Robust Numerical Valuation of European and American Options under the CGMY Process

Iris R. Wang * Justin W.L. Wan † Peter A. Forsyth ‡

May 8, 2007

Abstract

We develop an implicit discretization method for pricing European and American options when the underlying asset is driven by an infinite activity Lévy process. For processes of finite variation, quadratic convergence is obtained as the mesh and time step are refined. For infinite variation processes, better than first order accuracy is achieved. The jump component in the neighborhood of log jump size zero is specially treated by using a Taylor expansion approximation and the drift term is dealt with using a semi-Lagrangian scheme. The resulting Partial Integro-Differential Equation (PIDE) is then solved using a preconditioned BiCGSTAB method coupled with a fast Fourier transform. Proofs of fully implicit timestepping stability and monotonicity are provided. The convergence properties of the BiCGSTAB scheme are discussed and compared with a fixed point iteration. Numerical tests showing the convergence and performance of this method for European and American options under processes of finite and infinite variation are presented.

Keywords: CGMY, semi-Lagrangian, implicit timestepping

1 Introduction

It is well known that the standard Geometric Brownian Motion model for asset price returns is inconsistent with market prices. Models with jump processes are thought to be more representative of actual market behavior [1]. Recently, Lévy process models have become popular in the financial literature [1, 2, 3, 4, 5, 6, 7]. Option pricing, under exponential Lévy process with finite activity [5, 8, 9, 10, 11, 12] and infinite activity [13, 14, 15, 16, 17] has been extensively studied. In these papers, various numerical methods were proposed for solving the option pricing Partial Integro-Differential Equation (PIDE).

In this paper, we are specifically concerned with numerical methods for the infinite activity case. In the variance gamma case, a partially implicit method was suggested in [11], with the jump integral part split into a local and a non-local part. In [14], the jump integral part was

*David R. Cheriton School of Computer Science, University of Waterloo, Waterloo ON, Canada N2L 3G1 e-mail: r5wang@cs.uwaterloo.ca

†David R. Cheriton School of Computer Science, University of Waterloo, Waterloo ON, Canada N2L 3G1 e-mail: jwlwan@cs.uwaterloo.ca

‡David R. Cheriton School of Computer Science, University of Waterloo, Waterloo ON, Canada N2L 3G1 e-mail: paforsyt@uwaterloo.ca

also split into local and nonlocal parts with the local term computed using implicit timestepping and the nonlocal term computed using explicit timestepping. This method produced first order accuracy for the variance gamma case, but accuracy degrades for other processes. In [13, 18], an integration by parts technique was used to transform the integral part into a weakly singular Volterra equation. A collocation method was then used to achieve second order accuracy under a CGMY (Carr-Geman-Madan-Yor) process [2] with infinite activity and finite variation.

In this paper, we propose an efficient numerical scheme for pricing European and American options under the CGMY process [2] with not only finite variation but also infinite variation. Variance gamma process can be considered as a special case of the Lévy process which has infinite activity but finite variation. As the jump process moves from variance gamma to CGMY with infinite activity and infinite variation, the Lévy measure has an infinite first moment and thus behaves more like a diffusion process, which is more numerically challenging. We capture the diffusion behavior by approximating the component of small sized jumps with a diffusion term. Furthermore, by using an implicit timestepping method for both local and nonlocal terms of the integral term, we can achieve second order accuracy under a finite variation process and better than a first order convergence rate for an infinite variation process.

In [16], a Wavelet-Galerkin finite element approach is used to price options under an infinite activity process. Compared with [16], our approach can be easily implemented in existing option pricing software. We use only finite difference discretization methods and standard sparse matrix solvers. As well, our techniques can be easily generalized to handle American early exercise, and other path-dependent contract features.

Moreover, our fully implicit discretization leads to a monotone scheme. Monotonicity is an important property of a discretization method [19], which can be used to guarantee convergence to the viscosity solution. Although we do not exploit this property of our scheme in this paper, we anticipate that this property will prove to be useful for handling nonlinear pricing problems, such as optimal stochastic control, under an infinite activity jump process.

Since we use an implicit timestepping method, a straightforward approach would require a dense matrix solve at each time step due to the jump integral term. We avoid this problem by using either a fixed point iteration [12] or a BiCGSTAB [20] method for solving the discretized equations. This technique requires only a matrix-vector multiply, which can be conveniently carried out using an FFT [13, 14, 12, 21].

The outline of this paper is as follows. Section 2 gives the specification of the option pricing PIDE and boundary conditions. In section 3 we first transform the original PIDE by applying a Taylor expansion. The singular and nonsingular parts of the jump integral term are treated separately. In the case of a pure jump process, there is no diffusion term in the original PIDE. To avoid poor convergence due to first order upstream methods, we use a semi-Lagrangian discretization scheme.

In section 4 we extend our numerical scheme to price American options using a penalty method. In section 5 we provide stability and monotonicity analysis for fully implicit timestepping. A convergence comparison between a fixed point iteration and a preconditioned BiCGSTAB method is also presented. For the special case of the PIDE written in a $\log S$ coordinates, we can show stability of Crank-Nicolson timestepping using a Von Neumann analysis (See Appendix B). In section 6, a series of the numerical tests are carried out showing the convergence and the performance of pricing European and American options under CGMY, for both infinite and finite variation processes.

2 Mathematical Model

In this section, we give the mathematical model for pricing European options when the underlying asset price is modelled by a CGMY process.

2.1 The PIDE for Option Pricing

Let S denote the underlying risky asset price. We model the evolution of S driven by a Lévy process whose Lévy measure ν satisfies

$$\int_{|y|<1} y^2 \nu(dy) < \infty, \qquad \int_{|y|\geq 1} \nu(dy) < \infty. \tag{2.1}$$

Under the CGMY model, ν is defined as follows

$$\nu(y) = \frac{Ce^{-My}}{y^{1+Y}} 1_{y>0} + \frac{Ce^{-G|y|}}{|y|^{1+Y}} 1_{y<0}, \tag{2.2}$$

where

$$1_{y>0} = \begin{cases} 1, & \text{if } y > 0 \\ 0, & \text{otherwise,} \end{cases} \qquad 1_{y<0} = \begin{cases} 1, & \text{if } y < 0 \\ 0, & \text{otherwise,} \end{cases}$$

are the indicator variables; $C > 0$ is the measure of the overall level of activity; $G \geq 0$ and $M \geq 0$ control the rate of exponential decay on the left and right of the Lévy density; and $Y < 2$ describes the behavior of the Lévy density in the neighborhood of zero where the density tends to infinity.

If $Y < 0$, the measure ν integrates to a finite value yielding a process of finite activity. If $Y \in [0, 1]$, the process displays infinite activity but finite variation since $\int_{|y|<1} y\nu(dy) < \infty$. If $Y \in (1, 2)$, the process is said to have infinite activity and infinite variation.

Define $\tau = T - t$, where T is the expiry time, and t is the forward time. Let $V(S, \tau)$ be the option price with the underlying stock price S. Following standard methods [1], we can derive the following PIDE for the value of a contingent claim $V(S, \tau)$:

$$V_\tau = \frac{\sigma^2}{2} S^2 V_{SS} + (r - q) S V_S - rV + \int_{-\infty}^{\infty} \nu(y)[V(Se^y, \tau) - V(S, \tau) - S(e^y - 1)V_S]dy, \tag{2.3}$$

where r and q are the risk-free interest rate and the continuous dividend yield respectively. The parameter σ is the volatility associated with the continuous component of Lévy process.

2.2 Boundary Conditions

As $S \to 0$, equation (2.3) reduces to

$$V_\tau = -rV.$$

As $S \to \infty$, we make the common assumption that $V_{SS} \simeq 0$, which means that

$$V \simeq A(\tau)S + B(\tau), \qquad S \to \infty. \tag{2.4}$$

Assuming equation (2.4) holds, then equation (2.3) reduces to the PDE:

$$V_\tau = \frac{\sigma^2}{2} S^2 V_{SS} + (r - q) S V_S - rV, \qquad S \to \infty,$$

in which we retain the V_{SS} term for numerical stability purposes. In the numerical computation, the original infinite domain, $S \in [0, \infty)$, is truncated to a finite computational domain, $[0, S_{\max}]$. Note that at $S = S_{\max}$, we impose the Dirichlet boundary condition:

$$V(S_{\max}, \tau) = \begin{cases} \simeq S_{\max}, & \text{for a call;} \\ \simeq 0, & \text{for a put,} \end{cases}$$

which is simply the payoff of the option, as $S \to \infty$.

To summarize, we solve the following problem on $[0, S_{\max}]$

$$V_\tau = \begin{cases} -rV, & \text{if } S = 0 \\ \frac{\sigma^2}{2} S^2 V_{SS} + (r-q) S V_S - rV + \int_{-\infty}^{y^*(S)} \nu(y)[V(Se^y) - V(S) - S(e^y - 1)V_S]dy, & \text{if } S \in (0, S^*) \\ \frac{\sigma^2}{2} S^2 V_{SS} + (r-q) S V_S - rV, & \text{if } S \in [S^*, S_{\max}) \end{cases}$$

with the initial condition,

$$V(S, \tau = 0) \quad = \quad f(S),$$

where f is a payoff function. For example, for a vanilla call/put with strike price K,

$$f(S) = \begin{cases} \max(S - K, 0), & \text{for a call;} \\ \max(K - S, 0), & \text{for a put.} \end{cases}$$

We choose S^* sufficiently large so that equation (2.4) is valid in $[S^*, S_{\max}]$, and

$$y^*(S) = \log\left(\frac{S_{\max}}{S}\right).$$

Here $S_{\max}$ is chosen such that

$$\left| \nu\left(\log\left(\frac{S_{\max}}{S^*} \right) \right) S_{\max} \right| < \varepsilon, \tag{2.5}$$

where ε is selected so that the error in approximating the integral is small [12]. If an unequally spaced grid is used, it is inexpensive to select large values for $S_{\max}, S^*$.

3 Discretization Methods

In this section, we first provide the details of discretizing the integral term. The application of a semi-Lagrangian discretization method on the transformed PIDE is then discussed, coupled with a fully implicit or a Crank-Nicolson timestepping scheme. The resulting linear system is then solved by the appropriate iterative method.

3.1 Discretization of Jump Component

The discretization of the jump integral term in (2.3),

$$\mathcal{I}(V) = \int_{-\infty}^{y^*(S)} \nu(y)[V(Se^y, \tau) - V(S, \tau) - S(e^y - 1)V_S]dy, \tag{3.1}$$

is carried out on a finite computational domain. To be specific, we approximate $\mathcal{I}(V)$ by

$$\mathcal{I}_A(V) = \int_{y_{\min}}^{y_{\max}} \nu(y)[V(Se^y,\tau) - V(S,\tau) - S(e^y-1)V_S]dy. \tag{3.2}$$

We divide the interval $[y_{\min}, y_{\max}]$ into subintervals $[y_j - \Delta y/2, y_j + \Delta y/2]$, $j = 0, 1, \ldots, N-1$, with

$$y_j = y_{\min} + (2j+1)\frac{\Delta y}{2}, \quad \text{and} \quad \Delta y = \frac{y_{\max} - y_{\min}}{N},$$

where $y_{\max}$ and $y_{\min}$ are selected such that the error in approximating $\mathcal{I}(V)$ by $\mathcal{I}_A(V)$ can be made arbitrarily small.

The properties of the Lévy measure (2.1) indicate that $\nu(y)$ goes to infinity in the neighborhood of $y = 0$ and is only second moment integrable for all $|y| < 1$. Accordingly, we define

$$\Omega_0 = \left\{ y \middle| -\frac{\Delta y}{2} \le y \le \frac{\Delta y}{2} \right\}, \qquad \Omega_1 = \left\{ y \middle| \frac{\Delta y}{2} < |y| < 1 \right\}, \tag{3.3}$$
$$\text{and } \Omega_2 = \left\{ y \middle| y_{\min} \le y \le -1 \text{ or } 1 \le y \le y_{\max} \right\},$$

where Ω_0 refers to the infinite density region, i.e. $\nu(y) \to \infty$ when $y \to 0$; Ω_1 corresponds to the singular part of the Lévy measure excluding Ω_0; and Ω_2 refers to the smooth integrable part of $\nu(y)$. Note that the choice of $|y| = 1$ as the boundary defining Ω_2 is somewhat arbitrary. Other choices are possible, but numerically this choice works well. In the following, we will treat the integral differently according to the region over which the integral is computed.

For the integral over Ω_0, we use a similar approach as in [14] in approximating the integral of $\nu(y)$ by a finite process with an effective diffusion coefficient. Note that Se^y is the asset price after a jump, i.e. $Se^y = S + \delta S$. We can apply a Taylor expansion to $V(Se^y)$ so that

$$\begin{aligned}
&\int_{-\frac{\Delta y}{2}}^{\frac{\Delta y}{2}} \nu(y)[V(Se^y,\tau) - V(S,\tau) - S(e^y-1)V_S]dy \\
&= \int_{-\frac{\Delta y}{2}}^{\frac{\Delta y}{2}} \nu(y)[V(S,\tau) + V_S(S,\tau)\delta S + V_{SS}(S,\tau)\frac{(\delta S)^2}{2} - V(S,\tau) - \delta S V_S + O((\delta S)^3)]dy \\
&= \int_{-\frac{\Delta y}{2}}^{\frac{\Delta y}{2}} \nu(y)[V_{SS}(S,\tau)\frac{S^2(e^y-1)^2}{2} + O((\delta S)^3)]dy.
\end{aligned} \tag{3.4}$$

Note that if we carry out the Taylor expansion on e^y in the term, $\delta S = S(e^y - 1)$, we can see that δS behaves as $O(|y|)$ when the value of $|y|$ is small. The definition of the Lévy density (2.2) indicates that $\nu(y)$ behaves as $O\big(\frac{1}{|y|^{1+Y}}\big)$ in the neighborhood of zero. Thus equation (3.4) can be rewritten as

$$\int_{-\frac{\Delta y}{2}}^{\frac{\Delta y}{2}} \nu(y)[V_{SS}(S,t)\frac{S^2(e^y-1)^2}{2} + O(|y|^3)]dy = \frac{S^2}{2}V_{SS}\int_{-\frac{\Delta y}{2}}^{\frac{\Delta y}{2}} \nu(y)(e^y-1)^2 dy + O((\Delta y)^{3-Y}). \tag{3.5}$$

The error term, $O((\Delta y)^{3-Y})$, implies that we can anticipate quadratic convergence if $Y \le 1$, but when $1 < Y < 2$ the best achievable convergence rate is between first and second order, using the approximation (3.5).

Let $\bar{\sigma}$ denote the effective diffusion component in (3.5),

$$\bar{\sigma}(\Delta y) = \int_{-\frac{\Delta y}{2}}^{\frac{\Delta y}{2}} \nu(y)(e^y - 1)^2 dy.$$

The jump integral term $\mathcal{I}_A(V)$ (3.2) thus becomes

$$\begin{aligned}
\mathcal{I}_A(V) &= \frac{\bar{\sigma}}{2} S^2 V_{SS} + \sum_{j:y_j \in \Omega_1 \cup \Omega_2} \int_{y_j - \frac{\Delta y}{2}}^{y_j + \frac{\Delta y}{2}} \nu(y)[V(Se^y) - V(S) - S(e^y - 1)V_S] dy + O((\Delta y)^{3-Y}) \\
&= \frac{\bar{\sigma}}{2} S^2 V_{SS} + \sum_{j:y_j \in \Omega_1} \int_{y_j - \frac{\Delta y}{2}}^{y_j + \frac{\Delta y}{2}} \nu(y) F(y) dy + \sum_{j:y_j \in \Omega_2} \int_{y_j - \frac{\Delta y}{2}}^{y_j + \frac{\Delta y}{2}} \nu(y) F(y) dy + O((\Delta y)^{3-Y}),
\end{aligned} \tag{3.6}$$

where we define

$$F(y) \equiv V(Se^y) - V(S) - S(e^y - 1)V_S.$$

Note that by Taylor expansion, $F(y) = O(y^2)$ as $y \to 0$ (assuming $V(S)$ is a smooth function).

Next we consider the integrals over Ω_1 and Ω_2 respectively. For $y \in \Omega_2$, the Lévy density $\nu(y)$ is a smooth function. As a result, we can carry out a Taylor expansion on $F(y)$ near y_j for $y \in [y_j - \Delta y/2, y_j + \Delta y/2]$, so that the integral term over Ω_2 can be computed using a trapezoidal rule.

$$\begin{aligned}
&\sum_{j:y_j \in \Omega_2} \int_{y_j - \Delta y/2}^{y_j + \Delta y/2} \nu(y)[V(Se^y) - V(S) - S(e^y - 1)V_S] dy \\
&= \sum_{j:y_j \in \Omega_2} V(Se^{y_j})\gamma(y_j) - V(S) \sum_{j:y_j \in \Omega_2} \gamma(y_j) - SV_S \sum_{j:y_j \in \Omega_2} (e^{y_j} - 1)\gamma(y_j) + O((\Delta y)^2),
\end{aligned}$$

where

$$\gamma(y_j) = \int_{y_j - \Delta y/2}^{y_j + \Delta y/2} \nu(y) dy, \qquad y_j \in \Omega_2.$$

For $y \in \Omega_1$, the Lévy measure $\nu(y)$ is a singular function. We cannot directly carry out a Taylor expansion on the integrand as for the case of $y \in \Omega_2$. By noting that $F(y) = O(y^2)$, $y \to 0$, we define $\underline{F}(y) = F(y)/y^2$ which remains a smooth bounded function as $y \to 0$. Thus we can rewrite the integrand as $\nu(y)F(y) = \nu(y)y^2\underline{F}(y)$. We further define

$$\underline{\nu}(y) = \nu(y)y^2,$$

which is integrable over Ω_1. Then we apply Taylor expansion on $\underline{F}(y)$ in the transformed integrand:

$$\int_{y_j - \Delta y/2}^{y_j + \Delta y/2} \underline{F}(y)\underline{\nu}(y) dy = \int_{y_j - \Delta y/2}^{y_j + \Delta y/2} \underline{F}(y_j)\underline{\nu}(y) dy + E_j, \tag{3.7}$$

where the error term is

$$E_j = \int_{y_j - \Delta y/2}^{y_j + \Delta y/2} [\underline{F}'(y_j)(y - y_j) + \frac{\underline{F}''(\xi_j(y))}{2}(y - y_j)^2]\underline{\nu}(y) dy$$

in which $\xi_j(y) \in (y_j - \Delta y/2, y_j + \Delta y/2)$. It can be shown that (see Appendix A):

$$E_{\Omega_1} \equiv \sum_j E_j = O((\Delta y)^{\min(2-\epsilon, 3-Y)}),$$

where ϵ is an arbitrarily small positive constant. The integral over Ω_1 is thus computed by

$$\begin{aligned}
&\sum_{j:y_j \in \Omega_1} \int_{y_j - \Delta y/2}^{y_j + \Delta y/2} \underline{\nu}(y) \left(\frac{V(Se^y) - V(S) - S(e^y - 1)V_S}{y^2} \right) dy \\
&= \sum_{j:y_j \in \Omega_1} V(Se^{y_j})\gamma(y_j) - V(S) \sum_{j:y_j \in \Omega_1} \gamma(y_j) - SV_S \sum_{j:y_j \in \Omega_1} (e^{y_j} - 1)\gamma(y_j) + O((\Delta y)^{\min(2-\epsilon, 3-Y)}),
\end{aligned}$$

where

$$\gamma(y_j) = \frac{1}{y_j^2} \int_{y_j - \Delta y/2}^{y_j + \Delta y/2} y^2 \nu(y) dy, \qquad \forall y_j \in \Omega_1.$$

Putting all this together, the integral over $\Omega_1 \cup \Omega_2$ in (3.6) is approximated by

$$\begin{aligned}
&\sum_{j:y_j \in \Omega_1 \cup \Omega_2} \int_{y_j - \frac{\Delta y}{2}}^{y_j + \frac{\Delta y}{2}} \nu(y) F(y) dy = \\
&\sum_{j=0}^{N-1} V(Se^{y_j})\gamma(y_j) - V \sum_{j=0}^{N-1} \gamma(y_j) - SV_S \sum_{j=0}^{N-1} (e^{y_j} - 1)\gamma(y_j) + O((\Delta y)^{\min(2-\epsilon, 3-Y)}),
\end{aligned}$$

where

$$\gamma(y_j) = \begin{cases} \frac{1}{y_j^2} \int_{y_j - \Delta y/2}^{y_j + \Delta y/2} y^2 \nu(y) dy; & \text{if } y_j \in \Omega_1 \\ \int_{y_j - \Delta y/2}^{y_j + \Delta y/2} \nu(y) dy; & \text{if } y_j \in \Omega_2 \\ 0; & \text{if } y_j \in \Omega_0. \end{cases} \tag{3.8}$$

A quadrature rule is used for computing $\gamma(y_j)$ with accuracy (see Appendix B),

$$\gamma(y_j) = \begin{cases} \hat{\gamma}(y_j) + O\big(\frac{(\Delta y)^4}{(y_j)^2}\big) & \text{if } y_j \in \Omega_1 \\ \hat{\gamma}(y_j) + O((\Delta y)^3) & \text{if } y_j \in \Omega_2 \end{cases} \tag{3.9}$$

In summary, we obtain the final approximation for $\mathcal{I}_A(V)$:

$$\mathcal{I}_A(V) = \frac{\bar{\sigma}}{2} S^2 V_{SS} - \kappa S V_S - \lambda V + \sum_{j=0}^{N-1} V(Se^{y_j})\hat{\gamma}(y_j) + O((\Delta y)^{\min(2-\epsilon, 3-Y)}), \tag{3.10}$$

where

$$\lambda = \sum_{j=0}^{N-1} \hat{\gamma}(y_j), \qquad \kappa = \sum_{j=0}^{N-1} (e^{y_j} - 1)\hat{\gamma}(y_j). \tag{3.11}$$

Note that equation (3.10) has the same form as we would obtain for a finite activity Lévy process. As we shall see, this is very convenient, since we can use many of the techniques developed for finite activity processes.

3.2 Semi-Lagrangian Discretization

After the discretization of the jump integral part, the PIDE (2.3) is approximated by

$$V_\tau = \frac{\sigma^2 + \bar{\sigma}}{2} S^2 V_{SS} + (r - q - \kappa) S V_S - (r + \lambda) V + \sum_{j=0}^{N-1} V(S e^{y_j}) \hat{\gamma}(y_j). \tag{3.12}$$

In order to be able to handle the case of a pure jump process $\sigma = 0$, we use a semi-Lagrangian method [22, 23] to discretize the first order term to avoid problems with a drift dominated equation.

First we rewrite the PIDE (3.12) as

$$V_\tau - (r - q - \kappa) S V_S = \mathcal{L}V + \mathcal{Q}V, \tag{3.13}$$

where

$$\mathcal{L}V = \frac{\sigma^2 + \bar{\sigma}}{2} S^2 V_{SS} - (r + \lambda) V, \qquad \mathcal{Q}V = \sum_{j=0}^{N-1} V(S e^{y_j}) \hat{\gamma}(y_j).$$

The Lagrangian derivative, along the trajectory $S(\tau)$, is

$$\frac{DV}{D\tau} = \frac{\partial V}{\partial \tau} + \frac{dS}{d\tau} \frac{\partial V}{\partial S}.$$

Then along the trajectory

$$\frac{dS}{d\tau} = -(r - q - \kappa) S, \tag{3.14}$$

we can rewrite equation (3.13) as

$$\frac{DV}{D\tau} = \mathcal{L}V + \mathcal{Q}V. \tag{3.15}$$

Define discrete asset nodes $[S_1, \ldots, S_{i_{max}}]$ and a set of discrete times τ^n. Let $V_i^n \approx V(S_i, \tau^n)$ be the approximate value of the claim at (S_i, τ^n). Let $S = S(S_i, \tau^{n+1}, \tau)$ be a trajectory satisfying equation (3.14), which passes through the discrete S_i node at $\tau = \tau^{n+1}$. If we trace along the trajectory back to time $\tau = \tau^n$, the departure point of this trajectory, denoted by $S_{i(n+1)}$, will not necessarily coincide with a grid node S_j. In order to determine $S_{i(n+1)}$, we need to solve (3.14) from τ^{n+1} to τ^n. The exact solution of (3.14) is

$$S_{i(n+1)} = S_i e^{(r-q-\kappa)\Delta\tau},$$

where $\Delta\tau = \tau^{n+1} - \tau^n$. Note that in general, if the coefficients of the first order term are non-constant, then we would have to integrate (3.14) numerically [24].

Let $V^n_{i(n+1)}$ denote the option value at $S_{i(n+1)}$ at time τ^n. The value of $V^n_{i(n+1)}$ can be determined by using an interpolation scheme. In order to achieve second order accuracy we use an upwind quadratic interpolation method [22]. Suppose that we need to estimate $S_{i(n+1)}$ located between two S grid nodes, i.e., $S_k \le S^n_{i(n+1)} \le S_{k+1}$ for some k. For quadratic interpolation we need three points. If the coefficient $-(r - q - \kappa)$ in (3.14) is positive, we use $\{S_{k-1}, S_k, S_{k+1}\}$; if the coefficient $-(r - q - \kappa)$ is negative, we use $\{S_k, S_{k+1}, S_{k+2}\}$. The interpolation reduces to linear at the grid boundaries.

Let Φ^{n+1} be the upwind quadratic interpolation operator such that

$$(\Phi^{n+1} V_i^n) = V(S_{i(n+1)}, \tau^n) + \text{ interpolation error.} \tag{3.16}$$

Discretizing equation (3.15) along the characteristic trajectory for different timestepping schemes gives, a fully implicit:

$$\frac{V_i^{n+1} - (\Phi^{n+1} V_i^n)}{\Delta\tau} = (\mathcal{L}V^{n+1})_i + (\mathcal{Q}V^{n+1})_i, \tag{3.17}$$

and a Crank-Nicolson scheme:

$$\frac{V_i^{n+1} - (\Phi^{n+1} V_i^n)}{\Delta\tau} = \frac{1}{2}[(\mathcal{L}V^{n+1})_i + (\mathcal{Q}V^{n+1})_i] + \frac{1}{2}[(\Phi^{n+1}\mathcal{L}V_i^n) + (\Phi^{n+1}\mathcal{Q}V_i^n)]. \tag{3.18}$$

In order to compute $(\mathcal{Q}V^{n+1})_i$ as in

$$(\mathcal{Q}V^{n+1})_i = \sum_{j=0}^{N-1} V(S_i e^{y_j}, \tau^{n+1})\hat{\gamma}(y_j), \tag{3.19}$$

we define

$$\overline{V}(x_j, \tau^{n+1}) = V(e^{x_j}, \tau^{n+1}),$$

where $\{x_j\}$ is a set of equally spaced nodes in $\log S$ space, and where we determine $\overline{V}$ by a linear interpolation operator, T, of the discrete vector V^{n+1}, i.e. $\overline{V}^{n+1} = TV^{n+1}$. We can write (3.19) as

$$(\mathcal{Q}V^{n+1})_i = (T^{-1}\overline{\mathcal{I}}^{n+1})_i + \text{ interpolation error},$$

where

$$\begin{aligned}(\overline{\mathcal{I}}^{n+1})_k &= \sum_j \overline{V}_{k+j}\hat{\gamma}(y_j) \\ &= \sum_j (TV^{n+1})_{k+j}\hat{\gamma}(y_j).\end{aligned}$$

Note that $\sum \overline{V}_{k+j}\hat{\gamma}(y_j)$ is a discrete correlation, hence can be computed using an FFT (see [12] for the details). Now define the operator $\mathcal{B}$ such that

$$(\mathcal{B}V^{n+1})_i = (T^{-1}\overline{\mathcal{I}}^{n+1})_i,$$

then equation (3.19) can be further rewritten as

$$(\mathcal{Q}V^{n+1})_i = (\mathcal{B}V^{n+1})_i + \text{ interpolation error},$$

where the interpolation error is second order. Finally, the discretization of fully implicit timestepping, (3.17), can be rewritten as

$$V_i^{n+1}[1 + (\alpha_i + \beta_i + r + \lambda)\Delta\tau] - \Delta\tau\beta_i V_{i+1}^{n+1} - \Delta\tau\alpha_i V_{i-1}^{n+1} = (\Phi^{n+1}V_i^n) + \Delta\tau(\mathcal{B}V^{n+1})_i, \tag{3.20}$$

where α_i and β_i are given by applying second order central finite differencing to the spatial derivatives,

$$\alpha_i = \frac{(\sigma^2 + \bar{\sigma})S_i^2}{(S_i - S_{i-1})(S_{i+1} - S_{i-1})}, \qquad \beta_i = \frac{(\sigma^2 + \bar{\sigma})S_i^2}{(S_{i+1} - S_i)(S_{i+1} - S_{i-1})}, \tag{3.21}$$

for $i = 2, \ldots, i_{\max} - 1$. When $i = 1$, we set $\alpha_i = \beta_i = 0$ at $S_1 = 0$; and when $i = i_{\max}$, we set $V_{i_{\max}}^{n+1}$ equal to the relevant Dirichlet condition. Note that α_i and β_i are non-negative, which is an important property we will exploit in our analysis in later sections.

Similarly the discretization of Crank-Nicolson timestepping (3.18) can be written as

$$\begin{aligned} & V_i^{n+1}[1 + (\alpha_i + \beta_i + r + \lambda)\frac{\Delta\tau}{2}] - \frac{\Delta\tau}{2}\beta_i V_{i+1}^{n+1} - \frac{\Delta\tau}{2}\alpha_i V_{i-1}^{n+1} \\ =& (\Phi^{n+1}V_i^n)[1 - (\alpha_i + \beta_i + r + \lambda)\frac{\Delta\tau}{2}] + \frac{\Delta\tau}{2}\beta_i(\Phi^{n+1}V_{i+1}^n) + \frac{\Delta\tau}{2}\alpha_i(\Phi^{n+1}V_{i-1}^n) \\ & + \frac{\Delta\tau}{2}(\mathcal{B}V^{n+1})_i + \frac{\Delta\tau}{2}(\Phi^{n+1}(\mathcal{B}V^n)_i), \qquad 1 < i < i_{\max}. \end{aligned} \tag{3.22}$$

3.3 Iterative Methods

As discussed above, we can efficiently approximate the dense matrix-vector multiplication, $(\mathcal{Q}V^{n+1})$, by $(\mathcal{B}V^{n+1})$ using an FFT. However, the implicit timestepping methods (3.20 - 3.22) appear to require solution of a dense matrix at each timestep. We will avoid this difficulty by using iterative methods to solve equations (3.20-3.22). These iterative methods will require only computation of $(\mathcal{B}V^{n+1})$. First, we can write a compact matrix form for fully implicit timestepping and Crank-Nicolson timestepping by introducing a constant θ as follows

$$[I - (1-\theta)\Delta\tau L - (1-\theta)\Delta\tau B]V^{n+1} = \Phi^{n+1}[I + \theta\Delta\tau L + \theta\Delta\tau B]V^n, \tag{3.23}$$

where $\theta = 0$ corresponds to the fully implicit method and $\theta = \frac{1}{2}$ refers to the Crank-Nicolson method; I is the identity matrix; matrix L is defined such that the ith row of the product of L and vector V at the $(n+1)$th time step is

$$[LV^{n+1}]_i = -(\alpha_i + \beta_i + r + \lambda)V_i^{n+1} + \beta_i V_{i+1}^{n+1} + \alpha_i V_{i-1}^{n+1}, \tag{3.24}$$

and matrix B is defined such that the ith row of vector BV at $(n+1)$th time step is

$$[BV^{n+1}]_i = (T^{-1}\overline{\mathcal{I}}^{n+1})_i, \tag{3.25}$$

where, if T is a linear interpolation operator, then matrix B has the following properties [12]

$$\sum_j B_{ij} = \lambda, \qquad 0 \le B_{ij} \le 1, \tag{3.26}$$

and B is dense. However, BV^{n+1} can be computed in $N \log N$ operations using an FFT.

We can rewrite the linear system (3.23) in the form of $AV^{n+1} = b$:

$$\begin{aligned} A &= I - (1-\theta)\Delta\tau L - (1-\theta)\Delta\tau B, \\ b &= \Phi^{n+1}[I + \theta\Delta\tau L + \theta\Delta\tau B]V^n, \end{aligned} \tag{3.27}$$

where A is the coefficient matrix and vector b is a known vector at the nth time step. It is computationally infeasible to factor a large dense matrix at each timestep. One solution to this problem is to rewrite A as the difference of two matrices, $A = M - N$ where

$$M = I - (1-\theta)\Delta\tau L, \tag{3.28}$$

is a tridiagonal matrix and $N = (1-\theta)\Delta\tau B$. Then we solve the original problem $AV^{n+1} = b$ by iteratively solving $M\hat{V}^{k+1} = N\hat{V}^k + b$ for $\hat{V}^{k+1}$ until convergence, where $\hat{V}^k$ is the estimate to the solution V^{n+1} after k iterations. This approach is known as a fixed point iteration. Algorithm 1 shows the fixed point iteration to solve equation (3.23).

Algorithm 1: Fixed point iteration for European options

1. Let $(V^{n+1})^0 = \Phi^{n+1}V^n$
2. Let $\hat{V}^0 = (V^{n+1})^0$
3. For $k = 0, 1, 2, \ldots$ until convergence do
4. Solve $[I - (1-\theta)\Delta\tau L]\hat{V}^{k+1} = \Phi^{n+1}[I + \theta\Delta\tau L + \theta\Delta\tau B]V^n + (1-\theta)\Delta\tau B\hat{V}^k$
5. if $\max_i \frac{|\hat{V}_i^{k+1} - \hat{V}_i^k|}{\max(1, |\hat{V}_i^{k+1}|)} <$ tolerance then
6. Quit
7. end if
8. end for

The convergence rate of the fixed point iteration depends on how well the new coefficient matrix M approximates A. If the jump diffusion term, i.e., B is small, e.g. when $Y \simeq 0$, then M should be a good approximation [12]. However, when Y gets large, the matrix B becomes more important and hence the convergence of the fixed point iteration can be very slow. In this situation, we will need a better method than the simple fixed point iteration.

We propose to use preconditioned BiCGSTAB method [20] as an alternative. The primary concern, then, is choosing an optimal preconditioner M. If we use the same M (3.28) as in the fixed point iteration as the preconditioner, we do not need to form matrix B explicitly. We only need to compute matrix-vector products. The matrix-vector multiplications, BV, are computed efficiently using an FFT and interpolation. When Y gets large, we could use a better preconditioner M which includes the entries of the matrix B, B_{ij}, $|i-j| < c$ where c is a small number. In this case, we need to explicitly form part of the matrix B to extract the diagonal and certain off-diagonal entries. However numerical experiments using this idea did not show any improvement, in terms of CPU time, compared to preconditioner M (3.28). Hence we will use preconditioner M (3.28) in all subsequent numerical experiments.

4 American Options

In this section we extend our numerical scheme to handle American options. The difference between European and American options is that the holder of the latter can exercise at any time before the maturity to receive the payoff. Thus pricing American options under CGMY leads to a linear complementarity problem [25]:

$$V_\tau - \left(\frac{\sigma^2 S^2}{2}V_{SS} + (r-q)SV_S - rV + \int_{-\infty}^{\infty} \nu(y)[V(se^y, \tau) - V(S,\tau) - S(e^y - 1)V_S]dy\right) \geq 0$$
$$(V - V^*) \geq 0, \tag{4.1}$$

where V^* denotes the payoff received upon exercise. Note that at least one of the equations (4.1) holds with strict equality.

We use a penalty method [10, 26, 27] to convert the complementarity problem into a nonlinear algebraic problem by adding a penalty term. Thus, equations (4.1) are combined into a single

equation,

$$V_\tau = \frac{\sigma^2 S^2}{2} V_{SS} + (r-q)SV_S - rV + \int_{-\infty}^{\infty} \nu(y)[V(Se^y,\tau) - V(S,\tau) - S(e^y-1)V_S]dy + p(V,V^*),$$

in which the penalty term $p(V,V^*)$ satisfies $p(V,V^*)=0$ if $V \geq V^*$; $p(V,V^*) \to \infty$ otherwise.

Following the same approach as in the European case, we first discretize the jump integral term:

$$V_\tau = \frac{\sigma^2 + \bar{\sigma}^2}{2} S^2 V_{SS} + (r-q-\kappa)SV_S - (r+\lambda)V + \sum_{j=0}^{N-1} V(Se^{y_j})\hat{\gamma}(y_j) + p(V,V^*). \tag{4.2}$$

Then we apply a semi-Lagrangian scheme for the first order term, coupled with fully implicit timestepping:

$$V_i^{n+1}[1 + (\alpha_i + \beta_i + r + \lambda)\Delta\tau] - \Delta\tau\beta_i V_{i+1}^{n+1} - \Delta\tau\alpha_i V_{i-1}^{n+1} = \Phi^{n+1}V_i^n + \Delta\tau(\mathcal{I}V)_i^{n+1} + p_i^{n+1},$$

where the discrete penalty term $p_i^{n+1} = Large(V_i^* - V_i^{n+1})$ if $V_i^{n+1} < V_i^*$; $p_i^{n+1}=0$ otherwise. The value of *Large* is chosen depending on the accuracy desired (see [26]). Similarly, Crank-Nicolson timestepping for (4.2), after applying the semi-Lagrangian scheme, gives

$$\begin{aligned}
&V_i^{n+1}\left[1 + (\alpha_i + \beta_i + r + \lambda)\frac{\Delta\tau}{2}\right] - \frac{\Delta\tau}{2}\beta_i V_{i+1}^{n+1} - \frac{\Delta\tau}{2}\alpha_i V_{i-1}^{n+1} \\
=&(\Phi^{n+1}V_i^n)\left[1 - (\alpha_i + \beta_i + r + \lambda)\frac{\Delta\tau}{2}\right] + \frac{\Delta\tau}{2}\beta_i(\Phi^{n+1}V_{i+1}^n) + \frac{\Delta\tau}{2}\alpha_i(\Phi^{n+1}V_{i-1}^n) \\
&+ \frac{\Delta\tau}{2}(\mathcal{I}V)_i^{n+1} + \frac{\Delta\tau}{2}(\Phi^{n+1}(\mathcal{I}V)_i^n) + p_i^{n+1}.
\end{aligned}$$

The matrix form of the discrete equations for the penalty method can be written as

$$[I - (1-\theta)\Delta\tau L - (1-\theta)\Delta\tau B + P(V^{n+1})]V^{n+1} = \Phi^{n+1}[I + \theta\Delta\tau L + \theta\Delta\tau B]V^n + [P(V^{n+1})]V^*, \tag{4.3}$$

where $\theta = 0$ corresponds to the fully implicit method and $\theta = \frac{1}{2}$ refers to Crank-Nicolson method; matrices L and B are defined as in (3.24) and (3.25) respectively; P is the matrix form of the penalty term given by

$$\begin{aligned}
P(V^{n+1})_{ii} &= \begin{cases} Large, & \text{if } V_i^{n+1} < V_i^* \\ 0, & \text{otherwise;} \end{cases} \\
P(V^{n+1})_{ij} &= 0, \text{ if } i \neq j.
\end{aligned} \tag{4.4}$$

The coefficient matrix of V^{n+1} in (4.3) is a dense matrix due to the existence of B. As for the European case, we can apply a fixed point iteration method,

$$[I - (1-\theta)L + P(\hat{V}^k)]\hat{V}^{k+1} = \Phi^{n+1}[I + \theta\Delta\tau L + \theta\Delta\tau B]V^n + (1-\theta)\Delta\tau B\hat{V}^k + P(\hat{V}^k)V^*,$$

with initial value of $\hat{V}^0 = V^n$. The newly formed coefficient matrix

$$M = I - (1-\theta)L + P(\hat{V}^k)$$

is a tridiagonal matrix with diagonal entries including the penalty values. The algorithm for a fixed point iteration for American options is given in Algorithm 2.

Algorithm 2: Fixed point iteration for American options

1. Let $\hat{V}^0 = V^n$
2. For $k = 0, 1, 2, \ldots$ until convergence do
3. Solve:
4. $[I - (1-\theta)\Delta\tau L + P(\hat{V}^k)]\hat{V}^{k+1} = \Phi^{n+1}[I + \theta\Delta\tau L + \theta\Delta\tau B]V^n + (1-\theta)\Delta\tau B\hat{V}^k + P(\hat{V}^k)V^*$
5. if $\max_i \frac{|\hat{V}_i^{k+1} - \hat{V}_i^k|}{\max(1, |\hat{V}_i^{k+1}|)} <$ tolerance then
6. Quit
7. end if
8. end for

Similar to the European case, the preconditioned BiCGSTAB scheme can also be applied to (4.3) except that we need to linearize the penalty term at each iterative step. To do this, at the kth iterative step, we fix $\hat{P}^k = P(\hat{V}^k)$ and thus have a linearized system, i.e., $\hat{A}^k\hat{V}^{k+1} = \hat{b}^k$ where

$$\begin{aligned} \hat{A}^k &= I - (1-\theta)\Delta\tau L - (1-\theta)\Delta B + P(\hat{V}^k), \\ \hat{b}^k &= \Phi^{n+1}[I + \theta\Delta\tau L + \theta\Delta B]V^n + P(\hat{V}^k)V^*. \end{aligned} \tag{4.5}$$

We iteratively solve the linearized system by the preconditioned BiCGSTAB with

$$M^k = I - (1-\theta)\Delta\tau L + P(\hat{V}^k)$$

as the preconditioner until the stopping criteria is reached. Then we update the penalty term by $\hat{P}^{k+1} = P(\hat{V}^{k+1})$, and continue with the $(k+1)$th iterative step. Our numerical experiments indicated that it was not optimal to solve the linearized system $\hat{A}^k\hat{V}^{k+1} = \hat{b}^k$ at each step in Algorithm 2 to any great accuracy. In fact, all our numerical results will be reported for the case where we carry out only a single BiCGSTAB iteration for each linearized system solve in Algorithm 2.

5 Stability and Convergence Analysis

In this section, we provide the stability analysis for the fully implicit and Crank-Nicolson timestepping schemes. Our numerical scheme can be generalized to deal with nonlinear pricing problems, hence it is useful to include a proof of the monotonicity of fully implicit timestepping. Finally, we discuss and explore convergence properties of the fixed point and BiCGSTAB iterative methods in detail.

5.1 Stability and Monotonicity of Fully Implicit Discretization

As discussed in [19], stability and monotonicity are useful properties for a numerical scheme in order to ensure convergence to the viscosity solution. A fully implicit discretization of (4.3) with a penalty American constraint is

$$[I - \Delta\tau L - \Delta\tau B + P(V^{n+1})]V^{n+1} = \Phi^{n+1}V^n + [P(V^{n+1})]V^*, \tag{5.1}$$

in which we assume Φ is a linear interpolation operator. First, let's summarize some useful results for the implicit discretization scheme.

Lemma 5.1 *The matrix L in* (5.1) *has the the following properties*

- $\sum_j L_{ij} = -(r+\lambda)$, $i = 1, \ldots, i_{\max} - 1$;
- $L_{ij} = 0$, $i = i_{\max}$;
- $L_{ij} \geq 0$, $i \neq j$, *for* $i, j = 1, \ldots, i_{\max} - 1$.

Proof. This follows directly from the definition of the matrix L (3.24), and the fact that we impose a Dirichlet condition at $i = i_{\max}$ and $\alpha_i, \beta_i \geq 0$ (3.21). □

Lemma 5.2 *Let $A = I - \Delta\tau L - \Delta\tau B$. The matrix A is an M-matrix.*

Proof. It follows from the properties of the matrix B (3.26) and Lemma 5.1, that

$$\sum_j (-L - B)_{ij} \geq 0, \quad \forall i,$$

provided that $r \geq 0$, i.e. A has non-negative row sum. As well, since the off-diagonal entries of B are non-negative (3.26) and $\alpha_i, \beta_i \geq 0$, A has non-positive off-diagonal elements. Note that since we impose the Dirichlet condition at $i = i_{\max}$, A has at least one strictly positive row sum. Hence A is an M matrix. □

Theorem 5.1 *The fully implicit discretization method for American options* (5.1) *is unconditionally l_∞ stable, provided that Φ in equation (5.1) is a linear interpolation operator.*

Proof. By Lemma 5.2, the properties of matrix B (3.26), and using a similar proof as in [23], we can show that

$$\| V^{n+1} \|_\infty \leq \max(\| V^n \|_\infty, \| V^* \|_\infty, |D^{n+1}|),$$

where D^{n+1} is the Dirichlet condition imposed at $i = i_{\max}$. □

Theorem 5.2 *If Φ is a linear interpolation operator in equation (5.1), the fully implicit discretization* (5.1) *is unconditionally monotone.*

Proof. We can rewrite (5.1) at each grid node S_i, $i = 1, \ldots, i_{\max}$ as follows,

$$\mathcal{H}_i(V_i^{n+1}, \{V_j^{n+1}\}_i, V_i^n) \equiv [\Phi^{n+1} V^n]_i - [A V^{n+1}]_i + P(V^{n+1})_{ii}(V^* - V_i^{n+1}) = 0, \qquad (5.2)$$

where $\{V_j^{n+1}\}_i$ refers to the set of values V_j^{n+1} such that $j = 1, \ldots, i_{\max}$ and $j \neq i$. By Lemma 5.2, the definition of penalty matrix P (4.4) and using a similar proof as in [23], it is easy to show that

$$\begin{aligned}
&\mathcal{H}_i(V_i^{n+1}, \{V_j^{n+1}\}_i + \eta_j^{n+1}, V_i^n + \eta_i^n) \geq \mathcal{H}_i(V_i^{n+1}, \{V_j^{n+1}\}_i, V_i^n), \\
&\mathcal{H}_i(V_i^{n+1} + \eta_i^{n+1}, \{V_j^{n+1}\}_i, V_i^n) \leq \mathcal{H}_i(V_i^{n+1}, \{V_j^{n+1}\}_i, V_i^n),
\end{aligned}$$

for $\forall i, j \neq i$, $\eta_j^n \geq 0$, $\eta_j^{n+1} \geq 0$ and $\eta_i^{n+1} \geq 0$. Hence the discretization (5.1) is monotone. □

5.2 Stability of Crank-Nicolson Discretization (European options)

The following result can be shown for European options.

Theorem 5.3 *Suppose that the parameters, σ and r, in (2.3) are constant and upwind quadratic interpolation is used in the semi-Lagrangian discretization. Further, assume an equally spaced grid in $\log S$ coordinates, and periodic boundary conditions. Then Crank-Nicolson timestepping for European options is unconditionally strictly stable in the l_2 norm.*

Proof. We use a similar approach as in [12, 28] by applying Von Neumann analysis (see Appendix C). □

5.3 Convergence of Fixed Point Iteration

Theorem 5.4 *The fixed point iteration is globally convergent for both European and American cases.*

Proof. Consider the European case. Let $E^k = V^{n+1} - \hat{V}^k$ be the error at kth iterative step with $\hat{V}^k$ denoting the approximate value of V^{n+1}. Then E^k satisfies (from Algorithm 1)

$$[I - (1-\theta)\Delta\tau L]E^{k+1} = (1-\theta)\Delta\tau BE^k.$$

Using a similar proof as in [12] together with Lemma 5.2 and properties of matrix B (3.26), we can show that

$$\| E^{k+1} \|_\infty \leq \| E^k \|_\infty \frac{(1-\theta)\lambda\Delta\tau}{1+(1-\theta)(r+\lambda)\Delta\tau}, \tag{5.3}$$

where the scalar,

$$\frac{(1-\theta)\lambda\Delta\tau}{[1+(1-\theta)(r+\lambda)\Delta\tau]} < 1,$$

is independent of the iteration number k. In the case of American options, from Lemma 5.2 and results from [10], we have global convergence but we can not determine the rate of convergence. □

Let's consider the rate of convergence in the European case. Since λ (3.11) is a function of the density function $\nu(y)$ (2.2):

$$\lambda = O\Big(\int_{\Omega_1\cup\Omega_2} \frac{1}{|y|^{1+Y}} dy\Big) = \begin{cases} O((\Delta y)^{-Y}); & Y > 0 \\ O(|\ln \Delta y|); & Y = 0, \end{cases}$$

then we can rewrite equation (5.3) as

$$\frac{\| E^{k+1} \|_\infty}{\| E^k \|_\infty} = \begin{cases} O\left(\frac{(1-\theta)\frac{\Delta\tau}{(\Delta y)^Y}}{1+(1-\theta)\frac{\Delta\tau}{(\Delta y)^Y}}\right); & Y > 0 \\ O\left(\frac{(1-\theta)\Delta\tau|\ln\Delta y|}{1+(1-\theta)\Delta\tau|\ln\Delta y|}\right); & Y = 0. \end{cases}$$

If we take the limit as $\Delta\tau, \Delta y \to 0$, with $\Delta\tau/\Delta y =$ const (which would normally be the case) then, the number of iterations required for convergence to a fixed tolerance would be ($Y > 1$):

$$\#\text{ itns} = O\left(\frac{1}{\left|\ln\left[\frac{1}{1+\frac{(\Delta y)^{Y-1}}{1-\theta}}\right]\right|}\right) \sim O\left((\Delta y)^{1-Y}\right), \quad Y > 1, \quad \Delta y \to 0. \tag{5.4}$$

For the case of $Y < 1$, then the number of iterations per timestep should remain constant or decrease as $\Delta y \to 0$ with $\Delta\tau/\Delta y =$ const.

5.4 Convergence of the BiCGSTAB Method

Consider iterative solution of a matrix A, using a preconditioner M. If A and M are symmetric, then it is known that the number of iterations required for a Conjugate Gradient type method is [29]:

$$\#\text{ itns} = O\Big(\text{cond}(M^{-1}A)^{1/2}\Big),$$

where $\text{cond}(M^{-1}A)$ is the condition number of $M^{-1}A$. In the following, we give a heuristic analysis which provides some insight into the expected behavior of BiCGSTAB for our problem.

The numerical results in [20] show that BiCGSTAB often converges considerably faster than CG-S which is often observed as having a speed of convergence about twice as fast as for BiCG [30]. Since the convergence of BiCG is generally governed by the eigenvalues of the linear system to be solved, we shall estimate the modulus of the largest eigenvalue, $\mu_{\max}$, and the smallest eigenvalue, $\mu_{\min}$, of A (3.27) to gain some insight.

By Gerschgorin's theorem, the eigenvalues of A are upper bounded by the maximum row summation of the absolute value of diagonal entry and the sum of the off-diagonal values, and lower bounded by the minimum difference between them. More precisely,

$$|\mu_{\max}| \le \max_i(|A_{ii}| + \sum_{j=1,j\neq i}^{j_{\max}} |A_{ij}|), \qquad |\mu_{\min}| \ge \min_i(|A_{ii}| - \sum_{j=1,j\neq i}^{j_{\max}} |A_{ij}|)$$

where $j_{\max} = i_{\max}$ is the number of columns of the square matrix A. From (3.27)

$$\begin{aligned} |A_{ii}| &= |1 + (1-\theta)(\alpha_i + \beta_i + r + \lambda)\Delta\tau - (1-\theta)\Delta\tau B_{ii}| \\ &\le 1 + (1-\theta)(\alpha_i + \beta_i + r + \lambda)\Delta\tau + (1-\theta)\Delta\tau B_{ii} \end{aligned} \tag{5.5}$$

since $\alpha_i, \beta_i, B_{ii} \ge 0$ and $0 \le \theta \le 1$, and the sum of the absolute values of the off-diagonal entries is

$$\sum_{j=1,j\neq i}^{j_{\max}} |A_{ij}| = (1-\theta)(\alpha_i + \beta_i)\Delta\tau + (1-\theta)\Delta\tau \sum_{j=1,j\neq i}^{j_{\max}} B_{ij}. \tag{5.6}$$

Note that

$$\begin{aligned}\max_i(\alpha_i+\beta_i) &= \max_i\left((\sigma^2+\bar{\sigma})S_i^2\left[\frac{1}{(S_i-S_{i-1})(S_{i+1}-S_{i-1})}+\frac{1}{(S_{i+1}-S_i)(S_{i+1}-S_{i-1})}\right]\right)\\ &= \max_i\left(\frac{(\sigma^2+\bar{\sigma})S_i^2}{(S_i-S_{i-1})(S_{i+1}-S_i)}\right)\\ &= \frac{(\sigma^2+\bar{\sigma})S_{i_{\max}}^2}{(\Delta S)^2}, \end{aligned} \tag{5.7}$$

if we assume that $\Delta S = S_{i+1} - S_i =$ const. Now combining (5.5), (5.6), (5.7) and (3.11), we have

$$\begin{aligned}|\mu_{\max}| &\le 1 + 2(1-\theta)\Delta\tau\frac{(\sigma^2+\bar{\sigma})S_{i_{\max}}^2}{(\Delta S)^2} + (1-\theta)\Delta\tau(r+\lambda) + (1-\theta)\Delta\tau\max_i\left(\sum_{j=1}^{j_{\max}}B_{ij}\right)\\ &\le 1 + 2(1-\theta)\Delta\tau\frac{(\sigma^2+\bar{\sigma})S_{i_{\max}}^2}{(\Delta S)^2} + (1-\theta)\Delta\tau(r+\lambda) + (1-\theta)\Delta\tau\lambda\\ &= O\left(\Delta\tau\frac{\bar{\sigma}}{(\Delta S)^2}+\Delta\tau\lambda\right), \end{aligned} \tag{5.8}$$

since $\theta, \sigma, S_{i_{\max}}, r$ are constants. In (5.8), the first term $\Delta\tau\frac{\bar{\sigma}}{(\Delta S)^2}$ corresponds to diffusion and the second term $\Delta\tau\lambda$ corresponds to the jump integral. Note that at each time step we apply the preconditioner M (3.28) which we assume minimizes the effect of the diffusion term. Assuming the second term dominates in (5.8) for preconditioned BiCGSTAB, we conclude that

$$|\mu_{\max}| = O(\Delta\tau(\Delta y)^{-Y}), \qquad Y>0, \qquad \Delta y \to 0.$$

Similarly, we have

$$\begin{aligned}|\mu_{\min}| &\ge \min_i\left(1+(1-\theta)(\alpha_i+\beta_i+r+\lambda)\Delta\tau-(1-\theta)(\alpha_i+\beta_i)\Delta\tau-(1-\theta)\Delta\tau\sum_{j=1}^{j_{\max}}B_{ij}\right)\\ &\ge \min_i(1+(1-\theta)(\alpha_i+\beta_i+r+\lambda)\Delta\tau-(1-\theta)(\alpha_i+\beta_i)\Delta\tau-(1-\theta)\lambda\Delta\tau)\\ &= O(1+(1-\theta)r\Delta\tau)\\ &= O(1). \end{aligned} \tag{5.9}$$

Thus, we conjecture that the number of iterations required for convergence for preconditioned BiCGSTAB is

$$\#\text{ itns } = O\left(\sqrt{\frac{|\mu_{\max}|}{|\mu_{\min}|}}\right) = O(\sqrt{\Delta\tau(\Delta y)^{-Y}}), \qquad Y>0. \tag{5.10}$$

If we take the limit as $\Delta\tau, \Delta y \to 0$, with $\Delta\tau/\Delta y =$ const, then, for $Y > 1$, we have

$$\#\text{ itns } = O\left(\sqrt{(\Delta y)^{1-Y}}\right), \qquad Y>1, \qquad \Delta y \to 0,$$

which is a considerable improvement compared to equation (5.4).

6 Numerical Results

This section presents the numerical results for pricing European and American options under the CGMY model. Unless stated otherwise, we apply the semi-Lagrangian discretization method to (2.3) with Crank-Nicolson timestepping. The discrete system of the equations is solved using preconditioned BiCGSTAB. We also include numerical results using a fixed point iteration for comparison.

6.1 European Options

In this section, we carry out a series of convergence tests for evaluating European options. In these tests, we are interested in studying how fast the computational results converge to the exact solution. Let $\hat{V}(h)$ denote the approximate option value, for a given $\Delta\tau$ and grid spacing, and let $\hat{V}(h/2)$ denote the solution computed with the timestep reduced by a factor of two, and new nodes inserted halfway between each node in the original grid. The convergence ratio, consequently, is defined as

$$\text{ratio} = \frac{\hat{V}(\frac{h}{2}) - \hat{V}(h)}{\hat{V}(\frac{h}{4}) - \hat{V}(\frac{h}{2})}. \tag{6.1}$$

Suppose that the computational error is $\hat{V} - V = O(h^{\zeta})$. If ratio $= 2$ then from (6.1) we conclude $\zeta = 1$, which implies a linear convergence. If ratio $= 4$ then $\zeta = 2$, which indicates a quadratic convergence.

For both BiCGSTAB and the fixed point iteration, the matrix-vector multiply is the most costly operation in terms of flops. Since BiCGSTAB requires two matrix vector multiplies each iteration [20], this means that each BiCGSTAB iteration is roughly twice as expensive (in terms of flops) as a fixed point iteration. We remind the reader to take this into account when examining our numerical results. We include normalized CPU time in the table, but we caution the reader that the timing results are unreliable. This is because we oversample the S grid typically by about four times. This means that the number of nodes used as input to the FFT algorithm on the finest S grid is about 16,384. As discussed in [31], this is the size of an array which causes a dramatic decrease in performance of an FFT due to cache problems.

When $Y = 0$, the CGMY process is also known as Variance Gamma. Table 2 presents the numerical results for pricing a European call option as we carry out a series of tests, where we double the number of S grid nodes and the total number of timesteps for each test, with input parameters given in Table 1.

S	K	T	r	q	σ	C	G	M	Y
90	98	0.5	0.0	0.0	0.0	5.9311	20.2648	39.784	0.0

TABLE 1: *The input parameters (from [32]) for evaluating a European call option under a Variance Gamma process. Here K denotes the strike price and T the maturity time.*

.

The first and second columns in Table 2 show that we are refining ΔS in the space domain and $\Delta\tau$ in the time domain by half for each test. The constant number of iterations per time step for both iterative methods indicate that under a Variance Gamma process the convergence rate of the iterative methods is independent of the size of Δy, as predicted in our theoretical analysis of these

size of	time	Fixed Point				BiCGSTAB			
S grid	steps	itns	NCPU	value	ratio	itns	NCPU	value	ratio
129	50	6.04	1.00	0.60342750	n.a.	3.00	1.17	0.60342750	n.a.
257	100	5.02	2.17	0.61091895	n.a.	2.04	2.04	0.61091895	n.a.
513	200	5.01	4.06	0.61286313	3.8533	2.01	4.29	0.61286313	3.8533
1025	400	4.19	8.22	0.61326232	4.8703	2.00	8.79	0.61326232	4.8703
2049	800	4.00	16.64	0.61335445	4.3329	2.00	19.47	0.61335445	4.3329
4097	1600	4.00	35.57	0.61337338	4.8669	2.00	37.11	0.61337338	4.8669

TABLE 2: *Comparison of fixed point iteration and preconditioned BiCGSTAB method for evaluating a European call option under the Variance Gamma process. The tolerance for the stopping criteria in both iterative methods is* 10^{-8}. *"Itns" is the average number of iterations per time step. "NCPU" is the normalized average CPU time per time step. Note that each BiCGSTAB iteration requires about twice as many floating point operations as a fixed point iteration. Since we oversample the FFT grid by four times compared to the size of S grid, the CPU time is unreliable due to cache problems (see [31]).*

two iterative methods in sections 5.3 and 5.4. The column of ratios (see equation (6.1)) in Table 2 suggests that quadratic convergence is achieved as the mesh and timestep size are reduced.

When $0 < Y < 1$, the Lévy process has infinite activity but finite variation. We carry out two experiments when $Y = 0.6442$ and $Y = 1.0102$ with the corresponding input parameters listed in Table 3.

S	K	T	r	q	σ	C	G	M	Y
90	98	0.25	0.06	0.0	0.0	16.97	7.08	29.97	0.6442
90	98	0.25	0.06	0.0	0.0	0.42	4.37	191.2	1.0102

TABLE 3: *The input parameters (from [2]) for evaluating European call options under CGMY when* $Y = 0.6442$ *and* $Y = 1.0102$ *respectively. Here* K *denotes the strike price and* T *the maturity time.*

.

size of	time	Fixed Point				BiCGSTAB			
S grid	steps	itns	NCPU	value	ratio	itns	NCPU	value	ratio
129	25	64.68	1.00	16.640026	n.a.	25.68	1.10	16.640015	n.a.
257	50	57.82	1.77	16.327662	n.a.	19.44	1.12	16.327636	n.a.
513	100	50.89	3.21	16.242114	3.6513	8.35	1.31	16.242073	3.6509
1025	200	43.66	5.85	16.219638	3.8062	7.99	2.57	16.219588	3.8053
2049	400	36.32	10.38	16.213900	3.9170	7.41	5.25	16.213831	3.9057
4097	800	29.76	18.13	16.212478	4.0352	6.37	9.47	16.212375	3.9540

TABLE 4: *Comparison of fixed point iteration and preconditioned BiCGSTAB method in evaluating of a European call option under CGMY when* $Y = 0.6442$. *The tolerance for the stopping criteria in both iterative methods is* 10^{-8}. *"Itns" is the average number of iterations per time step. "NCPU" is the normalized average CPU time per time step. Note that each BiCGSTAB iteration requires about twice as many floating point operations as a fixed point iteration. Since we oversample the FFT grid by four times compared to the size of S grid, the CPU time is unreliable due to cache problems (see [31]).*

size of S grid	time steps	Fixed Point				BiCGSTAB			
		itns	NCPU	value	ratio	itns	NCPU	value	ratio
129	25	33.60	1.00	2.2691911	n.a.	19.68	1.70	2.2691906	n.a.
257	50	35.18	2.18	2.2411747	n.a.	9.96	1.85	2.241173	n.a.
513	100	37.32	5.07	2.2334153	3.6106	5.28	2.14	2.2334126	3.6103
1025	200	37.81	11.46	2.2313570	3.7698	4.89	4.21	2.2313511	3.7644
2049	400	37.62	25.63	2.2308335	3.9318	4.07	8.63	2.2308225	3.8999
4097	800	36.67	64.17	2.2307031	4.0146	3.09	14.59	2.2306913	4.0290

TABLE 5: *Comparison of fixed point iteration and preconditioned BiCGSTAB method in evaluating of a European call option under CGMY when* $Y = 1.0102$. *The tolerance for the stopping criteria in both iterative methods is* 10^{-8}. *"Itns" is the average number of iterations per time step. "NCPU" is the normalized average CPU time per time step. For comparison, a value of* 2.2306557 *was obtained in [33]. Note that each BiCGSTAB iteration requires about twice as many floating point operations as a fixed point iteration. Since we oversample the FFT grid by four times compared to the size of S grid, the CPU time is unreliable due to cache problems (see [31]).*

When Y is close to or smaller than one, the error of our discretization is bounded by $O((\Delta y)^2)$ which implies that quadratic convergence is expected. Our numerical results in Table 4 and 5 confirm this conclusion. If we compare the two iterative methods with the same Y, it takes no more than ten iterations per time step to converge for the preconditioned BiCGSTAB method whereas the fixed point iteration needs about five to ten times more iterations. As Y gets large, it is more advantageous to use preconditioned BiCGSTAB method over the fixed point iteration, bearing in mind that each BiCGSTAB iteration is twice as costly as a fixed point iteration.

When $1 < Y < 2$, this corresponds to a Lévy process with infinite activity and infinite variation. Table 7 and 8 show the numerical results for pricing a European put option with $Y = 1.4$ and $Y = 1.8$ respectively, with the input parameters listed in Table 6.

S	K	T	r	q	σ	C	G	M	Y
500	500	0.25	0.4	0.0	0.2	1.0	1.4	2.5	1.4
10	10	0.25	0.1	0.0	0.0	1.0	8.8	9.2	1.8

TABLE 6: *The input parameters for evaluating European put options under CGMY when* $Y > 1$. *When* $Y = 1.4$, *the parameters are taken from [16], whereas the parameters for* $Y = 1.8$ *are selected to stress our numerical algorithm.*

.

As mentioned in section 3, the error of our numerical scheme for pricing option values under an infinite variation process is bounded by $O((\Delta y)^{3-Y})$. Tables 7 and 8 show the expected results, i.e. our numerical scheme can achieve better than a first order convergence rate when $1 < Y < 2$, but not a second order rate.

6.2 American Options

In this section, a number of numerical experiments are carried out to illustrate the performance and convergence of pricing American options with our numerical scheme. First, we use a semi-Lagrangian fully implicit discretization to compute the American put options when $Y = 1.4$ and $Y = 1.6$ using the CGMY parameters listed in Table 9.

size of	time	Fixed Point				BiCGSTAB			
S grid	steps	itns	NCPU	value	ratio	itns	NCPU	value	ratio
139	25	51.64	2.28	108.41531	n.a.	8.72	1.00	108.41532	n.a.
277	50	60.86	5.27	108.47423	n.a.	8.86	2.18	108.47425	n.a.
553	100	72.14	12.69	108.49175	3.3630	9.13	4.83	108.49180	3.3578
1105	200	85.94	34.58	108.49707	3.2932	9.39	11.67	108.49724	3.2261
2209	400	102.56	80.99	108.49865	3.3671	9.60	24.22	108.49888	3.3171
4417	800	122.28	235.09	108.49914	3.2245	10.29	59.93	108.49939	3.2157

TABLE 7: *Comparison of fixed point iteration and BiCGSTAB method in evaluating of a European put option under CGMY when* $Y = 1.4$. *The tolerance for the stopping criteria in both iterative methods is* 10^{-8}. *"Itns" is the average number of iterations per time step. "NCPU" is the normalized average CPU time per time step. Note that each BiCGSTAB iteration requires about twice as many floating point operations as a fixed point iteration. Since we oversample the FFT grid by four times compared to the size of S grid, the CPU time is unreliable due to cache problems (see [31]).*

size of	time	Fixed Point				BiCGSTAB			
S grid	steps	itns	NCPU	value	ratio	itns	NCPU	value	ratio
129	25	48.44	3.33	4.1690294	n.a.	8.68	1.00	4.1690297	n.a.
257	50	88.46	9.29	4.2859081	n.a.	12.14	2.67	4.2859097	n.a.
513	100	144.12	26.59	4.3348687	2.3872	16.94	7.34	4.3348743	2.3870
1025	200	216.52	83.59	4.3563344	2.2809	24.43	21.61	4.3563531	2.2797
2049	400	309.92	248.91	4.3667237	2.0661	30.69	69.22	4.3667826	2.0594
4097	800	415.49	685.81	4.3714972	2.1765	49.32	212.04	4.3716708	2.1336

TABLE 8: *Comparison of fixed point iteration and BiCGSTAB method in evaluating of a European put option under CGMY when* $Y = 1.8$. *The tolerance for the stopping criteria in both iterative methods is* 10^{-8}. *"Itns" is the average number of iterations per time step. "NCPU" is the normalized average CPU time per time step. Note that each BiCGSTAB iteration requires about twice as many floating point operations as a fixed point iteration. Since we oversample the FFT grid by four times compared to the size of S grid, the CPU time is unreliable due to cache problems (see [31]).*

S	K	T	r	q	σ	C	G	M	Y
500	500	0.5	0.4	0.0	0.2	1.0	1.4	2.5	1.4
10	10	0.5	0.1	0.0	0.0	1.0	8.8	9.2	1.6

TABLE 9: *The input parameters (from [16]) for evaluating American put options under CGMY when* $Y > 1$. *Here* K *denotes the strike price and* T *the maturity time.*

.

In Figure 1(a) the American put option is compared with its counterpart European type, whereas in Figure 1(b) the value of an American put option is presented with varying time to maturity.

Second, we apply a semi-Lagrangian Crank-Nicolson discretization to carry out a convergence test for computing an American put option when $Y = 1.0102$. The numerical results are given in Table 10. The input parameters are the same as in Table 3, except that the option type now is an American put. The results show that we can achieve quadratic convergence for pricing American options when Y is close to 1.

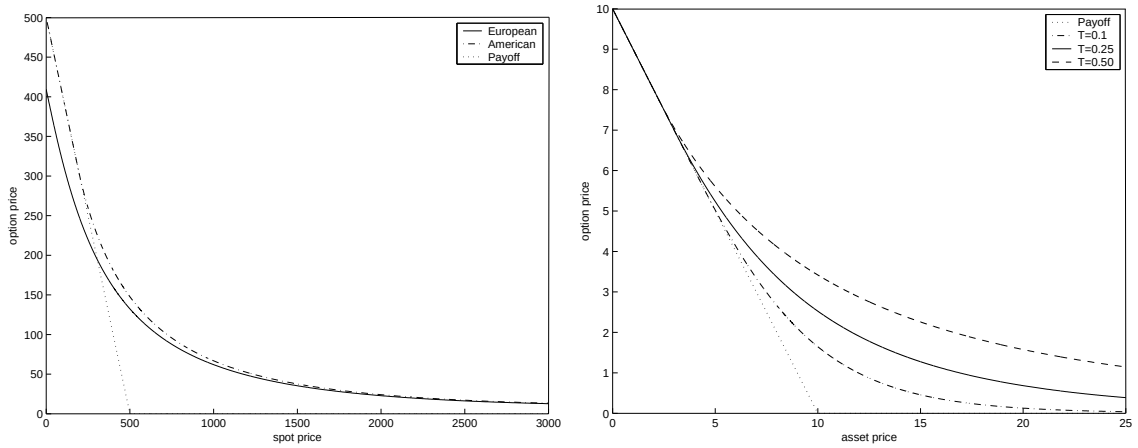

FIGURE 1: *(a) Left: Compare American and European put options under CGMY when* $Y = 1.4$*. The size of the* S *grid is 4417 and the total number of time steps is 800. (b) Right: American put options of different maturities under CGMY with zero volatility and* $Y = 1.6$*.*

Finally, we use a semi-Lagrangian and fully implicit discretization scheme to compute the American call options under CGMY with zero and non-zero diffusion components. Figure 2 shows the option values and their deltas with $Y = 0$ and $Y = 0.6$. The input parameters are listed in Table 11.

We note that in Figure 2(b) the delta shows a discontinuous jump without a diffusion component under a Variance Gamma process. As Y gets large, the discontinuity is smoothed out (see Figure 2(d)). The results are consistent with our intuition: as the Lévy process becomes more singular near zero, it behaves more like a diffusion process.

7 Conclusion

The difficulty of solving the option pricing PIDE under an infinite activity jump process can be characterized by the nature of the singularity of the jump size density function. Let

$$\nu(y) \quad = \quad O\left(\frac{1}{|y|^{1+Y}}\right) \quad ; \quad y \to 0 \quad . \tag{7.1}$$

The basic method developed in this paper splits the jump integral term into two components: a region near $y = 0$, and a region away from $y = 0$. An implicit timestepping technique is employed, and an iterative method is used to avoid a dense matrix solve. The iterative method lags the portion of the integral away from $y = 0$.

For the case where $0 \leq Y \leq 1$, a simple fixed point iteration is satisfactory, and the discretization method exhibits second order convergence as the mesh and time step are refined. In the case of $1 \leq Y < 2$, the fixed point iteration becomes inefficient. However, our numerical tests indicate that a BiCGSTAB iteration is very effective at keeping the number of iterations per time step small. As $Y \to 2$, the convergence of the method as the mesh and time step are refined degenerates gracefully to first order. This is a consequence of the method used to approximate the jump integral near $y = 0$. In addition, the number of iterations required to solve the discretized algebraic equation at each time step increases as $Y \to 2$. In summary, the method used here is very effective for

size of	time	Fixed Point				BiCGSTAB			
S grid	steps	itns	NCPU	value	ratio	itns	NCPU	value	ratio
129	25	38.28	1.18	9.2639099	n.a.	11.08	1.00	9.2639099	n.a.
257	50	38.92	2.92	9.2363535	n.a.	10.64	2.29	9.2363536	n.a.
513	100	39.00	8.24	9.2283662	3.4500	10.47	6.32	9.2283665	3.4501
1025	200	38.57	19.60	9.2261955	3.6796	10.34	14.52	9.2261963	3.6804
2049	400	37.65	45.74	9.2256268	3.8170	10.00	35.02	9.2256285	3.8221
4097	800	36.39	129.97	9.2254803	3.8819	9.48	100.39	9.2254842	3.9349

TABLE 10: *Comparison of fixed point iteration and preconditioned BiCGSTAB method in evaluating of an American put option under CGMY when* $Y = 1.0102$. *The tolerance for the stopping criteria in both iterative methods is* 10^{-8}. *For comparison, a value of* 9.225439 *was obtained in [33]. For the fixed point method, "Itns" is the average number of iterations per time step. For the BiCGSTAB method, "Itns" is the average total iterations per timestep. "NCPU" is the normalized average CPU time per time step. For each linearized solution in Algorithm 2, we carry out only a single BiCGSTAB iteration. Note that each BiCGSTAB iteration requires about twice as many floating point operations as a fixed point iteration. Since we oversample the FFT grid by four times compared to the size of S grid, the CPU time is unreliable due to cache problems (see [31]).*

S	K	T	r	q	σ	C	G	M	Y
100	100	9.0	0.1	0.1	0.0	0.0	9.5085	5.2585	0.0
100	100	9.0	0.1	0.1	0.0	0.0	9.5085	5.2585	0.6

TABLE 11: *The input parameters (from [9]) for evaluating American call options under CGMY process when* $Y = 0.0$ *and* $Y = 0.6$ *respectively. Here* K *denotes the strike price and* T *the maturity time.*

.

$0 \le Y \le 1$, and obtaining reasonable performance for $1 \le Y \le 1.5$. For $Y > 1.5$, convergence rate slows and computational costs increase.

The technique developed here is simple to implement in existing option pricing software. Since this method is modular, and builds on standard option pricing building blocks, we can easily generalize this method to handle American early exercise and other path dependent features. In addition, in the fully implicit case, where linear interpolation is used in the semi-Lagrangian step, the discretization method is unconditionally monotone and l_∞ stable. This makes this method potentially useful for nonlinear pricing problems, e.g. optimal stochastic control, where questions of convergence to the viscosity solution arise.

A Error Analysis

In this section we discuss the total error from computing the jump integral part in Ω_1 (3.3). To be more specific, the error term E_j in (3.7) is composed of the following two components:

$$E_j^1 = \int_{y_j-\Delta y/2}^{y_j+\Delta y/2} \underline{F}'(y_j)(y-y_j)\underline{\nu}(y)dy, \qquad E_j^2 = \int_{y_j-\Delta y/2}^{y_j+\Delta y/2} \underline{F}''(\xi(y))(y-y_j)^2\underline{\nu}(y)dy,$$

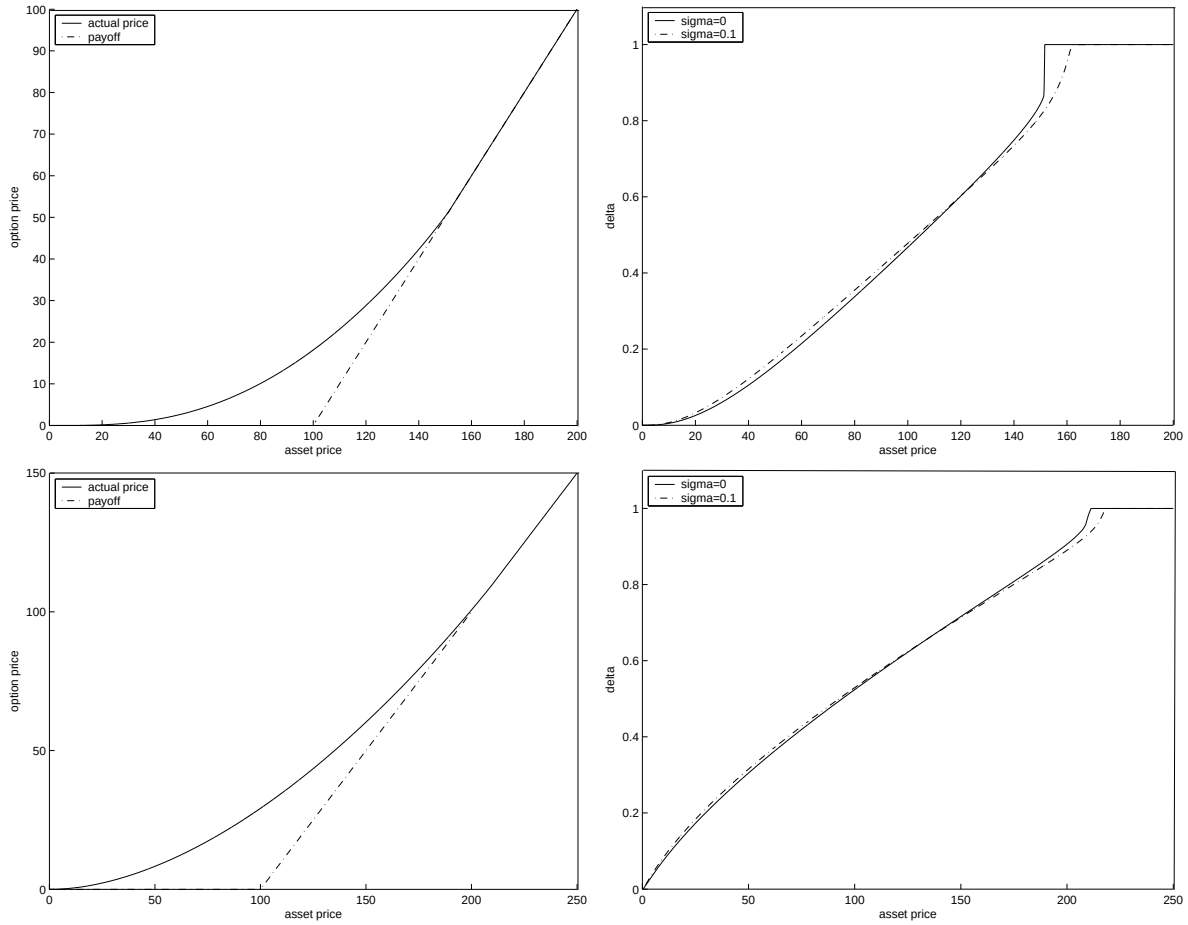

FIGURE 2: *(a) upper left: American call option with zero diffusion and payoff function when $Y = 0$; (b) upper right: deltas of the American calls with and without diffusion terms when $Y = 0$; (c) lower left: American call option and payoff with zero diffusion when $Y = 0.6$ (d) lower right: deltas of the American call with and without diffusion terms when $Y = 0.6$. The size of S grid is 1025 and the total number of time steps is 7200.*

with $\xi(y) \in [y_j - \Delta y/2, y_j + \Delta y/2]$. Let $\underline{F}''(\xi(y)) \leq \mathcal{C}$ where $\mathcal{C}$ denotes some finite constant number, then from $\underline{\nu}(y) = y^2\nu(y) = O(|y|^{1-Y})$ we have

$$\begin{aligned}
\sum_{j:y_j \in \Omega_1} E_j^2 &\leq \mathcal{C}_1(\Delta y)^2 \int_{\Omega_1} |y|^{1-Y} dy \\
&= \mathcal{C}_1(\Delta y)^2 \left(1_{y>0} \frac{y^{2-Y}}{2-Y} \Big|_{\Delta y/2}^{1} + 1_{y<0} \frac{|y|^{2-Y}}{2-Y} \Big|_{-1}^{-\Delta y/2} \right) \\
&\leq \mathcal{C}_1(\Delta y)^2 |\log(\Delta y)| \qquad \text{when} \qquad Y \to 2,
\end{aligned}$$

where $\mathcal{C}_1$ is some constant.

The error term E_j^1 can be rewritten as

$$E_j^1 = \int_{y_j-\Delta y/2}^{y_j+\Delta y/2} \underline{F}'(y_j)\underline{\nu}'(\eta(y))(y-y_j)^2 dy,$$

for some $\eta(y) \in [y_j - \Delta y/2, y_j + \Delta y/2]$ by doing the Taylor expansion of $\underline{\nu}(y)$ around y_j as well. Let $\underline{F}'(y_j)$ be bounded independent of y_j, then from $\underline{\nu}'(y) = O(|y|^{-Y})$ we have

$$\begin{aligned}
\sum_{j:y_j\in\Omega_1} E_j^1 &\le \mathcal{C}_2(\Delta y)^2 \int_{\Omega_1} |y|^{-Y} dy \\
&= \mathcal{C}_2(\Delta y)^2 \left(1_{y>0} \frac{y^{1-Y}}{|1-Y|}\Big|_{\Delta y/2}^{1} + 1_{y<0} \frac{|y|^{1-Y}}{|1-Y|}\Big|_{-1}^{-\Delta y/2} \right) \\
&\le \mathcal{C}_2 \max((\Delta y)^{2-\epsilon}, (\Delta y)^{3-Y}),
\end{aligned}$$

where $\mathcal{C}_2$ is some constant, and ϵ is an arbitrarily small positive number. Thus the total error from computing the integral in Ω_1 is

$$E_{\Omega_1} = \sum E_j^1 + \sum E_j^2 = O((\Delta y)^{\min(2-\epsilon, 3-Y)}).$$

B Quadrature Rules

In this section we provide the computational details for approximating the finite integral of the Lévy measure, i.e. $\gamma(y_j)$ (3.8), $\forall y_j \in \Omega_1 \cup \Omega_2$. As we have discussed in Section 3.1, the Lévy measure $\nu(y)$ over Ω_2 is a smooth function, hence we can use a standard numerical integration method, e.g. the composite Trapezoidal rule, to compute

$$\gamma(y_j) = \frac{\Delta y}{n}\big[\frac{1}{2}\nu(z_0) + \nu(z_1) + \ldots + \nu(z_{n-1}) + \frac{1}{2}\nu(z_n)\big] + \frac{1}{12}\frac{(\Delta y)^3}{n}\gamma''(\xi(y)), \qquad y_j \in \Omega_2$$

where $z_0 = y_j - \Delta y/2$, $z_n = y_j + \Delta y/2$, and $\xi(y) \in [y_j - \Delta y/2, y_j + \Delta y/2]$. If we choose $n = 4$ subdivisions, we have

$$\hat{\gamma}(y_j) = \frac{\Delta y}{4}\big[\frac{1}{2}\nu(z_0) + \nu(z_1) + \nu(z_2) + \nu(z_3) + \frac{1}{2}\nu(z_4)\big], \quad y_j \in \Omega_2$$

with a local error of $O((\Delta y)^3)$.

The approximation of $\gamma(y_j)$ for $y_j \in \Omega_1$ is more subtle due to the singularity of the Lévy measure $\nu(y)$ over Ω_1. In the Trapezoidal rule, the weights on the sample locations are independent of the integrand. A special quadrature rule, which systematically computes the weights on the evaluated points depending on the integrated function, is more desirable for the integral of $\gamma(y_j)$ when $y_j \in \Omega_1$.

The basic idea of the special quadratic rule is to choose weights $w_1, w_2, \ldots, w_m$ on the fixed uniform locations $z_1, z_2, \ldots, z_m$ in the interval $[a, b]$ to minimize the error in the approximation:

$$\int_b^a \frac{1}{y^\eta} f(y) dy \simeq \sum_{i=1}^m w_i f(z_i) + \text{Error}, \tag{B.1}$$

where in our particular case $\eta = -1 + Y$, $a = y_j - \Delta y/2$, $b = y_j + \Delta y/2$ and

$$f(y) = Ce^{-My}1_{y>0} + Ce^{-G|y|}1_{y<0}.$$

If we apply Taylor expansion on e^{-My} when $y > 0$, we have

$$f(y) = C(1 - My + \frac{M^2}{2}y^2 - \frac{M^3}{6}y^3 + O(y^4)).$$

Similarly, $f(y)$ can also be approximated by Taylor expansion when $y < 0$. Then substituting the expansion result into (B.1), when $y > 0$ we have, up to the 3rd order for $f(y)$,

$$\begin{aligned}
&C\int_a^b \frac{1}{y^\eta}dy - CM\int_a^b \frac{1}{y^\eta}ydy + \frac{CM^2}{2}\int_a^b \frac{1}{y^\eta}y^2dy - \frac{CM^3}{6}\int_a^b \frac{1}{y^\eta}y^3dy \\
\simeq &C\sum_{i=1}^m w_i - CM\sum_{i=1}^m w_i z_i + \frac{CM^2}{2}\sum_{i=1}^m w_i z_i^2 - \frac{CM^3}{6}\sum_{i=1}^m w_i z_i^3,
\end{aligned}$$

with the error term in equation (B.1):

$$\text{Error } = O\big(\int_a^b \frac{y^4}{y^\eta}dy\big). \tag{B.2}$$

In order to minimize the error terms, we must have

$$CM^j\int_a^b y^{j-\eta}dy - CM^j\sum_{i=1}^m w_i z_i^j = 0, \qquad 0 \le j \le 3.$$

Note that

$$\int_a^b y^{j-\eta}dy = \frac{1}{j+1-\eta}y^{j+1-\eta}\Big|_a^b = \frac{b^{j+1-\eta} - a^{j+1-\eta}}{j+1-\eta}.$$

Similar equations can be derived when $y < 0$. If we preallocate $m = 4$ points, z_1, z_2, z_3 and z_4, we only have four unknowns w_1, w_2, w_3 and w_4 with the following four constraint conditions

$$\begin{aligned}
j = 0 \qquad & w_1 + w_2 + w_3 + w_4 = \frac{b^{1-\eta} - a^{1-\eta}}{1-\eta}, \\
j = 1 \qquad & w_1 z_1 + w_2 z_2 + w_3 z_3 + w_4 z_4 = \frac{b^{2-\eta} - a^{2-\eta}}{2-\eta}, \\
j = 2 \qquad & w_1 z_1^2 + w_2 z_2^2 + w_3 z_3^2 + w_4 z_4^2 = \frac{b^{3-\eta} - a^{3-\eta}}{3-\eta}, \\
j = 3 \qquad & w_1 z_1^3 + w_2 z_2^3 + w_3 z_3^3 + w_4 z_4^3 = \frac{b^{4-\eta} - a^{4-\eta}}{4-\eta}.
\end{aligned}$$

By solving the above linear system to obtain the optimal weights, we have

$$\hat{\gamma}(y_j) = \frac{1}{y_j^2}\sum_{i=1}^4 w_i f(z_i), \quad z_i = y_j - \frac{\Delta y}{2} + i\frac{\Delta y}{5}, \quad y_j \in \Omega_1.$$

In the worst case, as $Y \to 2$, $\eta = -1 + Y = 1$, which makes the error term in (B.2) become

$$\int_a^b \frac{y^4}{y} dy = \int_a^b y^3 dy = \frac{(b-a)^4}{4} = O((\Delta y)^4),$$

since $b - a = \Delta y$, hence

$$\hat{\gamma}(y_j) = \gamma(y_j) + O\left(\frac{(\Delta y)^4}{y_j^2}\right), \qquad y_j \in \Omega_1 \quad . \tag{B.3}$$

C Von Neumann Stability Analysis of the Crank-Nicolson method

We assume constant coefficients in the transformed PIDE (3.12), with an evenly spaced grid along the $\log S$ coordinate, and periodic boundary conditions. The Crank-Nicolson method is applied after the first order term has been discretized by a semi-Lagrangian scheme with upwind quadratic interpolation.

First, by applying the change of variables $x = \log S$, and letting $\overline{V}(x, \tau) = V(e^x, \tau)$, we rewrite equation (3.12) as the following PIDE, as a function of $\overline{V}(x, \tau)$,

$$\overline{V}_\tau = \frac{\sigma^2 + \bar{\sigma}}{2} \overline{V}_{xx} + (r - q - \kappa - \frac{\sigma^2 + \bar{\sigma}}{2}) \overline{V}_x - (r + \lambda) \overline{V} + \overline{V} \otimes \hat{\gamma}, \tag{C.1}$$

where $\overline{V} \otimes \hat{\gamma}$ is a correlation product.

Then, we use the semi-Lagrangian discretization (see Section 3.2 for details) on equation (C.1). Let $x(x_i, \tau^{n+1}, \tau)$ be the trajectory satisfying

$$\frac{dx}{d\tau} = -(r - q - \kappa - \frac{\sigma^2 + \bar{\sigma}}{2}), \tag{C.2}$$

which passes through the grid point x_i at time τ^{n+1}. Let $x_{i(n+1)}$ be the departure point of this trajectory at time τ^n. Then $x_{i(n+1)}$ is determined by solving the ODE (C.2) starting at x_i from τ^{n+1} to τ^n:

$$x_{i(n+1)} = x_i - (r - q - \kappa - \frac{\sigma^2 + \bar{\sigma}}{2}) \Delta\tau. \tag{C.3}$$

The value of function $\overline{V}^n_{i(n+1)} \equiv \overline{V}(x_{i(n+1)}, \tau^n)$ is computed by an upwind quadratic interpolation from the original grid $\overline{V}^n$ and x.

The Crank-Nicolson discretization of equation (C.1), after applying semi-Lagrangian timestepping, can be written as

$$\begin{aligned} &\overline{V}_i^{n+1} \left[1 + \alpha\Delta\tau + (r + \lambda)\frac{\Delta\tau}{2}\right] - \frac{\Delta\tau}{2}\alpha\overline{V}_{i+1}^{n+1} - \frac{\Delta\tau}{2}\alpha\overline{V}_{i-1}^{n+1} - \frac{\Delta\tau}{2}(\overline{V} \otimes \hat{\gamma})_i^{n+1} \\ =&\overline{V}^n_{i(n+1)} \left[1 - \alpha\Delta\tau - (r + \lambda)\frac{\Delta\tau}{2}\right] + \frac{\Delta\tau}{2}\alpha\overline{V}^n_{i+1(n+1)} + \frac{\Delta\tau}{2}\alpha\overline{V}^n_{i-1(n+1)} + \frac{\Delta\tau}{2}(\overline{V} \otimes \hat{\gamma})^n_{i(n+1)}, \end{aligned} \tag{C.4}$$

where

$$\alpha = \frac{\sigma^2 + \bar{\sigma}}{2(\Delta x)^2}.$$

Suppose the departure point $x_{i(n+1)}$ lies between the grid points x_{i-p} and x_{i-p-1} for some integer p, i.e. $x_{i-p-1} \le x_{i(n+1)} \le x_{i-p}$. Because the difference between $x_{i(n+1)}$ and x_i, $\forall i$, in (C.3) is constant, we can consequently locate $x_{i-1(n+1)}$ and $x_{i+1(n+1)}$ such that

$$x_{i-p-2} \le x_{i-1(n+1)} \le x_{i-p-1}, \qquad x_{i-p} \le x_{i+1(n+1)} \le x_{i-p+1}.$$

By upwind quadratic interpolation, the location of the third point depends on the coefficient of the drift term,

$$d \equiv -(r - q - \kappa - \frac{\sigma^2 + \bar{\sigma}}{2})$$

in (C.2). If d is positive, we choose x_{i-p-3}, x_{i-p-2} and x_{i-p-1} as the third point to compute $\overline{V}^n_{i-1(n+1)}$, $\overline{V}^n_{i(n+1)}$ and $\overline{V}_{i+1(n+1)}$ respectively. If d is non-positive, we choose the third point in the other direction. Without loss of generality, we may assume that d is positive, so that we have

$$\begin{aligned}
\overline{V}^n_{i-1(n+1)} &= \overline{V}^n_{i-p-3}\psi_{i-1,i-p-3} + \overline{V}^n_{i-p-2}\psi_{i-1,i-p-2} + \overline{V}^n_{i-p-1}\psi_{i-1,i-p-1}, \\
\overline{V}^n_{i(n+1)} &= \overline{V}^n_{i-p-2}\psi_{i,i-p-2} + \overline{V}^n_{i-p-1}\psi_{i,i-p-1} + \overline{V}^n_{i-p}\psi_{i,i-p}, \\
\overline{V}^n_{i+1(n+1)} &= \overline{V}^n_{i-p-1}\psi_{i+1,i-p-1} + \overline{V}^n_{i-p}\psi_{i+1,i-p} + \overline{V}^n_{i-p+1}\psi_{i+1,i-p+1},
\end{aligned} \tag{C.5}$$

where ψ's are the Lagrangian basis functions. In general, let $\psi_{h,m}$ denote the basis function at the grid point x_m used to compute $\overline{V}^n_{h(n+1)}$:

$$\begin{aligned}
\psi_{h,m}(x) &= \frac{(x - x_{m+1})(x - x_{m+2})}{(x_m - x_{m+1})(x_m - x_{m+2})}, \text{ if } h - m = p + 2, \\
\psi_{h,m}(x) &= \frac{(x - x_{m-1})(x - x_{m+1})}{(x_m - x_{m-1})(x_m - x_{m+1})}, \text{ if } h - m = p + 1, \\
\psi_{h,m}(x) &= \frac{(x - x_{m-2})(x - x_{m-1})}{(x_m - x_{m-2})(x_m - x_{m-1})}, \text{ if } h - m = p.
\end{aligned} \tag{C.6}$$

Typically, we are interested in $\psi_{h,m}$'s such that $h = i-1, i, i+1$ and $m = i-p-3, \ldots, i-p+1$ as in (C.5).

Let $\overline{V}^n = [\overline{V}^n_i, \ldots, \overline{V}^n_{i_{\max}}]'$ to be the vector of discrete solution values. Substituting (C.5) into (C.4), we obtain the following equation

$$\begin{aligned}
&\overline{V}^{n+1}_i \left[1 + \alpha\Delta\tau + (r+\lambda)\frac{\Delta\tau}{2}\right] - \frac{\Delta\tau}{2}\alpha\overline{V}^{n+1}_{i+1} - \frac{\Delta\tau}{2}\alpha\overline{V}^{n+1}_{i-1} - \frac{\Delta\tau}{2}(\overline{V} \otimes \hat{\gamma})^{n+1}_i \\
=&\overline{V}^n_{i-p-1}\left[\left(1 - \alpha\Delta\tau - (r+\lambda)\frac{\Delta\tau}{2}\right)\psi_{i,i-p-1} + \alpha\frac{\Delta\tau}{2}\psi_{i+1,i-p-1} + \alpha\frac{\Delta\tau}{2}\psi_{i-1,i-p-1}\right] + \\
&\overline{V}^n_{i-p-2}\left[\left(1 - \alpha\Delta\tau - (r+\lambda)\frac{\Delta\tau}{2}\right)\psi_{i,i-p-2} + \alpha\frac{\Delta\tau}{2}\psi_{i-1,i-p-2}\right] + \\
&\overline{V}^n_{i-p}\left[\left(1 - \alpha\Delta\tau - (r+\lambda)\frac{\Delta\tau}{2}\right)\psi_{i,i-p} + \alpha\frac{\Delta\tau}{2}\psi_{i+1,i-p}\right] + \\
&\overline{V}^n_{i-p+1}\left[\alpha\frac{\Delta\tau}{2}\psi_{i+1,i-p+1}\right] + \overline{V}^n_{i-p-3}\left[\alpha\frac{\Delta\tau}{2}\psi_{i-1,i-p-3}\right] + \\
&(\overline{V} \otimes \hat{\gamma})_{i-p-2}\frac{\Delta\tau}{2}\psi_{i,i-p-2} + (\overline{V} \otimes \hat{\gamma})_{i-p-1}\frac{\Delta\tau}{2}\psi_{i,i-p-1} + (\overline{V} \otimes \hat{\gamma})_{i-p}\frac{\Delta\tau}{2}\psi_{i,i-p}.
\end{aligned} \tag{C.7}$$

We apply the discrete Fourier transform to (C.7) by defining the inverse discrete Fourier transform as follows

$$\overline{V}_i^n = \frac{1}{L}\sum_{k=0}^{N-1} e^{\frac{2\pi}{N}ik\sqrt{-1}} C_k^n, \qquad \hat{\gamma}_i = \frac{1}{L}\sum_{l=0}^{N-1} e^{\frac{2\pi}{N}il\sqrt{-1}} G_l, \tag{C.8}$$

where C_k^n and G_l are the discrete Fourier coefficients of $\overline{V}^n$ and $\hat{\gamma}$, $L = y_{\max} - y_{\min}$. Then the discrete correlation can be written as

$$(\overline{V} \otimes \hat{\gamma})_i^n = \frac{1}{L}\sum_{k=0}^{N-1} C_k^n G_{-k} e^{\frac{2\pi}{N}ik\sqrt{-1}}. \tag{C.9}$$

Substituting (C.8) and (C.9) into (C.7), we obtain, for each Fourier component,

$$\begin{aligned}
&C_k^{n+1}\big(1+\alpha\Delta\tau+(r+\lambda)\frac{\Delta\tau}{2}-\alpha\frac{\Delta\tau}{2}e^{\frac{2\pi}{N}k\sqrt{-1}}-\alpha\frac{\Delta\tau}{2}e^{-\frac{2\pi}{N}k\sqrt{-1}}-\frac{\Delta\tau}{2}G_{-k}\big)\\
&= C_k^n\Bigg(e^{-\frac{2\pi}{N}(p+1)k\sqrt{-1}}\Big[\big(1-\alpha\Delta\tau-(r+\lambda-G_{-k})\frac{\Delta\tau}{2}\big)\psi_{i,i-p-1}+\alpha\frac{\Delta\tau}{2}\psi_{i+1,i-p-1}+\alpha\frac{\Delta\tau}{2}\psi_{i-1,i-p-1}\Big]\\
&\quad+e^{-\frac{2\pi}{N}(p+2)k\sqrt{-1}}\Big[\big(1-\alpha\Delta\tau-(r+\lambda-G_{-k})\frac{\Delta\tau}{2}\big)\psi_{i,i-p-2}+\alpha\frac{\Delta\tau}{2}\psi_{i-1,i-p-2}\Big]\\
&\quad+e^{-\frac{2\pi}{N}pk\sqrt{-1}}\Big[\big(1-\alpha\Delta\tau-(r+\lambda-G_{-k})\frac{\Delta\tau}{2}\big)\psi_{i,i-p}+\alpha\frac{\Delta\tau}{2}\psi_{i+1,i-p}\Big]\\
&\quad+e^{-\frac{2\pi}{N}(p-1)k\sqrt{-1}}\big[\alpha\frac{\Delta\tau}{2}\psi_{i+1,i-p+1}\big]+e^{-\frac{2\pi}{N}(p+3)k\sqrt{-1}}\big[\alpha\frac{\Delta\tau}{2}\psi_{i-1,i-p-3}\big]\Bigg).
\end{aligned}\tag{C.10}$$

After simplification, we can rewrite the right hand side of (C.10) as

$$\begin{aligned}
&C_k^n\Bigg(\big(1-\alpha\Delta\tau-(r+\lambda-G_{-k})\frac{\Delta\tau}{2}\big)\sum_{m=i-p-2}^{i-p}\psi_{i,m}e^{-\frac{2\pi}{N}(i-m)k\sqrt{-1}}+\\
&\quad\alpha\frac{\Delta\tau}{2}\big[e^{\frac{2\pi}{N}k\sqrt{-1}}\sum_{m=i-p-1}^{i-p+1}\psi_{i+1,m}e^{-\frac{2\pi}{N}(i+1-m)k\sqrt{-1}}+e^{-\frac{2\pi}{N}k\sqrt{-1}}\sum_{m=i-p-3}^{i-p-1}\psi_{i-1,m}e^{-\frac{2\pi}{N}(i-1-m)k\sqrt{-1}}\big]\Bigg).
\end{aligned}$$

Since, by the definition of $\psi_{h,m}$ in (C.6),

$$\psi_{h,m} = \psi_{h+1,m+1}, \tag{C.11}$$

we have

$$\sum_{m=i-p-1}^{i-p+1}\psi_{i+1,m}e^{-\frac{2\pi}{N}(i+1-m)k\sqrt{-1}} = \sum_{m=i-p-2}^{i-p}\psi_{i,m}e^{-\frac{2\pi}{N}(i-m)k\sqrt{-1}} = \sum_{m=i-p-3}^{i-p-1}\psi_{i-1,m}e^{-\frac{2\pi}{N}(i-1-m)k\sqrt{-1}}.$$

Now we define

$$\rho_i \equiv \sum_{m=i-p-2}^{i-p}\psi_{i,m}e^{-\frac{2\pi}{N}(i-m)k\sqrt{-1}},$$

and derive the following equation

$$\frac{|C_k^{n+1}|}{|C_k^n|} = \frac{\left|\left(1-\alpha\Delta\tau-(r+\lambda-G_{-k})\frac{\Delta\tau}{2}+\alpha\frac{\Delta\tau}{2}\left(e^{\frac{2\pi}{N}k\sqrt{-1}}+e^{-\frac{2\pi}{N}k\sqrt{-1}}\right)\right)\rho_i\right|}{\left|1+\alpha\Delta\tau+(r+\lambda-G_{-k})\frac{\Delta\tau}{2}-\alpha\frac{\Delta\tau}{2}(e^{\frac{2\pi}{N}k\sqrt{-1}}+e^{-\frac{2\pi}{N}k\sqrt{-1}})\right|}. \tag{C.12}$$

It has been proved that for the case of constant coefficient equations with evenly spaced grid points and periodic boundary conditions, if the quadratic interpolation scheme satisfies (C.11), then $|\rho_i| \le 1$ (see [22] for details). Thus,

$$\begin{aligned}\frac{|C_k^{n+1}|^2}{|C_k^n|^2} &\le \frac{\left|1-r\frac{\Delta\tau}{2}-(\lambda-G_{-k})\frac{\Delta\tau}{2}-\alpha\Delta\tau\left(1-\cos\left(\frac{2\pi}{N}k\right)\right)\right|^2}{\left|1+r\frac{\Delta\tau}{2}+(\lambda-G_k)\frac{\Delta\tau}{2}+\alpha\Delta\tau\left(1-\cos(\frac{2\pi}{N}k)\right)\right|^2}\\ &= \frac{\left[1-r\frac{\Delta\tau}{2}-(\lambda-G_{-k}^R)\frac{\Delta\tau}{2}-\alpha\Delta\tau\left(1-\cos\left(\frac{2\pi}{N}k\right)\right]^2+\left[\frac{\Delta\tau}{2}G_{-k}^I\right]^2\right.}{\left[1+r\frac{\Delta\tau}{2}+(\lambda-G_{-k}^R)\frac{\Delta\tau}{2}+\alpha\Delta\tau\left(1-\cos(\frac{2\pi}{N}k)\right)\right]^2+\left[\frac{\Delta\tau}{2}G_{-k}^I\right]^2},\end{aligned}$$

where

$$G_{-k}^R = \mathrm{Re}(G_{-k}), \qquad G_{-k}^I = \mathrm{Im}(G_{-k}).$$

Note that

$$G_{-k} = \frac{L}{N}\sum_{j=0}^{N-1}\hat{\gamma}_j e^{\frac{2\pi}{N}kj\sqrt{-1}}.$$

From (3.11) we have

$$\frac{L}{N}\sum_{j=0}^{N-1}\hat{\gamma}_j \le \lambda,$$

so that $|G_{-k}| \le \lambda$, and hence $-\lambda \le G_{-k}^R \le \lambda$. Further note that $r \ge 0$ and $1-\cos(\frac{2\pi}{N}k) \ge 0$. It then follows that, $\forall k \in [-N/2+1, N/2]$,

$$\left|1+\frac{r\Delta\tau}{2}+\frac{\Delta\tau}{2}(\lambda-G_{-k}^R)+\Delta\tau\alpha\left(1-\cos(\frac{2\pi}{N}k)\right)\right| > \left|1-\frac{r\Delta\tau}{2}-\frac{\Delta\tau}{2}(\lambda-G_{-k}^R)-\Delta\tau\alpha\left(1-\cos(\frac{2\pi}{N}k)\right)\right|,$$

and consequently $|C_k^{n+1}| < |C_k^n|$, $\forall k$. As a result the scheme is unconditionally strictly stable.

References

[1] R. Cont and P. Tankov. *Financial Modelling With Jump Processes*. Chapman & Hall / CRC Press, 2003.

[2] P. Carr, H. Geman, D. B. Madan, and M. Yor. The fine structure of asset returns: An empirical investigation. *Journal of Business*, 75(2):305–332, 2002.

[3] P. Carr, H. Geman, D. B. Madan, and M. Yor. Stochastic volatility for Lévy processes. *Mathematical Finance*, 13(3):345–382, 2003.

[4] E. Eberlein. Application of generalized hyperbolic Lévy motion to finance. In O. Barndorff-Nielsen, T. Mikosch, and S. Resnick, editors, *Lévy Processes – Theory and Application*, pages 319–337, Birkhäuser, Boston, 2001.

[5] D. B. Madan, P. Carr, and E. Change. The Variance Gamma process and option pricing. *European Finance Rev.*, 2:79–105, 1998.

[6] D. Madan. Financial modeling with discontinuous price processes. In T. Mikosch O. Barndorff-Nielsen and S. Resnick, editors, *Lévy Processes – Theory and Application*, Birkhäuser, Boston, 2001.

[7] W. Schoutens. *Lévy Processes in Finance: Pricing Financial Derivatives*. Wiley, New Yourk, 2003.

[8] D. B. Madan and E. Seneta. The Variance Gamma (VG) model for share market returns. *J. Business*, 63:511–524, 1990.

[9] A. Almendral and C. W. Oosterlee. On American options under the Variance Gamma process. *Working paper, Delft University of Technology*, 2005.

[10] Y. D'Halluin, P. Forsyth, and G. Labahn. A penalty method for American options with jump-diffusion processes. *Numericshe Matematik*, 97:321–352, 2004.

[11] A. Hirsa and D. B. Madan. Pricing American options under Variance Gamma. *Journal of Computational Finance*, 7, 2004.

[12] Y. d'Halluin, P. A. Forsyth, and K. Vetzal. Robust numerical methods for contingent claims under jump diffusion processes. *IMA Journal on Numerical Analysis*, 25:87–112, 2005.

[13] A. Almendral. Numerical valuation of American options under the CGMY process. In A. Kyprianou W. Schoutens and P. Wilmott, editors, *Exotic Option Pricing and Advanced Lévy Models*, Wiley, UK, 2005.

[14] R. Cont and E. Voltchkova. A finite difference scheme for option pricing in jump diffusion and exponential Lévy models. *SIAM Journal on Numerical Analysis*, 43(4):1596–1626, 2005.

[15] H. Geman. Pure jump Lévy processes for asset price modeling. *Journal of Banking and Finance*, 26:1297–1316, 2002.

[16] A. M. Matache, P. A. Nitsche, and C. Schwab. Wavelet Galerkin pricing of American options on Lévy driven assets. *Quantitative Finance*, 5(4):403–424, 2005.

[17] S. Levendorskiĭ, O. Kudryavtsev, and V. Zherder. The relative efficiency of numerical methods for pricing American options under Lévy processes. *Journal of Computational Finance*, 9(2):69–97, 2005/06.

[18] A. Almendral and C. W. Oosterlee. Numerical valuation of options with jumps in the underlying. *Appl. Numer. Math.*, 53:1–18, 2005.

[19] G. Barles and P. E. Souganidis. Convergence of approximation schemes for fully nonlinear equations. *Asymptotic Analysis*, 4:271–283, 1991.

[20] H. A. van der Vorst. Bi-CGSTAB: A fast and smoothly converging variant of Bi-CG for the solution of non-symmetric linear systems. *SIAM Journal on Scientific and Statistical Computing*, 13:631–644, 1992.

[21] P. Carr and D. Madan. Option valuation using the fast Fourier transform. *J. Comput. Finance*, 2:61–73, 1998.

[22] M. Falcone and R. Ferretti. Convergence analysis for a class of high-order semi-Lagrangian avection schemes. *SIAM Journal on Numerical Analysis*, 35(3):909–940, 1998.

[23] Y. d'Halluin, P. A. Forsyth, and G. Labahn. A semi-Lagrangian approach for American Asian options under jump diffusion. *Journal of Scientific Computing*, 27:315–345, 2005.

[24] R. Bermejo. Analysis of a class of quasi-monotone and conservative semi-Lagrangian advection. *Numerische Mathematik*, 87:597–623, 2001.

[25] H. Pham. Optimal stopping of controlled jump diffusion process: a viscosity solution approach. *Journal of Mathematical Systems, Estimation and Control*, 8:1–27, 1998.

[26] P. A. Forsyth and K. R. Vetzal. Quadratic convergence of a penalty method for valuing American options. *SIAM Journal on Scientific Computation*, 23:2096–2123, 2002.

[27] R. Zvan, P. A. Forsyth, and K. R. Vetzal. Penalty methods for American options with stochastic volatility. *Journal of Computational and Applied Mathematics*, 91:199–218, 1998.

[28] R. D. Richtmyer and K. W. Morton. *Difference Methods for Initial-Value Problems*. Interscience Publishers, New York, 1967.

[29] G. Golub and C. van Loan. *Matrix Computations*. Johns Hopkins, 1991.

[30] R. Barrett, M. Berry, T. F. Chan, J. Demmel, J. Donato, J. Dongarra, V. Eijkhout, R. Pozo, C. Romine, and H. Van der Vorst. *Templates for the Solution of Linear Systems: Building Blocks for Iterative Methods*. SIAM, Philadelphia, PA, 1994.

[31] M. Frigo and S. G. Johnson. The design and implementation of FFTW3. *Proceedings of The IEEE*, 93(2):216–231, 2005.

[32] R. W. Lee. Option pricing by transform methods: Extensions, unification, and error control. *J. Comput. Finance*, 7(3):51–86, 2004.

[33] R. Lord, F. Fang, F. Bervoets, and C. W. Oosterlee. A fast and accurate FFT-based method for pricing early-exercise options under Lévy processes. *Working paper, Delft University of Technology*, 2007.