3.1}$$

by BVMs. Suppose the time direction $[0,T]$ is first divided into s time steps with $h_t=T/s$ and $t_j=jh_t$, $j=0,1,2,\ldots,s$. Let u_j be an approximation to $u(t_j)$ and g_j be an approximation to $g(t_j,u(t_j))$. By using a μ-step LMF, we have the following relations:

$$\sum_{l=-\nu}^{\mu-\nu}\alpha_{l+\nu}u_{j+l} = h_t\sum_{l=-\nu}^{\mu-\nu}\beta_{l+\nu}g_{j+l}, \quad j=\nu,\ldots,s-\mu+\nu, \tag{3.2}$$

where α_l and β_l are determined by different choice of BVM [5]. The BVM in (3.2) should be used with v initial conditions and $(\mu-v)$ final conditions. However, the IVP (3.1) only gives the initial value u_0. To obtain the other initial and final values, we

can apply adequate difference methods and obtain the additional $\mu-1$ equations:

$$\sum_{l=0}^{\mu}\alpha_l^{(k)}u_l = h_t\sum_{l=0}^{\mu}\beta_l^{(k)}g_l, \quad k=1,\dots,\nu-1, \tag{3.3}$$

$$\sum_{l=0}^{\mu}\alpha_{\mu-l}^{(k)}u_{s-l} = h_t\sum_{l=0}^{\mu}\beta_{\mu-l}^{(k)}g_{s-l}, \quad k=s-\mu+\nu+1,\dots,s. \tag{3.4}$$

Suppose p-th order accurate BVM formulas (3.2), (3.3) and (3.4) are chosen. Applying these formulas to (3.1), we obtain a linear system in matrix form as follows:

$$A_t\mathbf{u} = h_tB_t\mathbf{g} + \mathbf{e}_1u_0 + \mathcal{O}(h_t^p),$$

where $\mathbf{u}=[u_0,u_1,\dots,u_s]^\intercal$, $\mathbf{g}=[g_0,g_1,\dots,g_s]^\intercal$, $\mathbf{e}_1=[1,0,\dots,0]^\intercal\in\mathbb{R}^{(s+1)}$ and A_t and B_t are $(s+1)$ -by-$(s+1)$ banded matrices:

$$A_t = \begin{bmatrix}
1 & \cdots & 0 & & & & \\
\alpha_0^{(1)} & \cdots & \alpha_\mu^{(1)} & & & & \\
\vdots & \vdots & \vdots & & & & \\
\alpha_0^{(\nu-1)} & \cdots & \alpha_\mu^{(\nu-1)} & & & & \\
\alpha_0 & \cdots & \alpha_\mu & & & & \\
 & \alpha_0 & \cdots & \alpha_\mu & & & \\
 & & \ddots & \ddots & \ddots & & \\
 & & & \ddots & \ddots & \ddots & \\
 & & & & \alpha_0 & \cdots & \alpha_\mu \\
 & & & & \alpha_0^{(s-\mu+\nu+1)} & \cdots & \alpha_\mu^{(s-\mu+\nu+1)} \\
 & & & & \vdots & \vdots & \vdots \\
 & & & & \alpha_0^{(s)} & \cdots & \alpha_\mu^{(s)}
\end{bmatrix},$$

and

$$B_t = \begin{bmatrix}
0 & \cdots & 0 & & & & \\
\beta_0^{(1)} & \cdots & \beta_\mu^{(1)} & & & & \\
\vdots & \vdots & \vdots & & & & \\
\beta_0^{(\nu-1)} & \cdots & \beta_\mu^{(\nu-1)} & & & & \\
\beta_0 & \cdots & \beta_\mu & & & & \\
 & \beta_0 & \cdots & \beta_\mu & & & \\
 & & \ddots & \ddots & \ddots & & \\
 & & & \ddots & \ddots & \ddots & \\
 & & & & \beta_0 & \cdots & \beta_\mu \\
 & & & & \beta_0^{(s-\mu+\nu+1)} & \cdots & \beta_\mu^{(s-\mu+\nu+1)} \\
 & & & & \vdots & \vdots & \vdots \\
 & & & & \beta_0^{(s)} & \cdots & \beta_\mu^{(s)}
\end{bmatrix}.$$

See [3,5,8] for details.

3.2 FOCBVM

Recall the FOC-discretized ODE system (2.11) and let φ be the discrete initial condition. We consider the solution of

$$\begin{cases} \left[\widehat{B}_x\widehat{\mathbf{v}}(t)+\widehat{\mathbf{c}}(t)\right]' = \widehat{A}_x\widehat{\mathbf{v}}(t)+\widehat{\mathbf{d}}(t), \\ \widehat{\mathbf{v}}(0)=\varphi, \end{cases} \tag{3.5}$$

by BVMs. We remark that the formulas (3.2), (3.3) and (3.4) are still applicable in block cases. After applying the p-th order BVM for (3.5), we derive the following discretized form [23]:

$$\begin{aligned} &(A_t\otimes\widehat{B}_x)\overline{\mathbf{v}}+(A_t\otimes I_n)\overline{\mathbf{c}} \\ =&(h_tB_t\otimes\widehat{A}_x)\overline{\mathbf{v}}+(h_tB_t\otimes I_n)\overline{\mathbf{d}}+\mathbf{e}_1\otimes\xi+\mathcal{O}(h_x^4+h_t^p), \end{aligned} \tag{3.6}$$

where $\otimes$ is the Kronecker product, A_t and B_t are defined in the previous section, the vectors are

$$\begin{aligned} &\overline{\mathbf{v}}=[\widehat{\mathbf{v}}(t_0),\widehat{\mathbf{v}}(t_1),\ldots,\widehat{\mathbf{v}}(t_s)]^\intercal, \qquad &&\overline{\mathbf{c}}=[\widehat{\mathbf{c}}(t_0),\widehat{\mathbf{c}}(t_1),\ldots,\widehat{\mathbf{c}}(t_s)]^\intercal, \\ &\overline{\mathbf{d}}=[\widehat{\mathbf{d}}(t_0),\widehat{\mathbf{d}}(t_1),\ldots,\widehat{\mathbf{d}}(t_s)]^\intercal, \qquad &&\xi=\widehat{B}_x\varphi+\widehat{\mathbf{c}}(t_0). \end{aligned}$$

The matrix form of (3.6) is

$$M\overline{\mathbf{v}}=\mathbf{q}, \tag{3.7}$$

where

$$M=A_t\otimes\widehat{B}_x-h_tB_t\otimes\widehat{A}_x, \tag{3.8}$$

and the right-hand side is

$$\mathbf{q}=(h_tB_t\otimes I_n)\overline{\mathbf{d}}+\mathbf{e}_1\otimes\xi-(A_t\otimes I_n)\overline{\mathbf{c}}.$$

3.3 BDF2 startup

However, in the framework of BVMs, we have not taken into account the situation of the initial condition and are unaware of the numerical disasters it might bring. To uncover the temporal malfunction caused by initial conditions, we start with the classic Black-Scholes option pricing model [4]. In the Black-Scholes model, the price of an option satisfies a partial differential equation (PDE) and hence the well-known Crank-Nicolson method is highly recommended. Unfortunately, the unconditionally stable Crank-Nicolson method has been shown to have oscillation issues in time because most of the financial payoff contracts are non-smooth or discontinuous [24]. Take the European call option as an example, the second derivative of the initial data is the Dirac delta function. As a result, the Crank-Nicolson method does not restrain numerical oscillations effectively and fails to converge with second order accuracy as the mesh size decreases.

Since the non-decaying parts usually disappear after a small number of time steps [24], special initialization treatment should be satisfactory. In [20], Rannacher diagnosed the convergence of convection-diffusion approximations with discontinuous initial data. He managed to recover the second order convergence of Crank-Nicolson method by replacing the Crank-Nicolson approximation of the very first time step by two half time steps of Backward Euler timestepping. We refer to this technique as Rannacher timestepping, and it has been significantly used for numerical solutions in computational finance. Giles and Carter [13] also examined the convergence of Crank-Nicolson approximations of one-dimensional convection-diffusion equations and provided an asymptotic analysis.

Consider the semi-discretized (2.11), it is possible to achieve quadratic convergence by Crank-Nicolson approximations and Rannacher timestepping. Since we are aiming for high order convergence, another startup procedure should be exploited. Note that the BDF2 simply reaches second order accuracy in time because it already includes one step of fully implicit method [1]. Thus we decide to choose the second order accurate BDF2 as the startup unit. By any means, the fundamental idea is to replace the approximations of the starting time steps with BDF2 before applying BVMs to the adjusted data.

In this paper, we tend to replace the approximations of the first two time steps $\{0, h_t, 2h_t\}$ to make sure oscillations are dampened. For brevity, we consider applying the BDF2 to the following ODE:

$$\begin{cases} w'(t) = \phi(t, w), & t \in [0, 2h_t], \\ w(0) = w_0, \end{cases}$$

Suppose the interval $[0, 2h_t]$ is first divided into a uniform mesh with a smaller step size $\widehat{h}_t$. The step size $\widehat{h}_t$ is selected in accordance with the convergence order of the chosen BVM. For instance, if a fourth order BVM is used, then we can simply set $\widehat{h}_t \leq h_t^2$ because the BDF2 is second order accurate. The BDF2 approximations are as follows. For the first step, we employ a fully implicit scheme to approximate $w(\widehat{h}_t)$. For the remaining steps, we have the following three-steps algorithm for $t = \widehat{h}_t, 2\widehat{h}_t, \ldots$:

$$\frac{1}{2\widehat{h}_t}\left[3w(t+\widehat{h}_t) - 4w(t) + w(t-\widehat{h}_t)\right] = \phi\big(t+\widehat{h}_t, w(t+\widehat{h}_t)\big).$$

According to the above relations, we apply the BDF2 to (2.11) on a uniform mesh

$$\{0, \widehat{h}_t, 2\widehat{h}_t, \ldots, 2h_t\},$$

and the related linear system can be quickly solved by the iterative method in [1] or the fixed point iteration in [15]. Eventually, the BDF2 solution on $2h_t$ is reckoned as the renewed initial condition and the BVM is implemented on the uniform grid

$$\{2h_t, 3h_t, 4h_t, \ldots, T\},$$

instead. Note that all the items in the BVM are now linked with the number $s-1$ since we have already moved two steps forward.

4 GMRES method with Strang-type preconditioners

4.1 Strang-type circulant preconditioner

Though BVMs are known for their stability properties, the large size of the resulting system (3.7) makes direct methods numerically expensive. Consider the matrix (3.8) in system (3.7), the components A_t and B_t are Toeplitz matrices plus small rank perturbation. Hence preconditioned iterative methods are favorable [3, 8, 16, 17]. Instead of solving (3.6) directly, we consider the following preconditioned system:

$$P^{-1}M\overline{\mathbf{v}} = P^{-1}\mathbf{q}, \tag{4.1}$$

where P is called the preconditioner of M. If P is set accordingly, then we can accelerate the convergence rate of GMRES method [21] and solve (4.1) efficiently. In particular, P can be chosen as a circulant-type matrix as in [3, 8]. In this paper, we apply a Strang-type preconditioner, which is similar to the one in [8], for solving (3.7):

$$P = s(A_t) \otimes \widehat{B}_x - h_t s(B_t) \otimes \widehat{A}, \tag{4.2}$$

where $s(A_t)$, $s(B_t) \in \mathbb{R}^{(s-1)\times(s-1)}$ are the generalized Strang-type preconditioners of A_t and B_t [7, 9]. More specifically, $s(A_t)$ and $s(B_t)$ are circulant matrices generated by column vectors $[\alpha_\nu, \ldots, \alpha_0, 0, \ldots, 0, \alpha_\mu, \ldots, \alpha_{\nu+1}]^\intercal$ and $[\beta_\nu, \ldots, \beta_0, 0, \ldots, 0, \beta_\mu, \ldots, \beta_{\nu+1}]^\intercal$ respectively. Here α_l and β_l for $l = 1, 2, \ldots, \mu$ are the coefficients in the BVM formula (3.2).

To study the invertibility of the preconditioner P, we first introduce the two characteristic polynomials related to the given BVM (3.2) on the complex plane $\mathbb{C}$:

$$\rho(z) = z^\nu \sum_{l=-\nu}^{\mu-\nu} \alpha_{l+\nu} z^l, \qquad \text{and} \qquad \varsigma(z) = z^\nu \sum_{l=-\nu}^{\mu-\nu} \beta_{l+\nu} z^l.$$

Define

$$\mathcal{D}_{\nu,\mu-\nu} = \{\omega \in \mathbb{C} : \rho(z) - \omega\varsigma(z) \text{ has } \nu \text{ zeros in } |z| < 1 \text{ and } \mu - \nu \text{ zeros in } |z| > 1\}.$$

Then a BVM is said to be $A_{\nu,\mu-\nu}$-stable if $\mathbb{C}^- \subseteq \mathcal{D}_{\nu,\mu-\nu}$ [5], where $\mathbb{C}^-$ is the negative half of the complex plane. The following theorem is a straightforward result of the Corollary 1 in [8].

Theorem 4.1. [8] *If the chosen BVM is $A_{\upsilon,\mu-\upsilon}$-stable and the eigenvalues of $\widehat{B}_x^{-1}\widehat{A}$ are in the negative half of the complex plane, then the preconditioner P* (4.2) *is invertible.*

4.2 Operation cost of preconditioned system

In this section, we will study the operation cost of the preconditioned system (4.1). Recall that $\widehat{B}_x$, $\widehat{A}$ are n-by-n tridiagonal-like matrices, $\widehat{A}_x$ is a combination of $\widehat{A}$, $\lambda\widehat{B}_x\widehat{T}_f$

and a rank-1 matrix, and the two numbers m, n are about the same. Both A_t, B_t are $(s-1)$-by-$(s-1)$ banded matrices and $s(A_t)$, $s(B_t)$ are their circulant preconditioners respectively. The main work of the GMRES method in each iteration is the following matrix-vector product [21]:

$$P^{-1}M\psi = (s(A_t)\otimes\widehat{B}_x - h_t s(B_t)\otimes\widehat{A})^{-1}(A_t\otimes\widehat{B}_x - h_t B_t\otimes\widehat{A}_x)\psi,$$

where P is the preconditioner (4.2), M is the coefficient matrix (3.8) and ψ is a given vector. We first consider the latter part:

$$M\psi = (A_t\otimes\widehat{B}_x - h_t B_t\otimes\widehat{A}_x)\psi.$$

Since A_t and B_t are banded, $\widehat{B}_x$ is tridiagonal-like, these matrices do not cause too much trouble when they are involved in matrix-vector multiplications. Note that the full matrix $\widehat{A}_x$ is mainly composed of $\widehat{T}_f$ and the Toeplitz-like matrix T_f is a major sub-matrix of $\widehat{T}_f$. It is well known that the matrix-vector product for Toeplitz-like matrices can be calculated by FFTs [7, 9]. Totally speaking, the matrix-vector product $M\psi$ is obtained in $\mathcal{O}(sm\log m)$ operations.

To get $P^{-1}(M\psi)$, we first diagonalize the circulant matrices $s(A_t)$ and $s(B_t)$ by the Fourier matrix F [7,9]:

$$s(A_t) = F\Lambda_{A_t}F^*, \qquad \text{and} \qquad s(B_t) = F\Lambda_{B_t}F^*,$$

where Λ_{A_t} and Λ_{B_t} are $(s-1)$-by-$(s-1)$ diagonal matrices holding the eigenvalues of $s(A_t)$ and $s(B_t)$ respectively. It follows that

$$P^{-1}(M\psi) = (F^*\otimes I_n)(\Lambda_{A_t}\otimes\widehat{B}_x - h_t\Lambda_{B_t}\otimes\widehat{A})^{-1}(F\otimes I_n)(M\psi).$$

Note that F is an $(s-1)$-by-$(s-1)$ Fourier matrix, therefore $(F\otimes I_n)(M\psi)$ is a direct outcome of FFTs with $\mathcal{O}(ms\log s)$ complexity. The product for $(F^*\otimes I_n)$ is a similar case. As $\widehat{B}_x$, $\widehat{A}$ are tridiagonal-like, the central matrix

$$\Lambda_t = \Lambda_{A_t}\otimes\widehat{B}_x - h_t\Lambda_{B_t}\otimes\widehat{A},$$

is a large sparse matrix and consists of $s-1$ smaller systems with size n. Note that all these sub-systems are tridiagonal-like, they can be solved effortlessly and the related operation cost can be omitted. In conclusion, the operation cost of the product $P^{-1}(M\psi)$ is of order

$$\mathcal{O}(sm\log m) + \mathcal{O}(ms\log s) = \mathcal{O}(ms\log ms).$$

5 Numerical results

In the following experiments, we utilize two special families of BVMs, known as the Extended Trapezoidal Rules of first kind (ETR) and second kind (ETR_2). The ETR

and ETR_2 are families of symmetric schemes and the generalization of the trapezoidal rules. Such methods are noted for having good stability properties [5,6]. To match up the FOC scheme in spatial direction, here we prefer fourth order ETR and ETR_2 with μ=3 and ν=2. The associated equations for initial and final conditions can be found in [5]. After selecting the fourth order BVM, we can specify how the BDF2 startup procedure is to be used in our case. For the sake of fourth order convergence, we replace the approximations of the first two time steps by using BDF2 with step size h_t^2 as discussed before.

We regard the analytical formula of pricing European call options for Merton's model [19] as the true solution. For the GMRES method, the initial guess is the zero vector and the stopping criterion is

$$\frac{\|\mathbf{r}_k\|_2}{\|\mathbf{r}_0\|_2} < 10^{-14},$$

where $\mathbf{r}_k$ is the residual after k iterations, see [21] for details. We remark that the true solution increases exponentially along the spatial direction and hence a relatively small tolerance is needed to lower the errors.

The choice of truncation points $x_{\min}$ and $x_{\max}$ is also crucial. As mentioned before, $x_{\min}$ and $x_{\max}$ should be adjusted properly such that $x^* = 0$ belongs to the spatial computational grid. Moreover, we need to make sure x^* is the common boundary point of two integral subdomains, i.e., the term $v(x^*,t)f(x^*-x)$ should hold the coefficient 2 of the composite Simpson's rule. We can achieve this by using asymptotic grid points outside of the computational domain when we calculate the integral.

We first try out the FOCBVM with ETR_2, and compare the iteration numbers required for convergence of the preconditioned GMRES method with the unpreconditioned one. The first set of input parameters are

$$T = 0.25, \quad K = 100, \quad \sigma = 0.25, \quad r = 0.05, \quad \lambda = 0.1, \quad \delta = -0.9, \quad \gamma = 0.45.$$

In Fig. 1, we see that the eigenvalues of the matrix $\widehat{B}_x^{-1}\widehat{A}$ for different n are all negative real numbers and bounded away from zero. Since the fourth order ETR_2 is $A_{2,1}$-stable [5,6], according to Theorem 4.1, the preconditioner P is invertible.

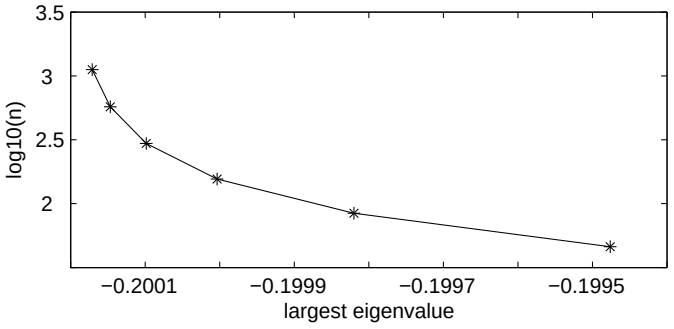

Figure 1: Largest eigenvalues of the matrix $\widehat{B}_x^{-1}\widehat{A}$ for different n.

Table 1: Errors and numbers of iterations of FOCBVM with no preconditioner or Strang-type preconditioner for pricing a European call option under Merton's jump-diffusion model.

n	s	$l^\infty(h_x,h_t)$	order	I_s	P_s
41	5	2.17e-2	-	26	23
73	10	4.62e-3	2.23	74	25
133	20	3.87e-4	3.58	242	27
249	40	2.76e-5	3.81	711	30
509	80	1.88e-6	3.87	>1000	38
1059	160	1.22e-7	3.95	>1000	50

Recall that n is the number of grid nodes in the refined spatial grid minus $x_{\min}$ and $x_{\max}$, and s is the number of time steps. In Table 1, $l^\infty(h_x,h_t)$ is the maximum absolute error between the true solution and the approximation with mesh size h_x and h_t. The "order" is the $\log 2$-ratio of $l^\infty(h_x,h_t)$ to $l^\infty(h_x/2,h_t/2)$. Since the FOCBVM is now fourth order accurate in both directions, we simultaneously halve h_x and h_t after each approximation and the convergence order should gradually approach 4. The term "I_s" denotes the number of iterations of the GMRES method without preconditioning, and ">1000" means that the scheme needs more than 1000 iterations to converge. Similarly, "P_s" denotes the number of iterations of the GMRES method with Strang-type preconditioner. We observe fourth order convergence from the numerical results, though the ordinary GMRES method demands a dramatic number of iterations to converge as the grid is refined. On the other hand, the preconditioned GMRES method needs far fewer iterations for convergence.

We would also like to compare the FOCBVM with other existing efficient methods, e.g., the Crank-Nicolson (CN) scheme. It is shown in [15] that the CN scheme with Rannacher timestepping procedure obtains second order accuracy, and systems at each time step can be rapidly solved by fixed point iteration. In Table 2, the column "CPU" displays the CPU time in seconds, and the other columns are analogous to Table 1. Though the FOCBVM is more costly to carry out than the CN scheme, it can

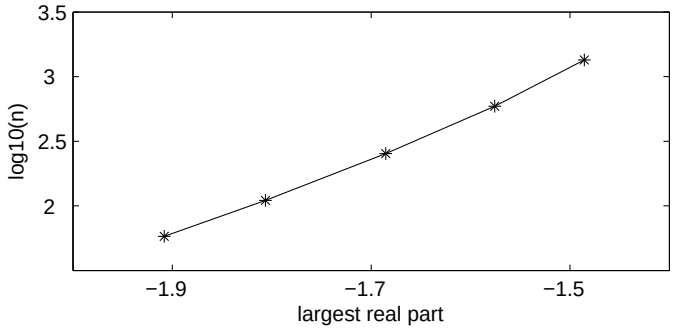

Figure 2: Largest real parts of eigenvalues of the matrix $\widehat{B}_x^{-1}\widehat{A}$ for different n.

Table 2: Errors and CPU time of CN scheme and FOCBVM.

	CN				FOCBVM			
s	m	$l^\infty(h_x,h_t)$	order	CPU	n	$l^\infty(h_x,h_t)$	order	CPU
5	31	2.08e+0	-	0.01	41	2.17e-2	-	0.07
10	63	7.40e-1	1.49	0.02	73	4.62e-3	2.23	0.12
20	127	1.57e-1	2.24	0.02	133	3.87e-4	3.58	0.30
40	255	3.74e-2	2.07	0.05	249	2.76e-5	3.81	0.97
80	511	9.27e-3	2.01	0.15	-	-	-	-
160	1023	2.31e-3	2.00	0.58	-	-	-	-
320	2047	5.78e-4	2.00	5.04	-	-	-	-

Table 3: Errors, numbers of iterations and CPU time of FOCBVM with Strang-type preconditioner for pricing a European call option under Merton's jump-diffusion model.

n	s	$l^\infty(h_x,h_t)$	order	P_s	CPU
41	10	3.24e-4	-	14	0.08
89	20	4.46e-5	2.86	17	0.15
229	40	3.50e-6	3.67	22	0.71
561	80	2.70e-7	3.69	27	3.88
1313	160	2.04e-8	3.73	50	44.53

reach the same error level as the second order scheme with fewer grid points and less time. In such sense, the FOCBVM is more effective than the CN scheme. For example, if a solution with maximum absolute error around 10^{-4} is wanted, the CN scheme finishes the job with m=2047 in about 5 seconds. However, the FOCBVM can do the same thing with only n=133 in 0.3 seconds.

At last we will test another set of parameters, which will show the essence of our proposed method. This time we choose the fourth order ETR as the BVM for time direction, and the input parameters are

$$T = 0.5, \quad K = 1, \quad \sigma = 0.1, \quad r = 0, \quad \lambda = 1, \quad \delta = 0, \quad \gamma = 0.5.$$

In Fig. 2, we see that the eigenvalues of the matrix $\widehat{B}_x^{-1}\widehat{A}$ for different n all lie in the negative half of complex plane. Since the fourth order ETR is also $A_{2,1}$-stable [5, 6], according to Theorem 4.1, the preconditioner P is invertible.

In Table 3, all the column tags are similar to the ones in the previous tables. We see that the FOCBVM still achieves fourth order convergence for both directions.

6 Concluding remarks

In this paper, we have shown that the FOCBVM with local mesh refinement and BDF2 startup procedure achieves high order convergence in both spatial and temporal directions for solving a PIDE, and the preconditioned GMRES method outperforms the

original one. We once again remark that the constant coefficient assumption is only for simplicity. In fact, the spatial operators will still be sparse matrices after applying an FOC scheme for variable coefficients. Therefore, the total operation cost is pretty much the same after all. Also, we note that inaccuracies mostly appear at the strike price and both ends of the spatial domain. Hence, we should concentrate grid nodes in these regions, especially when we are pricing other discontinuous options or maturity time T gets larger. For the BDF2 startup procedure, it is of great interest to optimize the number of initial time steps required for convergence.

By recalling the construction of the Strang-type preconditioner (4.2), we chose the tridiagonal-like matrix $\widehat{A}$ over full matrix $\widehat{A}_x$ because of the resulting sparse subsystems. Suppose we embed $\widehat{A}_x$, which contains a Toeplitz-like submatrix, in the preconditioner. Then all the subsystems after Fourier diagonalization will turn out to be Toeplitz-like, hence it is possible to create another preconditioner for these subsystems. In our numerical tests, we also notice that the iteration number of the preconditioned GMRES method steadily increases as we halve the mesh size in spatial direction. Therefore, for future work, we would consider other preconditioners for (3.7) so that we can further reduce the number of iterations and CPU time. We would also like to price different types of options and extend our method to advanced financial models.

Acknowledgments

The research was supported by the research grant UL020/08-Y2/MAT/JXQ01/FST, RG063/08-09S/SHW/FST from University of Macau, and the research grant from FDCT of Macao.

References

[1] A. ALMENDRAL AND C. OOSTERLEE, *Numerical valuation of options with jumps in the underlying*, Appl. Numer. Math., 53 (2005), pp. 1–18.
[2] L. ANDERSEN AND J. ANDEASEN, *Jump-diffusion processes: volatility smile fitting and numerical methods for option pricing*, Rev. Deriv. Res., 4 (2000), pp. 231–262.
[3] D. BERTACCINI, *A circulant preconditioner for the systems of LMF-based ODE codes*, SIAM J. Sci. Comput., 22 (2000), pp. 767–786.
[4] F. BLACK AND M. SCHOLES, *The pricing of options and corporate liabilities*, J. Polit. Economy, 81 (1973), pp. 637–654.
[5] L. BRUGNANO AND D. TRIGIANTE, Solving Differential Problems by Multistep Initial and Boundary Value Methods, Gordan and Berach, Amsterdam, 1998.
[6] L. BRUGNANO AND D. TRIGIANTE, *Stability properties of some BVM methods*, Appl. Numer. Math., 13 (1993), pp. 291–304.
[7] R. CHAN AND M. NG, *Conjugate gradient methods for Toeplitz systems*, SIAM Rev., 38 (1996), pp. 427–482.
[8] R. CHAN, M. NG AND X. JIN, *Strang-type preconditioners for systems of LMF-based ODE codes*, IMA J. Numer. Anal., 21 (2001), pp. 451–462.

[9] R. CHAN AND X. JIN, An Introduction to Iterative Toeplitz Solvers, SIAM, Philadelphia, 2007.
[10] R. CONT AND P. TANKOV, Financial Modeling with Jump Processes, Chapman & Hall/CRC Press, 2004.
[11] R. CONT AND E. VOLTCHKOVA, *A finite difference scheme for option pricing in jump diffusion and exponential Lévy models*, SIAM J. Numer. Anal., 43 (2005), pp. 1596–1626.
[12] L. FENG AND V. LINETSKY, *Pricing options in jump-diffusion models: an extrapolation approach*, Oper. Res., 56 (2008), pp. 304–325.
[13] M. GILES AND R. CARTER, *Convergence analysis of Crank-Nicolson and Rannacher time-marching*, J. Comput. Fin., 9 (2006), pp. 89–112.
[14] M. GUPTA, *A fourth-order poisson solver*, J. Comput. Phys., 55 (1984), pp. 166–172.
[15] Y. D'HALLUIN, P. FORSYTH AND K. VETZAL, *Robust numerical methods for contingent claims under jump diffusion processes*, IMA J. Numer. Anal., 25 (2005), pp. 87–112.
[16] X. JIN, S. LEI AND Y. WEI, *Circulant preconditioners for solving differential equations with multi-delays*, Comput. Math. Appl., 47 (2004), pp. 1429–1436.
[17] X. JIN, S. LEI AND Y. WEI, *Circulant preconditioners for solving singular perturbation delay differential equations*, Numer. Linear Algebra Appl., 12 (2005), pp. 327–336.
[18] S. LEE AND H. SUN, *Fourth order compact scheme with local mesh refinement for option pricing in jump-diffusion model*, unpublished.
[19] R. MERTON, *Option pricing when underlying stock returns are discontinuous*, J. Financ. Eco., 3 (1976), pp. 125–144.
[20] R. RANNACHER, *Finite element solution of diffusion problems with irregular data*, Numer. Math., 43 (1984), pp.309–327.
[21] Y. SAAD AND M. SCHULTZ, *GMRES: a generalized minimal residual algorithm for solving non-symmetric linear systems*, SIAM J. Sci. Stat. Comput., 7 (1986), pp. 856–869.
[22] W. SPOTZ AND G. CAREY, *High-order compact scheme for the steady stream-function vorticity equations*, Internat. J. Numer. Methods Engrg., 38 (1995), pp. 3497–3512.
[23] H. SUN AND J. ZHANG, *A high-order compact boundary value method for solving one-dimensional heat equations*, Numer. Methods Partial Differential Equations, 19 (2003), pp. 846–857.
[24] D. TAVELLA AND C. RANDALL, Pricing Financial Instruments: The Finite Difference Method, John Wiley and Sons, New York, 2000.
[25] J. ZHANG, H. SUN AND J. ZHAO, *High order compact scheme with multigrid local mesh refinement procedure for convection diffusion problems*, Comput. Methods Appl. Mech. Engrg., 191 (2002), pp. 4661–4674.