Operations Research
Vol. 56, No. 2, March–April 2008, pp. 304–325
ISSN 0030-364X | EISSN 1526-5463 | 08 | 5602 | 0304
DOI 10.1287/opre.1070.0419
© 2008 INFORMS

Pricing Options in Jump-Diffusion Models: An Extrapolation Approach

Liming Feng
Department of Industrial and Enterprise Systems Engineering, University of Illinois at Urbana-Champaign, Urbana, Illinois 61801, fenglm@uiuc.edu

Vadim Linetsky
Department of Industrial Engineering and Management Sciences, McCormick School of Engineering and Applied Sciences, Northwestern University, Evanston, Illinois 60208, linetsky@iems.northwestern.edu

We propose a new computational method for the valuation of options in jump-diffusion models. The option value function for European and barrier options satisfies a partial integrodifferential equation (PIDE). This PIDE is commonly integrated in time by implicit-explicit (IMEX) time discretization schemes, where the differential (diffusion) term is treated implicitly, while the integral (jump) term is treated explicitly. In particular, the popular IMEX Euler scheme is first-order accurate in time. Second-order accuracy in time can be achieved by using the IMEX midpoint scheme. In contrast to the above approaches, we propose a new high-order time discretization scheme for the PIDE based on the extrapolation approach to the solution of ODEs that also treats the diffusion term implicitly and the jump term explicitly. The scheme is simple to implement, can be added to any PIDE solver based on the IMEX Euler scheme, and is remarkably fast and accurate. We demonstrate our approach on the examples of Merton's and Kou's jump-diffusion models, the diffusion-extended variance gamma model, as well as the two-dimensional Duffie-Pan-Singleton model with correlated and contemporaneous jumps in the stock price and its volatility. By way of example, pricing a one-year double-barrier option in Kou's jump-diffusion model, our scheme attains accuracy of 10^{-5} in 72 time steps (in 0.05 seconds). In contrast, it takes the first-order IMEX Euler scheme more than 1.3 million time steps (in 873 seconds) and the second-order IMEX midpoint scheme 768 time steps (in 0.49 seconds) to attain the same accuracy. Our scheme is also well suited for Bermudan options. Combining simplicity of implementation and remarkable gains in computational efficiency, we expect this method to be very attractive to financial engineering modelers.

Subject classifications: finance, asset pricing: option pricing, barrier options, Bermudan options; probability, Markov processes: jump-diffusion processes.
Area of review: Financial Engineering.
History: Received January 2006; revisions received August 2006, September 2006, October 2006; accepted November 2006. Published online in *Articles in Advance* November 1, 2007.

1. Introduction

Since Merton's (1976) pioneering work, jump-diffusion processes have become a standard modeling tool in equity, foreign exchange, fixed income, commodity, and energy derivatives markets. Jump-diffusion models extend the classical diffusion modeling framework by adding jumps to the diffusion dynamics of the state variable. Jump diffusions are capable of modeling large and sudden changes in the state variable. Jump-diffusion models naturally exhibit high skewness and leptokurtosis levels typically observed in financial time series. At the same time, options markets typically exhibit the so-called *implied volatility smile* or *skew* effects across options with different strike prices. In the jump-diffusion framework, these are naturally generated by jumps in the underlying process (e.g., Andersen and Andreasen 2000; Bates 1996, 2000; Duffie et al. 2000; Kou 2002).

In Merton's (1976) model, the asset return follows a Brownian motion with drift punctuated by jumps arriving according to a compound Poisson process with constant intensity and with normally distributed jump sizes. Due to normality of the jump-size distribution, Merton was able to obtain explicit analytical solutions for European call and put options in this model. Kou (2002) recently proposed a double exponential jump-diffusion model where jump sizes are double exponentially distributed. One of the features of this model is that some path-dependent options, such as barrier and lookback options, can be priced analytically due to the memoryless property of the exponential distribution (Kou and Wang 2003, 2004).

More flexibility to model empirical financial data is obtained when the instantaneous variance of the diffusion component of the asset return process is allowed to be random. Heston (1993) considered a stochastic volatility (SV) model, where the variance is stochastic and follows the Cox-Ingersoll-Ross (CIR) square-root process. This model was extended by Bates (1996) by allowing jumps in the asset return process (stochastic volatility with jumps (SVJ) in return) and by Duffie et al. (2000) by allowing jumps both in the asset return process and in the variance process (stochastic volatility with correlated and contemporaneous jumps in return and variance (SVCJ)). Eraker et al. (2003) examine the SVCJ model empirically and find strong evidence for jumps both in the asset return and in the variance processes. Broadie and Kaya (2005) develop efficient Monte Carlo simulation methods for the SVCJ model.

When the jump-diffusion process is a Lévy process, such as in Merton's (1976) and Kou's (2002) models, or an affine jump-diffusion process, such as in the SVJ and SVCJ models based on Heston's SV model, the characteristic function of the process can be obtained in closed form and European options can be priced efficiently by inverting the Fourier transform by the fast Fourier transform algorithm (FFT) (Carr and Madan 1999, Duffie et al. 2000). However, this approach does not apply to path-dependent options such as barrier options.

For discrete barrier options under Merton's (1976) jump-diffusion model, Broadie and Yamamoto (2005) develop an efficient method based on the fast Gauss transform (previously introduced to options pricing in Broadie and Yamamoto 2003). However, their method does not directly apply to non-Gaussian jump-size distributions. For discrete barrier options under Lévy processes, Petrella and Kou (2004) develop a method based on Spitzer's identity and Laplace transform and Feng and Linetsky (2007) develop a method based on the Hilbert transform. However, their methods do not apply to more general jump-diffusion processes not in the Lévy class, and do not apply to continuous barrier options and Bermudan options. Furthermore, if the process parameters such as volatility, drift, jump intensity, and jump-size distribution are state dependent (such as in, e.g., Glasserman and Merener 2003), the characteristic function is not available analytically, and even European options have to be priced by numerical methods.

The option price (value function) for European and barrier options satisfies a partial integrodifferential equation (PIDE). This PIDE is commonly integrated in time by the implicit-explicit (IMEX) Euler time discretization, where the differential (diffusion) part is treated implicitly, while the integral (jump) part is treated explicitly. This approach has been widely applied in the literature (see Andersen and Andreasen 2000, Carr and Hirsa 2003, Cont and Voltchkova 2005, d'Halluin et al. 2005, Hirsa and Madan 2003, Tavella and Randall 2000, and Zhang 1997), and is very popular in practice. The desire to treat the integral (jump) term in the PIDE explicitly is motivated by the fact that the fully implicit treatment would necessitate inverting a dense matrix resulting from discretizing the integral in contrast to the banded matrix resulting from the spatial discretization of the diffusion operator (tridiagonal for one-dimensional diffusions and nine-diagonal for two-dimensional diffusions). However, the fully explicit treatment of the PIDE (e.g., binomial or trinomial lattices; see Amin 1993) is inefficient due to the stability restriction $\Delta t \leqslant C\Delta x^2$ on the size of the time step, because prohibitively small time steps are required to attain higher accuracy goals. The IMEX Euler scheme avoids both problems. It turns out that the explicit treatment of the integral term (at least for finite activity jump processes) does not generate any stability restrictions (the unconditional stability is proved in d'Halluin et al. 2005) and at the same time avoids inverting dense matrices. This combination of simplicity of implementation and stability secured the popularity of the IMEX Euler scheme among financial engineering practitioners.

However, the IMEX Euler scheme is only first-order accurate in time. This results in slow convergence even for one-dimensional problems. It is desirable to develop high-order time-stepping schemes for the jump-diffusion PIDE. In one dimension, Andersen and Andreasen (2000) propose a second-order accurate in time, stable operator-splitting ADI finite-difference scheme based on the application of the FFT. However, it is not straightforward to generalize this method to multidimensional jump-diffusion processes and processes with state-dependent jump magnitude distributions. Also in one dimension, d'Halluin et al. (2005) develop a fully implicit stable, second-order in time finite-difference scheme for the PIDE that utilizes Crank-Nicolson time stepping. Their scheme is more general than Andersen and Andreasen's (2000) scheme because it can be generalized to handle early exercise (d'Halluin et al. 2004) and exotic features. However, it is more implementationally involved due to the implicit handling of the integral term (a fixed-point iterative procedure is involved in solving the dense system of equations) than the simple IMEX Euler scheme. Furthermore, extensions to multidimensional jump-diffusion processes, processes with state-dependent jump magnitude distributions, and processes with discontinuous jump magnitude densities may not be straightforward. In particular, d'Halluin et al. (2004) report difficulties with attaining second-order convergence in space in Kou's (2002) double-exponential model with the double-exponential jump magnitude density discontinuous at zero (empirically the most relevant case because it produces skewness in the asset return distribution necessary to generate the implied volatility skew).

With increasing interest in multidimensional models such as the SVCJ with jumps both in the stock return and in the return variance and multiasset models with correlated jumps, the problem of developing efficient and relatively easy to implement numerical methods for general multidimensional jump-diffusion models is becoming increasingly important in financial engineering practice. Unfortunately, the first-order IMEX Euler scheme is prohibitively slow in multiple dimensions.

In this paper, we propose a new high-order time discretization scheme for the PIDE based on the extrapolation approach to integration of ODEs that treats the diffusion term implicitly and the jump term explicitly. The scheme is simple to implement, can be added to any existing PIDE solver based on the IMEX Euler scheme, and is remarkably fast and accurate. We demonstrate our approach on the examples of Merton's (1976) and Kou's (2002) jump-diffusion models, the diffusion-extended variance gamma (DEVG) model, as well as on the two-dimensional SVCJ model. By way of example, pricing a one-year double-barrier option in Kou's jump-diffusion model, our extrapolation scheme attains accuracy of 10^{-5} in 72 time steps (in 0.05 seconds). In contrast, it takes the first-order IMEX Euler scheme more than 1.3 million time steps (in 873 seconds) and the second-order IMEX midpoint scheme 768 time steps (in 0.49 seconds) to attain this accuracy in the same example. Combining simplicity of implementation and remarkable gains in computational efficiency, we expect this scheme to be very attractive to financial engineers.

This paper is organized as follows. In §2, we briefly review Merton's (1976), Kou's (2002), and SVCJ jump-diffusion models, and the corresponding PIDE formulations for European and single- and double-barrier options in these models. In §3, we introduce our spatial discretization approach. In this paper, we follow the Galerkin finite-element approach to discretize the variational (weak) form of the PIDE. We use the standard finite-element basis in one and two dimensions that ensures quadratic convergence in space. The result of the spatial discretization is a system of ordinary differential equations (ODE) to be integrated numerically. This approach is called the *finite-element method-of-lines* (MOL). In §4, we describe an extrapolation time-stepping scheme based on the IMEX Euler scheme that treats the (stiff) term in the ODE resulting from the differential terms in the original PIDE implicitly and the (nonstiff) term in the ODE resulting from the integral term in the PIDE explicitly. At each time step, the computation involves matrix-vector multiplication, where the matrix is Toeplitz for one-dimensional models, and block Toeplitz with Toeplitz blocks for the two-dimensional SVCJ model. FFT is used to perform these multiplications efficiently.

In the literature on numerical methods for stiff ODEs, extrapolation-based ODE solvers for stiff systems are surveyed by Deuflhard (1985), Deuflhard and Bornemann (2002, §6.4.2), and Hairer and Wanner (1996, §IV.9). In computational finance, Richardson extrapolation has been widely applied in the context of binomial and trinomial lattice approximations in the pure diffusion Black-Scholes model (e.g., Broadie and Detemple 1996, Ibanez 2003). The application of extrapolation we propose here is different, because it involves a time-stepping scheme for the system of stiff ODEs arising after the spatial discretization of the PIDE in general jump-diffusion models.

We emphasize that the time-stepping extrapolation scheme of §4 equally applies to any of the possible spatial discretization approaches, such as finite-difference discretizations of the PIDE (e.g., Andersen and Andreasen 2000, Cont and Voltchkova 2005, d'Halluin et al. 2005), finite-difference discretizations of the variational (weak) formulation of the PIDE (e.g., Zhang 1997), Galerkin finite-element methods (as in this paper), finite-volume methods, wavelet Galerkin methods (Matache et al. 2004; 2005a, b; 2006), or spectral methods. Our decision to implement the finite-element spatial discretization in this study was dictated by simplicity and robustness of handling the integral term and boundary conditions (we are interested both in European and barrier options in this paper) in the finite-element formulation. In particular, we do not observe any difficulties with Kou's (2002) double-exponential model with discontinuous jump-size density in our approach, and the treatment of the two-dimensional SVCJ model with bivariate jumps is straightforward as well. Nevertheless, the contribution of §4 describing our time-stepping approach is independent of the choice of spatial discretization and should equally apply to other spatial discretization choices. This is important for financial engineering practice. The time-stepping approach of §4 can be coupled with any existing spatial discretization software already in use in organizations. In §5, we consider further applications and extensions of our approach, including the pricing of Bermudan options and extensions to processes with infinite activity jumps. Section 6 presents extensive computational tests of our approach and demonstrates remarkable efficiency gains resulting from the extrapolation based on the IMEX Euler scheme. Section 7 concludes the paper. The appendix in the online companion contains some results on Toeplitz matrices and block Toeplitz matrices with Toeplitz blocks used in our implementation. An electronic companion to this paper is available as part of the online version that can be found at http://or.journal.informs.org/.

2. Option Pricing in Jump-Diffusion Models

2.1. Jump-Diffusion Processes

We take an equivalent martingale measure (EMM) as given and model the risk-neutral stock price process in the form $S_t = Ke^{X_t}$, $t \geqslant 0$, where $\{X_t,\ t \geqslant 0\}$ is a jump-diffusion process and $K > 0$ is some reference price level. One typically sets $K = S_0$, the initial stock price at time zero. This corresponds to starting the process X_t at the origin, $X_0 = 0$ (X_t has the interpretation of the continuously compounded return process net of dividends). Alternatively, when pricing call and put options, it will be convenient for us to set K equal to the strike price of the option to be priced. This corresponds to starting the process X_t at $X_0 = x := \ln(S_0/K)$.

In one dimension, we assume that

$$dX_t = \mu\, dt + \sigma\, dB_t + dJ_t,$$
$$\mu = r - q - \sigma^2/2 + \lambda(1 - \mathbb{E}[e^Z]),$$

where $r \geqslant 0$ is the risk-free interest rate, $q \geqslant 0$ is the continuous dividend yield, $\sigma > 0$ is the stock return volatility,

$\{B_t,\, t \geqslant 0\}$ is a standard Brownian motion, and $\{J_t,\, t \geqslant 0\}$ is a jump process, a compound Poisson process with intensity $\lambda > 0$ and a given jump-size (magnitude) distribution, i.e., $J_t = \sum_{n=1}^{N_t} Z_n$, where N_t is a Poisson process with intensity λ and $\{Z_n\}$ are independent and identically distributed (i.i.d.) jump magnitudes. It is also assumed that the Brownian motion, the Poisson process, and the jump magnitudes are all independent. The drift constant μ is adjusted so that the total gains process, including price changes and dividends, is a martingale under the EMM, i.e., $\mathbb{E}[S_t] = e^{(r-q)t} S_0$ for each $t > 0$ for the price process.

In Merton's (1976) model, the jump magnitude distribution is normal with mean m and standard deviation s with the probability density

$$p(z) = \frac{1}{\sqrt{2\pi s^2}} \exp\left(-\frac{(z-m)^2}{2s^2}\right).$$

In this model, the drift parameter is $\mu = r - q - \sigma^2/2 + \lambda[1 - \exp(m + s^2/2)]$. In Kou's (2002) model, the jump magnitude distribution is double exponential with the density

$$p(z) = p\eta_1 e^{-\eta_1 z} 1_{\{z \geqslant 0\}} + (1-p)\eta_2 e^{\eta_2 z} 1_{\{z<0\}}$$

and $\mu = r - q - \sigma^2/2 + \lambda[(1-p)(\eta_2+1)^{-1} - p(\eta_1 - 1)^{-1}]$. In this model, positive jumps occur with probability p and are exponentially distributed with mean $1/\eta_1$ with $\eta_1 > 1$, and negative jumps occur with probability $1-p$ and are exponentially distributed with mean $1/\eta_2$ with $\eta_2 > 0$.

The infinitesimal generator of the Markov jump-diffusion process X_t in one-dimensional jump-diffusion models is

$$\mathcal{G}f(x) = \tfrac{1}{2}\sigma^2 f_{xx} + \mu f_x + \lambda \int_{\mathbb{R}} [f(x+z) - f(x)] p(z)\, dz,$$

where $p(z)$ is the jump magnitude probability density.

In the one-dimensional model, the volatility σ is a constant. In contrast, in the SVCJ model, the instantaneous variance $V_t = \sigma_t^2$ is assumed to follow a CIR diffusion punctuated by positive jumps. The corresponding two-dimensional stochastic differential equation (SDE) is (Duffie et al. 2000)

$$dX_t = (\mu - V_{t-}/2)dt + \sqrt{V_{t-}}\left[\sqrt{1-\rho_D^2}\, dB_t^1 + \rho_D\, dB_t^2\right] + dJ_t^X,$$
$$dV_t = \kappa(\theta - V_{t-})dt + \xi\sqrt{V_{t-}}\, dB_t^2 + dJ_t^V,$$

where θ is the long-run variance level, κ is the rate of mean reversion, ξ is the volatility-of-volatility parameter, B^1 and B^2 are two independent standard Brownian motions, ρ_D is the correlation coefficient correlating Brownian shocks in the return and variance processes, and (J_t^X, J_t^V) is a two-dimensional jump process, a $\mathbb{R} \times \mathbb{R}^+$–valued compound Poisson process with intensity $\lambda > 0$ and a bivariate jump magnitude distribution in $\mathbb{R} \times \mathbb{R}^+$. The process starts at $X_0 := x = \ln(S_0/K)$ and $V_0 = v > 0$ at time zero. The jump magnitudes (Z_n^X, Z_n^V) are i.i.d. with a joint bivariate probability density $p(z^x, z^v)$. The marginal distribution of the jump size in variance is assumed to be exponential with mean ν. Conditional on a jump of size z^v in the variance process, the jump size in the return process X_t is assumed to be normally distributed with mean $m + \rho_J z^v$ (where ρ_J defines correlation between jumps in return and variance) and standard deviation s. The bivariate density is

$$p(z^x, z^v) = \frac{1}{\nu\sqrt{2\pi s^2}} \exp\left(-\frac{z^v}{\nu} - \frac{(z^x - m - \rho_J z^v)^2}{2s^2}\right), \quad z^x \in \mathbb{R},\ z^v \geqslant 0.$$

The drift parameter is

$$\mu = r - q + \lambda[1 - (1 - \nu\rho_J)^{-1} \exp(m + s^2/2)].$$

The infinitesimal generator of the two-dimensional Markov process (X_t, V_t) is given by

$$\begin{aligned}&\mathcal{G}f(x,v)\\ &= \tfrac{1}{2}vf_{xx} + \rho\xi v f_{vx} + \tfrac{1}{2}\xi^2 v f_{vv} + \left(\mu - \tfrac{1}{2}v\right)f_x + \kappa(\theta - v)f_v\\ &\quad + \lambda\int_{-\infty}^{\infty}\int_0^{\infty} [f(x+z^x, v+z^v) - f(x,v)] p(z^x, z^v)\, dz^v\, dz^x.\end{aligned}$$

For future convenience, we introduce a scaled and centered dimensionless variance process $Y_t = (V_t - \theta)/\theta$. A jump of size δV in V_t corresponds to a jump of size $\delta Y = \delta V/\theta$ in Y_t. Hence, the joint distribution of jumps in the state variables (X_t, Y_t) has a density

$$p(z^x, z^y) = \frac{\theta}{\nu\sqrt{2\pi s^2}} \exp\left(-\frac{\theta z^y}{\nu} - \frac{(z^x - m - \rho_J \theta z^y)^2}{2s^2}\right), \quad z^x \in \mathbb{R},\ z^y \geqslant 0.$$

The infinitesimal generator of the two-dimensional Markov process (X_t, Y_t) is given by

$$\begin{aligned}&\mathcal{G}f(x,y)\\ &= \frac{1}{2}\theta(y+1)f_{xx} + \rho\xi(y+1)f_{yx} + \frac{\xi^2}{2\theta}(y+1)f_{yy}\\ &\quad + \left(\mu - \frac{1}{2}\theta(y+1)\right)f_x - \kappa y f_y\\ &\quad + \lambda\int_{-\infty}^{\infty}\int_0^{\infty} [f(x+z^x, y+z^y) - f(x,y)] p(z^x, z^y)\, dz^y\, dz^x.\end{aligned}$$

2.2. Option Pricing in Jump-Diffusion Models

Consider a European-style option contract that delivers a payoff $F(S_T)$ at expiration $T > 0$. The payoff function F is assumed to depend on the underlying stock price at expiration. The price of the option at time $t \in [0, T]$ is given by its expected discounted payoff, where the expectation is taken under the EMM:

$$V(t,x) = e^{-r(T-t)}\mathbb{E}_{t,x}[\psi(X_T)],$$

where $r \geqslant 0$ is the risk-free interest rate. Here, we substituted $S_T = Ke^{X_T}$ and defined the payoff function $\psi(X_T) = F(Ke^{X_T})$ in terms of the variable $X_T = \ln(S_T/K)$. The subscript in the conditional expectation operator $\mathbb{E}_{t,x}$ signifies that at time t the state of the process X is known, $X_t = x$. For call and put options, we set K equal to the strike price, and the payoffs are ($x^+ \equiv \max\{x, 0\}$): $\psi_{\text{call}}(x) = K(e^x - 1)^+$ and $\psi_{\text{put}}(x) = K(1 - e^x)^+$. The value function V can be characterized as the solution of the following PIDE (see, e.g., Cont and Tankov 2004):

$$V_t + \mathcal{G}V - rV = 0, \quad t \in [0, T),$$

with the terminal (payoff) condition $V(T, x) = \psi(x)$. For future convenience, we separate the differential (diffusion) and integral (jump) components in the infinitesimal generator $\mathcal{G}$. We denote the differential part of the operator (with discounting at the constant rate $r + \lambda$ included in the differential operator) by $\mathcal{A}$ and the integral part by $\mathcal{B}$. We also transform the terminal value problem into the initial value problem (IVP) by defining $U(t, x) := V(T - t, x)$. Then, we need to solve the following PIDE:

$$U_t - \mathcal{A}U - \mathcal{B}U = 0, \quad t \in (0, T],$$

with the initial condition: $U(0, x) = \psi(x)$. For one-dimensional jump-diffusion models, we have

$$\mathcal{A}U = \tfrac{1}{2}\sigma^2 U_{xx} + \mu U_x - (r + \lambda)U,$$

$$\mathcal{B}U(t, x) = \lambda \int_{\mathbb{R}} U(t, x + z) p(z)\, dz.$$

In this paper, we are also interested in *knock-out options*. Generally, a knock-out option contract is canceled (knocked out) if the underlying state variable exits a prespecified (open) domain Ω in the state space. Let τ be the first exit time of the underlying process from Ω (a first passage time into Ω^c, the complement of Ω in the state space). The knock-out option is then canceled (declared null and void) at time τ if $\tau \leqslant T$, where T is the option expiration. Otherwise, the option holder receives the payoff at expiration. In some cases, the option holder receives a *rebate* $R(X_\tau)$ at time τ if $\tau \leqslant T$. This rebate can be thought of as the "consolation prize" to mitigate the loss of the option contract. It can be constant or depend on the state of the underlying process at time τ (more generally, it can also depend on time, i.e., the rebate is $R(\tau, X_\tau)$ if the process exits Ω at time τ and is in state $X_\tau \in \Omega^c$ at τ, but for simplicity, we assume that it only depends on the state and is independent of time). Note that because we are dealing with jump diffusions, it is possible that the process will jump right through the boundary of Ω and straight into the interior of Ω^c (*overshoot*). Hence, the rebate needs to be specified for all states in Ω^c. This is in contrast with pure diffusions, which, by continuity, always hit the boundary.

In particular, we are interested in six types of knock-out options: calls and puts with lower barriers, calls and puts with upper barriers, and calls and puts with both lower and upper barriers. A down-and-out call (put) delivers a call (put) payoff at T if the stock does not fall to or below a lower barrier L, $0 < L < S_0$, prior to and including expiration T. An up-and-out call (put) delivers a call (put) payoff if the stock does not increase to or above the upper barrier U, $S_0 < U < \infty$, prior to and including expiration T. A double-barrier call (put) delivers a call (put) payoff at T if the stock does not exit from the interval (L, U) prior to and including expiration T. Here, L and U, $0 < L < S_0 < U < \infty$, are lower and upper barriers. In terms of the process $X_t = \ln(S_t/K)$, the lower and upper barriers are $\underline{x} = \ln(L/K)$ and $\bar{x} = \ln(U/K)$. In terms of the PIDE, the value function of the knock-out option satisfies the following *knock-out condition with rebate*: $U(t, x) = R(x)$, $x \in \Omega^c$, $t \in [0, T]$, where $\Omega^c = (-\infty, \underline{x}]$, $\Omega^c = [\bar{x}, \infty)$, and $\Omega^c = (-\infty, \underline{x}] \cup [\bar{x}, \infty)$ for down-and-out, up-and-out, and double-barrier options, respectively. If there is no rebate, $R = 0$, then the value function vanishes everywhere in Ω^c. Note that the knock-out condition has to be imposed *everywhere* in Ω^c because of the overshoot (the process can jump from Ω to any point in Ω^c; at the PIDE level, this manifests itself in the nonlocal nature of the integral operator). The PIDE is then solved in Ω subject to the initial condition at time zero and the knock-out condition with rebate R imposed everywhere in Ω^c. This is in contrast with diffusions, for which only a boundary condition on the boundary $\partial\Omega$ is needed.

In the two-dimensional SVCJ model, the value function also depends on the initial variance at time t, represented by the scaled and centered state variable $Y_t = y$, $U = U(t, x, y)$, and the differential and integral parts of the operator are

$$\mathcal{A}U = \frac{1}{2}\theta(y+1)U_{xx} + \rho_D \xi(y+1)U_{xy} + \frac{\xi^2}{2\theta}(y+1)U_{yy} + \left(\mu - \frac{1}{2}\theta(y+1)\right)U_x - \kappa y U_y - (r+\lambda)U,$$

$$\mathcal{B}U(t,x,y) = \lambda \int_{-\infty}^{\infty}\int_0^{\infty} U(t, x+z^x, y+z^y) p(z^x, z^y)\, dz^y\, dz^x.$$

The initial condition is $U(0, x, y) = \psi(x)$, where $\psi(x)$ is the payoff (which for call and put options depends only on the x variable). Finally, for knock-out options, the appropriate knock-out conditions are imposed as discussed previously.

3. Spatial Discretization

3.1. Localization to Bounded Domains

For European and single-barrier calls and puts in one-dimensional jump-diffusion models, the state space Ω is unbounded ($\Omega = (\underline{x}, \infty)$ for down-and-out options, $\Omega = (-\infty, \bar{x})$ for up-and-out options, and $\Omega = \mathbb{R}$ for European options without barriers). In the SVCJ model, Ω is unbounded even for double-barrier options due to the additional state variable $y \in (-1, \infty)$. To develop numerical approximations, we localize the original problem by considering an increasing exhausting sequence of bounded open domains $\{\Omega_k\}_{k=1}^{\infty}$ such that $\Omega_k \subset \Omega_{k+1}$ and $\bigcup_{k=1}^{\infty} \Omega_k = \Omega$. Then, the value function U of the original problem on the unbounded domain Ω is realized as the limit of a sequence of functions U_k that solve the localized PIDEs on bounded domains

$$U_{k,t} - \mathscr{A}U_k - \mathscr{B}U_k = 0, \quad t \in (0, T],\ x \in \Omega_k, \tag{3.1}$$

with the initial condition

$$U_k(0, x) = \psi(x), \quad x \in \Omega_k. \tag{3.2}$$

An *artificial knock-out condition* is imposed everywhere in $(\Omega_k)^c$:

$$U_k(t, x) = R(x), \quad x \in (\Omega_k)^c, \tag{3.3}$$

where $R(x)$ is the *artificial rebate*. In other words, we approximate the original option contract with an artificial option contract that knocks out when the process exits a bounded domain Ω_k and pays a rebate $R(X_\tau)$ at the first exit time τ. The economics of the problem typically dictate an appropriate choice of the artificial knock-out condition. For European options, the payoff function provides a reasonable choice for the artificial knock-out condition: $R(x) = \psi(x)$ for $x \in (\Omega_k)^c$, where in one-dimensional models $\Omega_k = (\underline{x}_k, \bar{x}_k)$ and $\underline{x}_k \to -\infty$, $\bar{x}_k \to \infty$ as $k \to \infty$ (in the SVCJ model, $\Omega_k = (\underline{x}_k, \bar{x}_k) \times (\underline{y}_k, \bar{y}_k)$). For down-and-out put options with the lower barrier L, the vanishing knock-out condition $R(x) = 0$, $x \notin \Omega_k$, provides a reasonable choice. In one-dimensional models, $\Omega_k = (\underline{x}, \bar{x}_k)$, where $\underline{x} = \ln(L/K)$ is determined by the contractual lower barrier L specified in the option contract and $\bar{x}_k \to \infty$ as $k \to \infty$ is the artificial upper barrier. For $x \geqslant \bar{x}_k$, it provides a reasonable choice for the artificial knock-out condition because the value function of the down-and-out put rapidly decreases toward zero for high stock prices. Other types of knock-out options are treated similarly.

For the localized problem, we have (see Bensoussan and Lions 1984 for general results on localization of PIDE problems to bounded domains)

$$\max_{t \in [0, T]} \|U(t, \cdot) - U_k(t, \cdot)\|_{L^\infty(G)} \to 0 \quad \text{as } k \to \infty$$

for any fixed compact set $G \in \Omega_k$. The set G is referred to as the *approximation domain*, where we are interested in the value function U. The bounded domain Ω_k is referred to as the *computational domain* (see Marcozzi 2001 for details in the diffusion setting). Kangro and Nicolaides (2000), Matache et al. (2004), and Hilber et al. (2005) show that the localization error decays exponentially in the size of the computational domain in the Black-Scholes setting, in the Lévy process setting, and in the stochastic volatility setting, respectively.

Other choices for artificial knock-out conditions can be used to localize the problem to a bounded domain. In this paper, we use the payoff function as the artificial rebate in the artificial knock-out condition for European options, down-and-out calls, and up-and-out puts ($R(x) = \psi(x)$) and vanishing knock-out conditions for double-barrier call and put options, down-and-out puts, and up-and-out calls ($R(x) = 0$) (for simplicity, we assume that the original knock-out contracts do not pay any contractual rebates). In the remainder of the paper, we take a bounded computational domain Ω_k as given and solve the PIDE on Ω_k with the artificial knock-out condition in $(\Omega_k)^c$. We drop the index k to lighten notation.

3.2. Variational Formulation

We consider a variational formulation of the problem (3.1)–(3.3) on a given bounded domain Ω. This PIDE may have a nonhomogeneous knock-out condition (with rebate $R(x) \neq 0$). We homogenize it as follows. The rebate $R(x)$ is defined everywhere in Ω^c. We extend it to Ω as well and also denote the extended version by R (in this paper, we either have $R(x) = 0$ or $R(x) = \psi(x)$, the option payoff). Let $u := U - R$ (for $R = \psi, u$ can be interpreted as the excess option premium over the payoff; for European options it can be negative because the exercise is not allowed until expiration).

A *variational* (*weak*) *formulation* of the PIDE problem is obtained by considering a space of test functions square-integrable on Ω, vanishing in Ω^c, and with their (weak) first derivatives square-integrable on Ω. Multiplying the PIDE with a test function $v = v(x)$, integrating over Ω, and integrating by parts, we arrive at the variational (weak) formulation of the PIDE:

$$(u_t, v) + a(u, v) - b(u, v) + a(R, v) - b(R, v) = 0, \tag{3.4}$$

$$(u(0, \cdot), v) = (\psi - R, v), \tag{3.5}$$

where $(u, v) = \int_\Omega u(x)v(x)\,dx$ is the inner product in $L^2(\Omega)$, and the two bilinear forms $a(\cdot, \cdot)$ and $b(\cdot, \cdot)$ in one-dimensional jump-diffusion models with $\Omega = (\underline{x}, \bar{x})$ are defined by

$$a(u,v) = \tfrac{1}{2}\sigma^2 \int_{\underline{x}}^{\bar{x}} u_x v_x\,dx - \mu \int_{\underline{x}}^{\bar{x}} u_x v\,dx + (r+\lambda) \int_{\underline{x}}^{\bar{x}} uv\,dx,$$

$$b(u, v) = (\mathscr{B}u, v) = \lambda \int_{\underline{x}}^{\bar{x}} \int_{-\infty}^{\infty} u(x+z)v(x)p(z)\,dz\,dx.$$

In the SVCJ model, when the computational domain is $\Omega = (\underline{x}, \bar{x}) \times (\underline{y}, \bar{y})$,

$$\begin{aligned} a(u,v) &= \int_{\underline{x}}^{\bar{x}} \int_{\underline{y}}^{\bar{y}} \left(\frac{1}{2}(y+1)\left(\theta u_x v_x + \rho\xi u_y v_x + \rho\xi u_x v_y + \frac{1}{\theta}\xi^2 u_y v_y \right) \right. \\ &\qquad + \left(ky + \frac{\xi^2}{2\theta} \right) u_y v \\ &\qquad \left. - \left(\mu - \frac{1}{2}\rho\xi - \frac{1}{2}\theta - \frac{1}{2}\theta y \right) u_x v + (r+\lambda)uv \right) dy\,dx, \end{aligned}$$

$$b(u,v)=\lambda\int_{\underline{x}}^{\bar{x}}\int_{\underline{y}}^{\bar{y}}\int_{-\infty}^{\infty}\int_{0}^{\infty}u(x+z^x,y+z^y)v(x,y)\cdot p(z^x,z^y)\,dz^y\,dz^x\,dy\,dx.$$

To solve the variational formulation, we seek a function $u=u(t,x)$ in an appropriate function space such that (3.4)–(3.5) hold for any test function v. The solution u vanishes outside of Ω. The value function U is then obtained by $U=R+u$.

More details on the variational formulation of parabolic PDEs associated with diffusion processes can be found in Quarteroni and Valli (1997) and Thomée (1997), where the relevant functional analytic background can be found. The variational formulation of PIDEs associated with jump-diffusion processes can be found in Bensoussan and Lions (1984). A variational formulation of Merton's (2002) jump-diffusion model can be found in Zhang (1997). Matache et al. (2004, 2005a) develop a variational formulation for Lévy process-based models. Hilber et al. (2005) develop a variational formulation for the Heston SV model.

3.3. Finite-Element Discretization: One-Dimensional Models

We now consider a spatial discretization of the variational formulation (3.4)–(3.5) by the finite-element method (see Ciarlet 1978, Johnson 1987, Larsson and Thomée 2003, Quarteroni and Valli 1997, and Thomée 1997 for textbook treatments of the finite-element method). Consider a one-dimensional problem on a bounded domain $\Omega=(\underline{x},\bar{x})$. We divide the interval $\bar{\Omega}=[\underline{x},\bar{x}]$ into $m+1$ subintervals (*elements*), each having length of $h=(\bar{x}-\underline{x})/(m+1)$. Let $x_i=\underline{x}+ih$, $i=0,1,\ldots,m+1$, be the nodes in $[\underline{x},\bar{x}]$. We define the following piecewise-linear finite-element basis functions $\{\phi_{h,i}(x)\}_{i=1}^m$:

$$\phi_{h,i}(x)=\begin{cases}(x-x_{i-1})/h, & x_{i-1}\leqslant x\leqslant x_i,\\ (x_{i+1}-x)/h, & x_i<x\leqslant x_{i+1},\\ 0, & x\notin[x_{i-1},x_{i+1}].\end{cases}$$

The ith basis function $\phi_{h,i}(x)$ is a hat function equal to one at the node x_i and zero at all other nodes. If we define the hat function $\phi(x):=(x+1)\mathbf{1}_{\{-1\leqslant x\leqslant 0\}}+(1-x)\mathbf{1}_{\{0<x\leqslant 1\}}$, then $\phi_{h,i}(x)=\phi((x-x_i)/h)$. More generally, we can define $\phi_{h,i}(x)$ for all integer $i\in\mathbb{Z}$. The nodes $x_0=\underline{x}$ and $x_{m+1}=\bar{x}$ are on the boundary, and the nodes x_i with $i<0$ or $i>m+1$ are outside of the interval $[\underline{x},\bar{x}]$ (we will need to consider nodes outside of $(\underline{x},\bar{x})$ when computing the load vector below due to the nonlocal character of the integral operator $\mathcal{B}$).

We look for a finite-element approximation u_h to the solution u of the variational formulation (3.4)–(3.5) as a linear combination of the finite-element basis functions with time-dependent coefficients

$$u_h(t,x)=\sum_{i=1}^{m}u_i(t)\phi_{h,i}(x),\quad t\in[0,T]. \tag{3.6}$$

Note that u_h vanishes outside of Ω by construction. Denote by $\mathbf{u}(t)=(u_1(t),\ldots,u_m(t))^\top$ the m-dimensional vector of time-dependent coefficients to be determined. Substituting (3.6) into (3.4)–(3.5) and letting the test function v in (3.4)–(3.5) run through the set of all basis functions $\{\phi_i\}_{i=1}^m$, we obtain the following m-dimensional system of ODEs:

$$\mathbb{M}\mathbf{u}'(t)+\mathbb{A}\mathbf{u}(t)-\mathbb{B}\mathbf{u}(t)+\mathbf{F}=0,\quad t\in(0,T], \tag{3.7}$$

with the initial condition

$$\mathbb{M}\mathbf{u}(0)=\mathbf{C}. \tag{3.8}$$

Here, $\mathbf{u}'(t)=(u_1'(t),\ldots,u_m'(t))^\top$, $u_i'(t)\equiv du_i(t)/dt$, $\mathbb{M}=(m_{ij})_{i,j=1}^m$, $m_{ij}=(\phi_j,\phi_i)$, $\mathbb{A}=(a_{ij})_{i,j=1}^m$, $a_{ij}=a(\phi_j,\phi_i)$, $\mathbb{B}=(b_{ij})_{i,j=1}^m$, $b_{ij}=b(\phi_j,\phi_i)$, $\mathbf{C}=(c_1,\ldots,c_m)^\top$, $c_i=(\psi-R,\phi_i)$, and $\mathbf{F}=(F_1,\ldots,F_m)^\top$, $F_i=a(R,\phi_i)-b(R,\phi_i)$ (to lighten notation, we omit the index h in $\phi_{h,i}$).

This ODE system is referred to as a *semidiscretization* of the variational problem. The problem is reduced to the integration of this ODE system. This is referred to as the *finite-element method-of-lines* (MOL) ("lines" is a metaphor for the lines (x_i,t), $t\geqslant 0$ in the (x,t)-domain, x_i fixed, along which the approximations to the PDE solution are studied; see Hundsdorfer and Verwer 2003). Due to the origins of the finite-element method in structural engineering, $\mathbb{M}$ is referred to as the *mass matrix*, $\mathbb{A}$ as the *stiffness matrix*, and $\mathbf{F}$ as the *load vector*. We call $\mathbb{B}$ the *jump matrix*. For each t, on a bounded domain Ω, the semidiscrete finite-element approximation is second-order accurate in the spatial step size h:

$$\|u_h(t,\cdot)-u(t,\cdot)\|\leqslant Ch^2,$$

both in the $L^2(\Omega)$ norm and in the $L^\infty(\Omega)$ norm (more precisely, in the maximum norm, the error estimate is $Ch^{2-\epsilon}$ for any $\epsilon>0$; maximum norm error estimates in the finite-element method are particularly relevant in financial engineering as the *worst-case pricing error estimates*).

The mass matrix $\mathbb{M}$ and the stiffness matrix $\mathbb{A}$ can be computed in closed form in this model (for more general processes, they can be computed by Gaussian quadrature as discussed below). For any $i,j\in\mathbb{Z}$, from the definition of the bilinear form $a(\cdot,\cdot)$, we have

$$a(\phi_i,\phi_i)=a_0=\frac{2}{3}(r+\lambda)h+\frac{1}{h}\sigma^2,$$

$$a(\phi_i,\phi_{i\pm1})=a_{\mp1}=\pm\frac{1}{2}\mu+\frac{1}{6}(r+\lambda)h-\frac{1}{2h}\sigma^2,$$

and $a(\phi_i,\phi_j)=0$ for $|i-j|>1$. Moreover,

$$(\phi_i,\phi_i)=\tfrac{2}{3}h,\qquad(\phi_i,\phi_{i\pm1})=\tfrac{1}{6}h,$$
$$(\phi_i,\phi_j)=0,\quad |i-j|>1.$$

Therefore, both $\mathbb{A}$ and $\mathbb{M}$ are tridiagonal $m \times m$ matrices with constant diagonals:

$$\mathbb{A}=\begin{pmatrix} a_0 & a_1 & & 0 \\ a_{-1} & a_0 & \ddots & \\ & \ddots & \ddots & a_1 \\ 0 & & a_{-1} & a_0 \end{pmatrix}, \quad \mathbb{M}=\frac{h}{6}\begin{pmatrix} 4 & 1 & & 0 \\ 1 & 4 & \ddots & \\ & \ddots & \ddots & 1 \\ 0 & & 1 & 4 \end{pmatrix}.$$

The elements of the jump matrix

$$b_{ij}=b(\phi_j,\phi_i)=(\mathscr{B}\phi_j,\phi_i)$$

have the form

$$\begin{aligned} b_{ij} &= \lambda\int_{\underline{x}}^{\bar{x}}\int_{-\infty}^{\infty}\phi_j(x+z)\phi_i(x)p(z)\,dz\,dx \\ &= \lambda\int_{x_{i-1}}^{x_{i+1}}\int_{x_{j-1}}^{x_{j+1}}\phi_j(y)\phi_i(x)p(y-x)\,dy\,dx \\ &= \lambda h^2\int_{-1}^{1}\int_{-1}^{1}\phi(u)\phi(w)p((w-u+j-i)h)\,dw\,du \qquad (3.9) \end{aligned}$$

and depend only on the difference $j-i$. Therefore, $\mathbb{B}$ is a *Toeplitz matrix* of the form

$$\mathbb{B}=\begin{pmatrix} b_{11} & b_{12} & \cdots & b_{1m} \\ b_{21} & b_{11} & \cdots & b_{1,m-1} \\ \vdots & \vdots & & \vdots \\ b_{m1} & b_{m-1,1} & \cdots & b_{11} \end{pmatrix}.$$

We only need to compute $2m-1$ values. The double integral (3.9) can be computed directly by two-dimensional Gaussian quadrature. Alternatively, in our implementation, we compute $b_{ij}=(\mathscr{B}\phi_j,\phi_i)$ as follows. We approximate the function $(\mathscr{B}\phi_j)(x)$ by its *finite-element interpolant* $I_h\mathscr{B}\phi_j(x)$:

$$\mathscr{B}\phi_j(x)\approx I_h\mathscr{B}\phi_j(x)=\sum_{l\in\mathbb{Z}}(\mathscr{B}\phi_j)(x_l)\phi_l(x),$$

where I_h is the finite-element interpolation operator. The finite-element interpolant $I_hf(x)=\sum_l f(x_l)\phi_l(x)$ of a function $f(x)$ is equal to the value of the function at the nodes x_l and interpolates between the nodes with the piecewise-linear finite-element basis functions $\phi_l(x)$. The error of the finite-element interpolation is $O(h^2)$ and is, thus, of the same order as the spatial discretization error in our semidiscrete formulation of the PIDE. Hence, we do not decrease the order of our numerical approximation by doing the interpolation. We then have the following approximation:

$$\begin{aligned} &(\mathscr{B}\phi_j,\phi_i) \\ &\quad\approx\sum_{l=i-1}^{i+1}\mathscr{B}\phi_j(x_l)\cdot(\phi_l,\phi_i) \\ &\quad=\tfrac{1}{6}h\,\mathscr{B}\phi_j(x_{i-1})+\tfrac{2}{3}h\mathscr{B}\phi_j(x_i)+\tfrac{1}{6}h\mathscr{B}\phi_j(x_{i+1}) \qquad (3.10) \end{aligned}$$

for any $i,j\in\mathbb{Z}$, where

$$\begin{aligned} \mathscr{B}\phi_j(x_l) &= \lambda\int_{\mathbb{R}}\phi_j(x_l+z)p(z)\,dz \\ &= \lambda h\int_{-1}^{1}\phi(x)p((x+j-l)h)\,dx \\ &= \lambda h\int_0^1[xp((x-1+j-l)h) \\ &\qquad +(1-x)p((x+j-l)h)]\,dx. \qquad (3.11) \end{aligned}$$

Note that $\mathscr{B}\phi_j(x_l)$ depends only on the difference $j-l$. Hence, to compute the jump matrix $\mathbb{B}$, we need to compute $2m+1$ values $\mathscr{B}\phi_j(x_l)$. In Kou's (2002) and Merton's (1976) models, the integrals in (3.11) can be calculated analytically. In Kou's (2002) model, $\mathscr{B}\phi_j(x_l)$ are given by

$$\mathscr{B}\phi_j(x_l)=\frac{p\lambda}{\eta_1 h}(1-e^{-\eta_1 h})^2e^{-(j-l-1)\eta_1 h}, \quad j-l\geqslant 1,$$

$$\mathscr{B}\phi_j(x_l)=\lambda-\frac{p\lambda}{\eta_1 h}(1-e^{-\eta_1 h})-\frac{(1-p)\lambda}{\eta_2 h}(1-e^{-\eta_2 h}), \quad j-l=0,$$

$$\mathscr{B}\phi_j(x_l)=\frac{(1-p)\lambda}{\eta_2 h}(1-e^{-\eta_2 h})^2e^{(j-l+1)\eta_2 h}, \quad j-l\leqslant -1.$$

In Merton's (1976) model,

$$\begin{aligned} &\mathscr{B}\phi_j(x_l) \\ &\quad=\frac{\lambda}{2}\bigg[\frac{\sqrt{2}s}{\sqrt{\pi}h}\Big(e^{-((j-l-1)h-m)^2/2s^2}-2e^{-((j-l)h-m)^2/2s^2} \\ &\qquad\qquad +e^{-((j-l+1)h-m)^2/2s^2}\Big) \\ &\qquad+\Big(j-l-1-\frac{m}{h}\Big)E\Big(\frac{h}{\sqrt{2}s}\Big(j-l-1-\frac{m}{h}\Big)\Big) \\ &\qquad-2\Big(j-l-\frac{m}{h}\Big)E\Big(\frac{h}{\sqrt{2}s}\Big(j-l-\frac{m}{h}\Big)\Big) \\ &\qquad+\Big(j-l+1-\frac{m}{h}\Big)E\Big(\frac{h}{\sqrt{2}s}\Big(j-l+1-\frac{m}{h}\Big)\Big)\bigg], \end{aligned}$$

where $E(x)=(2/\sqrt{\pi})\int_0^x e^{-t^2}\,dt$ is the error function.

For a general jump-magnitude distribution, the integral (3.11) is computed by numerical quadrature. The one-point Gaussian quadrature rule that evaluates the integrand at the center of the integration interval of length h has errors of the order $O(h^2)$ and is, thus, sufficient for the finite-element approximation (see, e.g., Ciarlet 1978, §4.1):

$$\begin{aligned} \mathscr{B}\phi_j(x_l) &\approx \tfrac{1}{2}h\lambda[p((j-l-1/2)h) \\ &\qquad +p((j-l+1/2)h)]. \qquad (3.12) \end{aligned}$$

The elements of the load vector $\mathbf{F}$, $F_i=a(R,\phi_i)-b(R,\phi_i)$, $i=1,\ldots,m$, are computed as follows. The first term $a(R,\phi_i)$ can be computed analytically in this model for $R=\psi$, where ψ is a call or put payoff. Generally, it can be computed by Gaussian quadrature. The second term $b(R, \phi_i) = (\mathcal{B}R, \phi_i)$ can be computed by two-dimensional Gaussian quadrature. Alternatively, in our implementation, we approximate the rebate by its finite-element interpolant $R(x) \approx I_h R(x) = \sum_{j\in\mathbb{Z}} R_j \phi_j(x)$, where $R_j = R(x_j)$, $j \in \mathbb{Z}$, are nodal values. Then, $b(R, \phi_i)$ is approximated by

$$b(R, \phi_i) \approx \sum_{j=-\infty}^{\infty} b(\phi_j, \phi_i) R_j \approx \sum_{j=i-m+1}^{i+m-1} b(\phi_j, \phi_i) R_j. \quad (3.13)$$

Here, we have truncated the infinite summation because by (3.11)–(3.12), $b_{ij} = b(\phi_j, \phi_i)$ are negligibly small for $|i - j| \geqslant m$ due to decay of the jump-size probability density $p(x)$.

Finally, the initial condition $\mathbb{M}\mathbf{u}(0) = \mathbf{C}$ with the vector $\mathbf{C}$, $c_i = (\psi - R, \phi_i)$, is treated as follows. For $R = \psi$, $\mathbf{C}$ vanishes identically, and we have a vanishing initial condition $\mathbf{u}(0) = 0$. For $R = 0$, $c_i = (\psi, \phi_i)$ can be computed analytically for simple payoffs or by Gaussian quadrature in general. Then, $\mathbf{u}(0)$ is obtained by solving $\mathbb{M}\mathbf{u}(0) = \mathbf{C}$.

REMARK 3.1. The jump matrix $\mathbb{B}$ is Toeplitz. In the numerical solution of the ODE system (3.7), we need to perform the jump matrix-vector multiplication at each time step. The Toeplitz matrix-vector multiplication can be accomplished efficiently in $O(m \log_2(m))$ floating-point operations using FFT (see the online appendix).

3.4. Finite-Element Discretization: The SVCJ Model

Now consider the two-dimensional SVCJ model on a bounded computational domain $\Omega = (\underline{x}, \bar{x}) \times (\underline{y}, \bar{y})$. We divide $[\underline{x}, \bar{x}]$ into $m + 1$ equal intervals of length $h_x = (\underline{x} - \bar{x})/(m + 1)$ and $[\underline{y}, \bar{y}]$ into $n + 1$ equal intervals of length $h_y = (\underline{y} - \bar{y})/(n + 1)$. The nodes are $(x_i, y_j) = (\underline{x} + ih_x, \underline{y} + jh_y)$, $i = 0, 1, \ldots, m+1$, $j = 0, 1, \ldots, n+1$. The rectangular two-dimensional finite-element basis functions are defined for any $i = 1, \ldots, m$ and $j = 1, \ldots, n$ as the product of the one-dimensional basis functions

$$\phi_{ij}(x, y) = \phi_{h_x, i}(x)\phi_{h_y, j}(y) = \phi((x - x_i)/h_x)\phi((y - y_j)/h_y),$$

where $\phi_{h,i}(\cdot)$ and $\phi(\cdot)$ are as defined in the previous subsection. The two-dimensional pyramid function ϕ_{ij} is equal to one at the node (x_i, y_j) and zero at all other nodes. There are $m \times n$ nodes in $[\underline{x}, \bar{x}] \times [\underline{y}, \bar{y}]$. We arrange the nodes as follows: $(x_1, y_1), (x_1, y_2), \ldots, (x_1, y_n), (x_2, y_1), (x_2, y_2), \ldots, (x_m, y_n)$. More generally, for arbitrary $i, j \in \mathbb{Z}$, we define the nodes (x_i, y_j) and basis functions ϕ_{ij} as above.

The elements of the mass matrix can be computed in closed form:

$$(\phi_{kl}, \phi_{ij}) = \begin{cases} 4h_x h_y/9, & i = k,\ j = l, \\ h_x h_y/9, & i = k \pm 1,\ j = l \text{ or } i = k,\ j = l \pm 1, \\ h_x h_y/36, & i = k \pm 1,\ j = l \pm 1, \\ 0, & |i - k| > 1 \text{ or } |j - l| > 1. \end{cases}$$

Hence, $\mathbb{M}$ is block tridiagonal with tridiagonal blocks:

$$\mathbb{M} = \begin{pmatrix} M_{11} & M_{12} & & 0 \\ M_{21} & M_{11} & \ddots & \\ & \ddots & \ddots & M_{12} \\ 0 & & M_{21} & M_{11} \end{pmatrix},$$

$$M_{11} = \frac{h_x h_y}{9} \begin{pmatrix} 4 & 1 & & 0 \\ 1 & 4 & \ddots & \\ & \ddots & \ddots & 1 \\ 0 & & 1 & 4 \end{pmatrix}, \qquad M_{12} = M_{21} = \tfrac{1}{4} M_{11},$$

with a total of nine nonzero diagonals. To compute elements of the stiffness matrix $\mathbb{A}$, we need to compute $a(\phi_{kl}, \phi_{ij})$ for $i, k = 1, \ldots, m$ and $j, l = 1, \ldots, n$, with the bilinear form defined in §3.2. The integrands in $a(\phi_{kl}, \phi_{ij})$ are polynomials with the highest-order terms $x^2 y^2$ and xy^3. For such integrands, the 2×2 Gaussian quadrature rule (the tensor product of two-point Gaussian quadrature rules for each coordinate) is exact. For fixed j and l, $a(\phi_{kl}, \phi_{ij})$ depend only on the difference $i - k$. Moreover, $a(\phi_{kl}, \phi_{ij}) = 0$ for $|i - k| > 1$ or $|j - l| > 1$. Therefore, $\mathbb{A}$ is also a block tridiagonal matrix with tridiagonal blocks:

$$\mathbb{A} = \begin{pmatrix} A_{11} & A_{12} & & 0 \\ A_{21} & A_{11} & \ddots & \\ & \ddots & \ddots & A_{12} \\ 0 & & A_{21} & A_{11} \end{pmatrix},$$

with a total of nine nonzero diagonals. It suffices to compute the blocks A_{11}, A_{12}, A_{21} with the total of $3(3n - 2)$ nonzero values.

The elements of the jump matrix $\mathbb{B}$ are defined by

$$\begin{aligned} &b(\phi_{kl}, \phi_{ij}) \\ &= \lambda \int_{x_{i-1}}^{x_{i+1}} \int_{y_{j-1}}^{y_{j+1}} \int_{-\infty}^{\infty} \int_{0}^{\infty} \phi_{ij}(x, y)\phi_{kl}(x + z^x, y + z^y) \\ &\qquad \cdot p(z^x, z^y)\, dz^y\, dz^x\, dy\, dx \\ &= \lambda \int_{x_{i-1}}^{x_{i+1}} \int_{y_{j-1}}^{y_{j+1}} \int_{x_{k-1}}^{x_{k+1}} \int_{y_{l-1}}^{y_{l+1}} \phi_{ij}(x, y)\phi_{kl}(u, w) \\ &\qquad \cdot p(u - x, w - y)\, dw\, du\, dy\, dx \\ &= \lambda h_x^2 h_y^2 \int_{-1}^{1} \int_{-1}^{1} \int_{-1}^{1} \int_{-1}^{1} \phi(x)\phi(y)\phi(u)\phi(w) \\ &\qquad \cdot p((u - x + k - i)h_x, \\ &\qquad (w - y + l - j)h_y)\, dw\, du\, dy\, dx. \quad (3.14) \end{aligned}$$

The $b(\phi_{kl}, \phi_{ij})$ depend only on the differences $k - i$ and $l - j$. Hence, $\mathbb{B}$ is a block Toeplitz matrix with Toeplitz blocks (BTTB) with m^2 blocks, and each block is an $n \times n$ Toeplitz matrix:

$$\mathbb{B} = \begin{pmatrix} B_{11} & B_{12} & \cdots & B_{1m} \\ B_{21} & B_{11} & \cdots & \vdots \\ \vdots & \vdots & & B_{12} \\ B_{m1} & \cdots & B_{21} & B_{11} \end{pmatrix}.$$

For the SVCJ model, because there is no negative jump in variance, $b(\phi_{kl}, \phi_{ij}) = 0$, $l - j < -1$, and each block in $\mathbb{B}$ is nearly upper diagonal, and there are a total of $(2m-1)(n+1)$ values to compute. As in the one-dimensional case, the $b(\phi_{kl}, \phi_{ij})$ can be approximated by the finite-element interpolation

$$b(\phi_{kl}, \phi_{ij}) \approx \sum_{r=i-1}^{i+1} \sum_{s=j-1}^{j+1} (\mathscr{B}\phi_{kl})(x_r, y_s)(\phi_{rs}, \phi_{ij}),$$

where

$$\begin{aligned}
&(\mathscr{B}\phi_{kl})(x_r, y_s) \\
&= \lambda \int_{-\infty}^{\infty} \int_0^{\infty} \phi_{kl}(x_r + z^x, y_s + z^y) p(z^x, z^y)\, dz^y\, dz^x \\
&= \lambda h_x h_y \int_{-1}^{1} \int_{-1}^{1} \phi(u)\phi(w) p\big((u+k-r)h_x, (w+l-s)h_y\big)\, dw\, du \\
&= \lambda h_x h_y \int_0^1 \int_0^1 \big[uwp((u-1+k-r)h_x, (w-1+l-s)h_y) \\
&\qquad + (1-u)wp((u+k-r)h_x, (w-1+l-s)h_y) \\
&\qquad + (1-u)(1-w)p((u+k-r)h_x, (w+l-s)h_y) \\
&\qquad + u(1-w)p((u-1+k-r)h_x, \\
&\qquad\qquad (w+l-s)h_y)\big]\, dw\, du.
\end{aligned}$$

We use the Gaussian quadrature to compute this integral (1×1 Gaussian quadrature rule with the integrand evaluated at the center of the rectangle $[0, 1] \times [0, 1]$ (the tensor product of one-point Gaussian quadrature rules for each coordinate) has errors of the order $O(h^2)$, the same order as the error of the finite-element spatial discretization; higher-order Gaussian quadrature rules such as 2×2 (the tensor product of two-point Gaussian quadrature rules) can be used to improve the constant in the Ch^2 error estimate without affecting the order).

To compute the load vector, the quantities $a(R, \phi_{ij})$ can be computed by Gaussian quadrature and $b(R, \phi_{ij})$ can be computed by Gaussian quadrature or approximated as in the one-dimensional case

$$b(R, \phi_{ij}) \approx \sum_{k=i-m+1}^{i+m-1} \sum_{l=j-n+1}^{j+n-1} (\mathscr{B}\phi_{kl}, \phi_{ij}) R_{kl},$$

where $R_{kl} = R(x_k, y_l) = R(x_k)$ (the option payoff is independent of the volatility variable y). Finally, as in the one-dimensional case, the initial condition is either $\mathbf{u}(0) = 0$ when $R = \psi$, or the solution of $\mathbb{M}\mathbf{u}(0) = \mathbf{C}$ when $R = 0$. Here, $\mathbf{C} = (c_{ij})$, $c_{ij} = (\psi, \phi_{ij}) = h_y c_i$, where $c_i = (\psi, \phi_i)$ is computed as in the one-dimensional case.

4. Integrating the ODE System: An Extrapolation Approach

We have reduced the option-pricing problem to the solution of the ODE system (3.7). In this section, we present a time-stepping scheme to solve this ODE system numerically. The scheme is based on applying an extrapolation procedure to the IMEX Euler time-stepping scheme. We first described the IMEX Euler scheme and then the extrapolation procedure.

For the one-dimensional jump-diffusion model, we observe that $\mathbb{M} \sim O(h)$, $\mathbb{A} \sim O(h^{-1})$, and $\mathbb{B} \sim O(h^2)$ (for the latter, see Equations (3.10) and (3.12)). Hence, the system (3.7) is stiff. In particular, the term $\mathbb{A}\mathbf{u}$ resulting from the discretization of the diffusion part of the PIDE generates stiffness, while the term $\mathbb{B}\mathbf{u}$ resulting from the discretization of the integral operator does not generally generate stiffness. For stiff systems, fully explicit schemes are only conditionally stable and may require prohibitively small time steps when h is small. Therefore, we treat the term $\mathbb{A}\mathbf{u}$ implicitly for stability reasons. Recall that $\mathbb{B}$ is a dense matrix (as opposed to the tridiagonal $\mathbb{A}$). Because it does not generate stiffness, we treat the term $\mathbb{B}\mathbf{u}$ explicitly to avoid inverting the dense matrix. This is an example of an IMEX time-stepping scheme (see Hundsdorfer and Verwer 2003, §IV.4 for a survey of IMEX methods for ODEs and PDEs).

Divide the time interval $[0, T]$ into N time steps, each having length $k = T/N$, and with the nodes $t_i = ik$, $i = 0, 1, \ldots, N$. Define $\mathbf{u}^i := \mathbf{u}(t_i)$, $i = 0, 1, \ldots, N$. Then, the IMEX Euler time-stepping scheme starts with the initial condition $\mathbb{M}\mathbf{u}^0 = \mathbf{C}$ and marches forward according to

$$(\mathbb{M} + k\mathbb{A})\mathbf{u}^i = (\mathbb{M} + k\mathbb{B})\mathbf{u}^{i-1} - k\mathbf{F}, \quad i = 1, \ldots, N. \tag{4.1}$$

At each step, the linear system (4.1) is solved to determine the m-dimensional vector $\mathbf{u}^i$. This scheme is unconditionally stable,[1] first-order accurate in time, and its error is known to have an asymptotic expansion in the powers of the time step k:

$$\mathbf{u}(T) - \mathbf{u}^N = \mathbf{e}_1(T)k + \mathbf{e}_2(T)k^2 + \cdots. \tag{4.2}$$

Generally, the Euler scheme for the ODE $u' = G(u)$ has an asymptotic error expansion (4.2) if the right-hand side G is smooth. In our case, G is linear and, hence, (4.2) holds.

The asymptotic error expansion (4.2) suggests applying extrapolation to cancel lower-order terms in the error expansion and to increase the order of the scheme. References on extrapolation methods for stiff ODE systems include Deuflhard (1985), Deuflhard and Bornemann (2002, §6.4.2), and Hairer and Wanner (1996, §IV.9).

We now describe the extrapolation scheme based on the IMEX Euler scheme. We need to integrate the ODE system on the interval $[0, T]$. Assume that a *basic step size* H ($H = T/N$) and an *extrapolation stage number* $s \geqslant 1$ are given. Then, one constructs a sequence of approximations to $\mathbf{u}(H)$

(at time H) using the discretization method (4.1) with *internal step sizes* $k_i = H/n_i$, $i = 1, 2, \ldots, s+1$, where $\{n_i\}_{i=1}^{s+1}$ is the *step number sequence*. For the extrapolation scheme based on the IMEX Euler time stepping, we use the harmonic sequence $\{1, 2, 3, \ldots, s+1\}$. Denoting the approximation obtained at time H (after one basic step) with internal step size k_i by $\mathbf{T}_{i,1} = \mathbf{u}(H; k_i)$, the *extrapolation tableau* is constructed as follows:

$$\mathbf{T}_{i,j} = \mathbf{T}_{i,j-1} + \frac{\mathbf{T}_{i,j-1} - \mathbf{T}_{i-1,j-1}}{(n_i/n_{i-j+1}) - 1}, \qquad i = 2, \ldots, s+1,\ j = 2, \ldots, i. \quad (4.3)$$

The extrapolation tableau can be graphically depicted as follows:

$$\begin{array}{llll} \mathbf{T}_{1,1} & & & \\ \mathbf{T}_{2,1} & \mathbf{T}_{2,2} & & \\ \vdots & \vdots & \ddots & \\ \mathbf{T}_{s+1,1} & \mathbf{T}_{s+1,2} & \cdots & \mathbf{T}_{s+1,s+1}. \end{array}$$

The value $\mathbf{T}_{s+1,s+1}$ after s extrapolation stages is accepted as the approximation to $\mathbf{u}(H)$ and is used as the starting point to launch a new basic integration step over the next interval $[H, 2H]$. The procedure is continued in this way for N basic steps until the approximation of $\mathbf{u}(T)$ at time $T = NH$ is obtained. We now investigate the properties of the extrapolation scheme.

Proposition 4.1. *We have the following error estimate*:

$$\mathbf{u}(H) - \mathbf{T}_{s+1,s+1} = O(k_1 k_2 \cdots k_{s+1}) = O(H^{s+1}/(n_1 n_2 \cdots n_{s+1})). \quad (4.4)$$

For the step number sequence $\{1, 2, 3, \ldots, s+1\}$, *we have, in particular,*

$$\mathbf{u}(H) - \mathbf{T}_{s+1,s+1} = O(H^{s+1}/(s+1)!). \quad (4.5)$$

The total number of time steps required to compute $\mathbf{T}_{s+1,s+1}$ (*the total number of times the linear system* (4.1) *needs to be solved*) *is*

$$\mathcal{N}_s = (s+1)(s+2)/2. \quad (4.6)$$

Proof. For each $\mathbf{T}_{i,1}$, we have the asymptotic error expansion of the form (4.2):

$$\mathbf{u}(H) - \mathbf{T}_{i,1} = \mathbf{e}_1(H) k_i + \mathbf{e}_2(H) k_i^2 + \cdots,$$

where $k_i = H/n_i$. From the asymptotic error expansions for $\mathbf{T}_{1,1}$ and $\mathbf{T}_{2,1}$, we can eliminate the terms $\mathbf{e}_1(H)k_1$ and $\mathbf{e}_1(H)k_2$ by forming a new approximation for $\mathbf{u}(H)$:

$$\mathbf{T}_{2,2} = \frac{k_1 \mathbf{T}_{2,1} - k_2 \mathbf{T}_{1,1}}{k_1 - k_2} = \mathbf{T}_{1,1} + \frac{\mathbf{T}_{2,1} - \mathbf{T}_{1,1}}{(n_2/n_1) - 1},$$

which has the asymptotic error expansion $\mathbf{u}(H) - \mathbf{T}_{2,2} = -\mathbf{e}_2(H) k_1 k_2 + \cdots$. The error is of the order $O(k_1 k_2)$. Continuing recursively down the extrapolation tableau, we determine that the error estimate for each $\mathbf{T}_{i,j}$ is $\mathbf{u}(H) - \mathbf{T}_{i,j} = O(k_i k_{i-1} \cdots k_{i-j+1})$, and, in particular, for $i = j = s+1$, we obtain (4.4). For the harmonic step number sequence $\{1, 2, 3, \ldots, s+1\}$, we obtain (4.5). The total number of time steps required to compute $\mathbf{T}_{s+1,s+1}$ is $1 + 2 + 3 + \cdots + (s+1) = (s+1)(s+2)/2$. $\square$

Thus, the error after s extrapolation stages is of the order $O(H^{s+1}/(s+1)!)$. Recall that the factorial can be well approximated by $n! \approx \sqrt{2\pi(1 + 1/(6n))}\, n^{n+1/2} e^{-n}$ (a refinement of the Stirling formula). This approximation is very accurate even for small n (e.g., for $n = 2$, this, gives 1.9974). For fixed s, the error of the extrapolation scheme is thus

$$\mathbf{u}(H) - \mathbf{T}_{s+1,s+1} = O\big(\{2\pi(1 + 1/(6(s+1)))\}^{-1/2} \cdot \exp\{-(s+3/2)\ln(s+1) + (s+1)(1 + \ln H)\}\big).$$

To get some intuition on the dependence of the error on the number of time steps, recalling that the total number of time steps needed to integrate the ODE on the interval $[0, H]$ is $\mathcal{N}_s = (s+1)(s+2)/2$, we write the error estimate as follows:

$$\mathbf{u}(H) - \mathbf{T}_{s+1,s+1} = O\big(e^{-c\sqrt{\mathcal{N}_s}\ln \mathcal{N}_s}\big). \quad (4.7)$$

This suggests that the error decreases as $e^{-c\sqrt{\mathcal{N}_s}\ln \mathcal{N}_s}$ with the increasing number of time steps $\mathcal{N}_s$. We stress that the argument above is not rigorous. Proposition 4.1 only states that for fixed s and $H \to 0$, the error is asymptotically $O(H^{s+1}/(s+1)!)$. Generally, it does not say anything about the behavior of the error with increasing s because the constant C in the estimate $CH^{s+1}/(s+1)!$ may depend on s and, hence, on $\mathcal{N}_s$. If a hypothesis that the constant can be made independent of s (or increases slowly with s) holds, then Equation (4.7) would, in fact, provide an error estimate in terms of the number of time steps. Unfortunately, it appears difficult to prove this hypothesis. However, in our numerical experiments with option pricing applications, we do observe the rate of convergence suggested by the heuristic (4.7).

For a fixed basic step H and a fixed number of extrapolation stages s, the total number of time steps needed to integrate the ODE on the time interval $[0, T]$ is $\mathcal{N}_{T,H,s} = N(s+1)(s+2)/2$, where $N = T/H$. Due to rapid convergence of the extrapolation scheme, N and s are typically small in option pricing applications, resulting in the small total number of time steps $\mathcal{N}$ required to achieve desired accuracy.

At each time step, we need to solve the linear system (4.1). In one-dimensional models, we have a tridiagonal system. This can be solved using the LU decomposition with forward/backward substitution in $O(m)$ operations for a system of order m. In the two-dimensional SVCJ model, we need to solve a block tridiagonal system. Direct solvers such as LU decomposition become computationally expensive, and iterative solvers are typically preferred. In our numerical experiments, we use the successive over-relaxation (SOR) algorithm. Because the coefficient matrix of the system for the SVCJ model is block tridiagonal with tridiagonal blocks (nine nonzero diagonals), SOR takes $9mn$ multiplications for each iterate, as compared to the m^2n^2 multiplications for a full coefficient matrix. At each time step, we also need to compute the jump matrix-vector multiplication $\mathbb{B}\mathbf{u}^{i-1}$. In one-dimensional models, the jump matrix is Toeplitz. The Toeplitz matrix-vector multiplication is accomplished in $O(m\log_2(m))$ floating-point operations using FFT, as compared to the m^2 operations for direct matrix-vector multiplication (see the online appendix). Hence, our time-stepping scheme takes $O(\mathcal{N}m\log_2(m))$ floating-point operations, where $\mathcal{N}$ is the total number of time steps. In two-dimensional applications such as the SVCJ model, the jump matrix is a block Toeplitz matrix with Toeplitz blocks (BTTB). The BTTB matrix-vector multiplication can be done in $O(mn\log_2(mn))$ floating-point operations (see the online appendix), as compared to the m^2n^2 operations for direct matrix-vector multiplication. Hence, the total operation count is $O(\mathcal{N}mn\log_2(mn))$.

So far, we have taken the basic step size H and the number of extrapolation stages s as given. We now discuss the selection of H and s in practice. First, suppose that H is fixed, and select a local error tolerance $\epsilon > 0$. After j extrapolation stages, $\mathbf{T}_{j+1,j+1}$ approximates $\mathbf{u}(H)$. The error estimate is $\mathscr{E}_j := \|\mathbf{T}_{j+1,j+1} - \mathbf{T}_{j+1,j}\|_{L^\infty}$ (the so-called *subdiagonal error estimate*; see Hairer and Wanner 1996, p. 140). After each extrapolation stage with $j \geqslant 2$, we compare the estimated error $\mathscr{E}_j$ with the preceding error $\mathscr{E}_{j-1}$ and with the desired local error tolerance ϵ. Whenever $\mathscr{E}_j \leqslant \epsilon$ for some $j \leqslant s_{\max}$, we accept $\mathbf{T}_{s+1,s+1}$ as the approximation to $\mathbf{u}(H)$ and move on to compute the solution over the next basic step $[H, 2H]$ starting with $\mathbf{u}(H)$. Alternatively, whenever $\mathscr{E}_j \geqslant \mathscr{E}_{j-1}$ (i.e., increasing extrapolation depth does not result in further error reduction) or if the desired error tolerance is not achieved in $s_{\max}$ stages, we restart the computation of the step with a smaller H, say $H_{\text{new}} = H_{\text{old}}/2$. In our numerical experiments, we typically set $s_{\max} = 10$, so that if the desired error tolerance is not achieved in 10 extrapolation stages, we reduce the basic step size. This simple procedure allows us to select the basic step size H and the extrapolation stage s adaptively. The only user-specified parameter in addition to the desired local error tolerance is the initial basic step size H. If the initial H is too large relative to the error tolerance, the adaptive procedure will reduce it and restart the computation with smaller H. If H is selected too small relative to the desired error tolerance, more time steps than necessary will be computed. In our computational experiments with options pricing problems in §6, $H = 0.5$ year has proven adequate as a starting basic step size for local error tolerances up to 10^{-5} (without the need to reduce the basic step size in most cases). For problems with maturities less than six months, we set $H = T$. For faster computations with less precision (e.g., error tolerances up to 10^{-3}), the basic step $H = 1$ can be used as a starting step. For more precise benchmark computations, $H = 0.25$ can be used.

5. Further Applications and Extensions of the Extrapolation Method

5.1. Pricing Bermudan Options

Our method also directly applies to Bermudan options. A Bermudan option can be exercised on any date in a given discrete set of dates. Consider a Bermudan option with maturity T. Suppose that the option holder has the right to exercise the option at any date in the set $\{0, \Delta, 2\Delta, \ldots, N_B\Delta = T\}$, where $\Delta > 0$ is a fixed-time interval between the possible exercise dates and $N_B = T/\Delta$. We denote the value function of the option at time $j\Delta$ by $V^j(x)$, $j = 0, 1, \ldots, N_B$. At the option maturity $T = N_B\Delta$, the option value is given by the payoff function $V^{N_B}(x) = \psi(x)$. In the time interval $[(N_B - 1)\Delta, N_B\Delta)$ proceeding expiration, the option value as a function of the state variable x and time remaining to maturity t solves the following PIDE (see §2.2):

$$U_t - \mathscr{A}U - \mathscr{B}U = 0, \quad t \in (0, \Delta],\ x \in \Omega,$$

with the initial condition $U(0, x) = V^{N_B}(x) = \psi(x)$, $x \in \Omega$, and the artificial knock-out condition $U(t, x) = \psi(x)$, $x \in \Omega^c$ (as in the case of European options, we approximate U in the knock-out region Ω^c by the payoff function ψ). The above PIDE need to be solved to obtain the *continuation value function* $U(\Delta, x)$, i.e., the value function at (calendar) time $(N_B - 1)\Delta$ before applying the early exercise constraint. Because the option holder can exercise the option at time $(N_B - 1)\Delta$, the option value V^{N_B-1} at time $(N_B - 1)\Delta$ is therefore given by

$$V^{N_B-1}(x) = \max\{\psi(x), U(\Delta, x)\}.$$

Then, starting from $V^{N_B-1}(x)$, we compute the option value at time $(N_B - 2)\Delta$ using the same procedure. By induction, to compute the option value $V^j(x)$ at time $j\Delta$, $0 \leqslant j \leqslant N_B - 1$, we solve the following PIDE problem:

$$U_t - \mathscr{A}U - \mathscr{B}U = 0, \quad t \in (0, \Delta],\ x \in \Omega, \tag{5.1}$$

$$U(0, x) = V^{j+1}(x), \quad x \in \Omega, \tag{5.2}$$

$$U(t, x) = \psi(x), \quad x \in \Omega^c,\ t \in (0, \Delta], \tag{5.3}$$

and then apply the early exercise condition

$$V^j(x) = \max\{U(\Delta, x), \psi(x)\}. \tag{5.4}$$

Therefore, we need to successively solve N_B initial value PIDE problems between the consecutive early exercise dates to obtain the value function of the Bermudan option at the contract inception. Each of these initial value problems can be discretized in space as described in §3. Let $u = U - \psi$. Then, the linear-finite element discretization of §3 leads to the following recursive system of ODEs:

$$\mathbb{M}\mathbf{u}'(t) + \mathbb{A}\mathbf{u}(t) - \mathbb{B}\mathbf{u}(t) + \mathbf{F} = 0, \quad t \in (0, \Delta],$$
$$\mathbb{M}\mathbf{u}(0) = C^{j+1},$$

where $\mathbb{A}$, $\mathbb{B}$, $\mathbf{F}$ are the same as in §3 with the payoff serving as the artificial rebate in the boundary condition, $R = \psi$, and the initial condition vector $C^{j+1} = (c_1^{j+1}, \ldots, c_m^{j+1})^\top$, $c_i^{j+1} = (V^{j+1} - \psi, \phi_i)$ for each of the ODEs. The function V^{j+1} is obtained at the previous step and is used in the initial condition for the current step. Approximating the function V^{j+1} and the payoff ψ by their finite-element interpolants, the initial conditions simplify to

$$\mathbf{u}(0) = (u_1(0), u_2(0), \ldots, u_m(0)),$$
$$u_i(0) = V_i^{j+1} - \psi_i, \quad i = 1, 2, \ldots, m,$$

where ψ_i and V_i^{j+1} are the nodal values. At each step of the recursion, the ODE is integrated in time using the extrapolation scheme described in §4. Numerical results will be presented in §6.

While the treatment of Bermudan options is straightforward, a more significant extension is required for the efficient pricing of American options with continuously allowed early exercise. One approach is to approximate American options with Bermudan options with a large but finite number of allowed early exercise dates. It is a viable approach, and our method can be used to price the approximating Bermudan option. However, the convergence of Bermudan option prices to the continuous American price as $\Delta \to 0$ is known to be slow. Thus, to get accurate approximations in this approach, one will have to price Bermudan options with very small Δ, which will require solving a large number of ODE systems. Alternatively, one may try applying extrapolation in Δ, by pricing a sequence of Bermudan options with decreasing Δ and then extrapolating to the limit $\Delta \to 0$.

A possibly more efficient (and significantly more involved) alternative is an adaptive implementation of the extrapolation scheme with adaptive time step and extrapolation order selection to integrate a nonlinear system of ODEs resulting from the semidiscretization of the penalized PIDE approximation of the variational inequality (see d'Halluin et al. 2004 for the penalty method for American options). It is well known in the literature that the American option problem generally requires adaptive time stepping for high-order schemes to realize their expected high-order convergence. This makes the American option problem more computationally challenging. Adaptive extrapolation schemes with automatic adaptive selection of the basic step size H and the extrapolation stage s are described in Deuflhard (1985) and Deuflhard and Bornemann (2002). We leave the study of these approaches to American options with continuous early exercise for future research.

5.2. Lévy Processes with Infinite Activity Jumps

So far, we have limited ourselves to jump-diffusion processes with nontrivial diffusion components and finite activity jumps. In this section, we discuss to what extent our approach can be extended to the pricing of options under Lévy processes with infinite activity jumps. One approach due to Cont and Voltchkova (2005) regularizes the PIDE by approximating small jumps with magnitudes less than some fixed $\epsilon > 0$ with a Brownian motion process with matched volatility parameter and, hence, reduces the problem to a jump-diffusion process with finite-activity jumps. This approach is also taken in Asmussen and Rosinski (2001) in the context of Lévy process simulation. Following this approach, one can extend our results in this paper to infinite activity jumps. However, this solves the problem only in part because the convergence estimates depend on the size of the artificial diffusion coefficient introduced to approximate small jumps.

We now discuss the possibility of directly applying our extrapolation scheme based on the IMEX Euler method to Lévy processes with infinite-activity jumps. Suppose that the Lévy process has a nontrivial diffusion component with volatility $\sigma > 0$ and an infinite-activity jump component. The infinitesimal generator of the Markovian jump process can be interpreted as a pseudodifferential operator of order $\nu \in [0, 2)$ (the diffusion operator, the Laplacian, is the limiting case with $\nu = 2$). For example, the popular variance gamma (VG) model has order $\nu = 0$, the Carr-Geman-Madan-Yor (CGMY) model has order $\nu = Y \in (0, 2)$, and the normal inverse Gaussian (NIG) model has order $\nu = 1$. The IMEX Euler scheme treats the jump part explicitly. As discussed in Matache et al. (2004), treating a pseudo-differential operator of order ν explicitly leads to a stability condition of the form $\Delta t \leqslant C(\Delta x)^\nu$. For the VG jumps, this gives a mild stability condition $\Delta t \leqslant C$ independent of the spatial discretization step size Δx. We thus expect that our extrapolation scheme based on the IMEX Euler will work well for the diffusion-extended VG model. As we will see in §6, this is indeed supported by numerical experiments. For other Lévy jumps with $\nu > 0$, the stability condition is more stringent because it depends on Δx. Still, for processes with $\nu < 1$, it may be a viable method because the stability condition is still quite mild. For $\nu \geqslant 1$, however, the stability condition becomes more restrictive. For such models, fully implicit schemes will be expected to be more efficient. Furthermore, in the case of pure jump Lévy processes with no diffusion component, the operator consists of a first-order drift operator (a hyperbolic operator) and a pseudodifferential jump operator of order ν. Such problems without the diffusion operator require special care in the spatial discretization as well to realize the expected convergence order. Our linear finite-element discretization in this paper requires nonvanishing diffusion to realize the expected $O(h^2)$ spatial convergence. More powerful, high-order finite-element and time-stepping schemes, such as wavelet Galerkin spatial discretization of the integrodifferential operator and *hp*-discontinuous Galerkin time stepping, can handle in a unified way all Lévy processes, including pure jump processes with order $\nu \in [0, 2)$. These powerful methods have been applied to option pricing under Lévy processes in a series of interesting recent papers by Matache et al. (2005b, 2006). While these methods are more general than the extrapolation method of this paper and are well adapted for pure jump infinite-activity processes, they are significantly more complicated to implement and integrate in existing financial software systems. In contrast, the extrapolation method is very straightforward to implement. We thus expect it to be a method of choice in practice for jump-diffusion processes with either finite-activity jumps or infinite-activity jumps with $\nu = 0$ (such as the VG).

In the rest of this section, we develop the application of our method to the diffusion-extended VG model. The asset price process is assumed to follow $S_t = Ke^{X_t}$, $t \geqslant 0$, where $X_t = \ln(S_0/K) + \mu t + \sigma B_t + W(\gamma_t; \theta, s)$, where B_t is a standard Brownian motion, and W is an independent Brownian motion with drift θ and diffusion coefficient s time changed with a random time process $\gamma_t = \gamma(t; 1, \nu)$, a gamma Lévy subordinator with mean rate one and variance rate ν (see Madan and Seneta 1990, Madan and Milne 1991, and Madan et al. 1998). To ensure that the total discounted gains process including capital gains and dividends is a martingale under the risk-neutral measure, the drift parameter is restricted to $\mu = r - q + \ln(1 - \theta\nu - s^2\nu/2)/\nu$. The infinitesimal generator $\mathscr{G}$ of X is given by

$$\mathscr{G}f(x) = \tfrac{1}{2}\sigma^2 f_{xx} + \mu f_x + \int_{\mathbb{R}} [f(x+z) - f(x)]\pi(z)\,dz,$$

where $\pi(\cdot)$ is the Lévy density of the variance gamma process

$$\pi(z) = \frac{1}{\nu z} e^{-\lambda_p z} 1_{\{z>0\}} + \frac{1}{\nu|z|} e^{-\lambda_n |z|} 1_{\{z<0\}},$$

where

$$\lambda_p = \sqrt{\frac{\theta^2}{s^4} + \frac{2}{s^2\nu}} - \frac{\theta}{s^2}, \qquad \lambda_n = \sqrt{\frac{\theta^2}{s^4} + \frac{2}{s^2\nu}} + \frac{\theta}{s^2}.$$

Note that $\int_R \pi(z)\,dz = \infty$. Therefore, the integral component of $\mathscr{G}$ cannot be separated into two parts as in the cases of finite-activity models. Correspondingly, the differential and integral operators are defined as follows:

$$\mathscr{A}U = \tfrac{1}{2}\sigma^2 U_{xx} + \mu U_x - rU,$$

$$\mathscr{B}U(t, x) = \int_{\mathbb{R}} [U(t, x+z) - U(t, x)]\pi(z)\,dz.$$

For the finite-element discretization, we follow the same procedure as in §3.3. In particular, we need to compute $\mathscr{B}\phi_j(x_l)$. When $j \neq l$, $\phi_j(x_l) = 0$ and

$$\mathscr{B}\phi_j(x_l) = h\int_{-1}^{1} \phi(x)\pi((j - l + x)h)\,dx.$$

When $j = l$, $\phi_j(x_l) = 1$ and $\phi_j(x_l + z) = 0$ for $|z| \geqslant h$. Then,

$$\mathscr{B}\phi_j(x_l) = h\int_{-1}^{1} (\phi(x) - 1)\pi(xh)\,dx - \int_{\{|z|\geqslant h\}} \pi(z)\,dz.$$

The above integrals can be computed analytically in terms of the exponential integral. Let E_1 be the exponential integral defined by $E_1(a) = \int_a^\infty (1/x)e^{-x}\,dx = \int_1^\infty (1/x)e^{-ax}\,dx$. Then, for $j - l \geqslant 1$,

$$\begin{aligned}&\mathscr{B}\phi_j(x_l)\\&= \frac{1}{\lambda_p h\nu} e^{-\lambda_p h(j-l-1)}(1 - e^{-\lambda_p h})^2\\&\quad - \frac{1}{\nu}\big[(j-l-1)E_1((j-l-1)\lambda_p h) - 2(j-l)E_1((j-l)\lambda_p h)\\&\qquad + (j-l+1)E_1((j-l+1)\lambda_p h)\big].\end{aligned}$$

For $j - l \leqslant -1$,

$$\begin{aligned}&\mathscr{B}\phi_j(x_l)\\&= \frac{1}{\lambda_n h\nu} e^{\lambda_n h(j-l+1)}(1 - e^{-\lambda_n h})^2\\&\quad - \frac{1}{\nu}\big[-(j-l+1)E_1(-(j-l+1)\lambda_n h) + 2(j-l)\\&\qquad \cdot E_1(-(j-l)\lambda_n h) - (j-l-1)E_1(-(j-l-1)\lambda_n h)\big].\end{aligned}$$

In the above, we use the convention that $jE_1(ja) = 0$ for $j = 0$ (note that $\lim_{x\downarrow 0} xE_1(x) = 0$). Finally, when $j = l$,

$$\begin{aligned}\mathscr{B}\phi_j(x_j) &= -\frac{1}{\nu}(E_1(\lambda_p h) + E_1(\lambda_n h))\\&\quad - \frac{1}{\lambda_n h\nu}(1 - e^{-\lambda_n h}) - \frac{1}{\lambda_p h\nu}(1 - e^{-\lambda_p h}).\end{aligned}$$

Therefore, the jump matrix $\mathbb{B}$ can be constructed using the above analytical formulas. The mass matrix, the stiffness matrix, and the load vector are as in §3.3. From the above expressions, we note that the diagonal elements of the jump matrix $\mathbb{B}$ are now of order $O(h)$, in contrast to the $O(h^2)$ jump matrix for jump diffusions with finite-activity jumps. This is similar to reaction terms in PDEs and leads to a mild stability condition of the form $\Delta t \leqslant C$ for some constant independent of the spatial discretization and dependent only on the process parameters for explicit discretizations of the jump term. The numerical integration of the resulting system of ODEs is then accomplished using the extrapolation of the IMEX Euler scheme as described in §4.

While in this section we discuss Lévy processes, our discussion generally applies to any jump-diffusion processes, not necessarily with stationary and independent increments, including processes with time- and state-dependent coefficients (diffusion, drift, interest rate, jump arrival intensity, and jump-size distribution). In the ODE formulation, the stiffness and jump matrices $\mathbb{A}$ and $\mathbb{B}$ and the load vector $\mathbf{F}$ depend on the choice of the process. If the coefficients are time dependent, they become time dependent, and in the discretization scheme, $\mathbb{A}^i$, $\mathbb{B}^i$, and $\mathbf{F}^i$ will change from time step to time step.

6. Numerical Experiments

In this section, we present our numerical results for European, Bermudan, and single- and double-barrier options in one- and two-dimensional jump-diffusion models. Parameters used in our numerical experiments are given in Table 1. We investigate pricing errors due to localization and spatial and time discretizations. Errors are in the maximum norm on the approximation domain $G = [\log(0.8), \log(1.2)]$ in the x-variable (corresponding to the approximation domain $G = [80, 120]$ in the underlying stock price) in one-dimensional models and $G = [\log(0.8), \log(1.2)] \times [0, 3]$ in the (x, y)-plane (corresponding to $[80, 120]$ in the underlying stock price and $[20\%, 40\%]$ in the volatility) in the SVCJ model.

For Bermudan and barrier options, benchmark prices are computed with large enough computational domains and small enough space steps and time steps. For European options in Kou's (2002) and Merton's (1976) models, accurate benchmark prices can be computed using available analytical solutions or the Fourier method (Carr and Madan 1999, Feng and Linetsky 2007). For European options in the SVCJ model, the benchmark prices can be computed using the Fourier method for affine jump diffusions due to Duffie et al. (2000).

Figure 1 illustrates convergence of the finite-element spatial discretization and effects of localization and numerical integration in Kou's (2002) and Merton's (1976) models. The four plots on the left show pricing errors for one-year double-barrier put (DBP), down-and-out put (DOP), and up-and-out put (UOP) options in Kou's (2002) model. The four plots on the right show pricing errors of one-year double-barrier call (DBC), down-and-out call (DOC), and up-and-out call (UOC) options in Merton's (1976) model. The two plots in the first row plot the maximum norm pricing error as a function of the spatial discretization step size h for double-barrier options with the jump matrix computed by the one-point and two-point Gaussian quadrature, as well as analytically, as described in §3. The error plots are in log-log scale and clearly demonstrate the Ch^2 convergence of the finite-element approximation. For Merton's (1976) model, using the one-point Gaussian quadrature to compute the jump matrix produces errors nearly indistinguishable from the case where the analytical expressions for b_{ij} are used. For Kou's (2002) model, using the two-point Gaussian quadrature improves the constant in the error estimate Ch^2 without affecting the order. The two-point Gaussian quadrature produces errors nearly indistinguishable from the case where analytical expressions are used.

While the double-barrier problem has a bounded domain and no localization is needed, for single-barrier and European options, we localize to bounded computational domains. In particular, for down-and-out options, the domain is $[\underline{x}, \bar{x}_k]$, where the lower barrier $\underline{x}$ is fixed contractually and a sequence of increasing artificial upper barriers $\bar{x}_k$ is considered (denoted by $x\max$ in the plots). Similarly, for up-and-out options, the upper barrier $\bar{x}$ is fixed contractually, and a sequence of decreasing artificial lower barriers $\underline{x}_k$ is considered (denoted by $x\min$ in the plots). When the computational domain is fixed and h is refined, there is a minimum error beyond which no further error reduction can be obtained by refining h. This is the localization error corresponding to this computational domain. The computational domain needs to be enlarged to obtain further error reduction. This can be clearly seen in the two plots in the second row for down-and-out options and in the two plots in the third row for up-and-out options. For each fixed computational domain, these plots exhibit the Ch^2 convergence up until the localization error starts to dominate. The localization error itself decays exponentially in the size of the computational domain, as shown in the two plots in the last row, where the localization error is plotted in log-scale.

To study temporal convergence in integrating the ODE system (3.7), for the purpose of comparison we also consider an IMEX midpoint scheme that is second-order accurate (Bader and Deuflhard 1983; Hairer and Wanner 1996, Chapter IV). Let the number of time steps N be even. Then, the IMEX midpoint scheme proceeds as follows:

$$\mathbb{M}\mathbf{u}^0 = \vec{b},$$

$$(\mathbb{M} + k\mathbb{A})\mathbf{u}^1 = (\mathbb{M} + k\mathbb{B})\mathbf{u}^0 - k\mathbf{F},$$

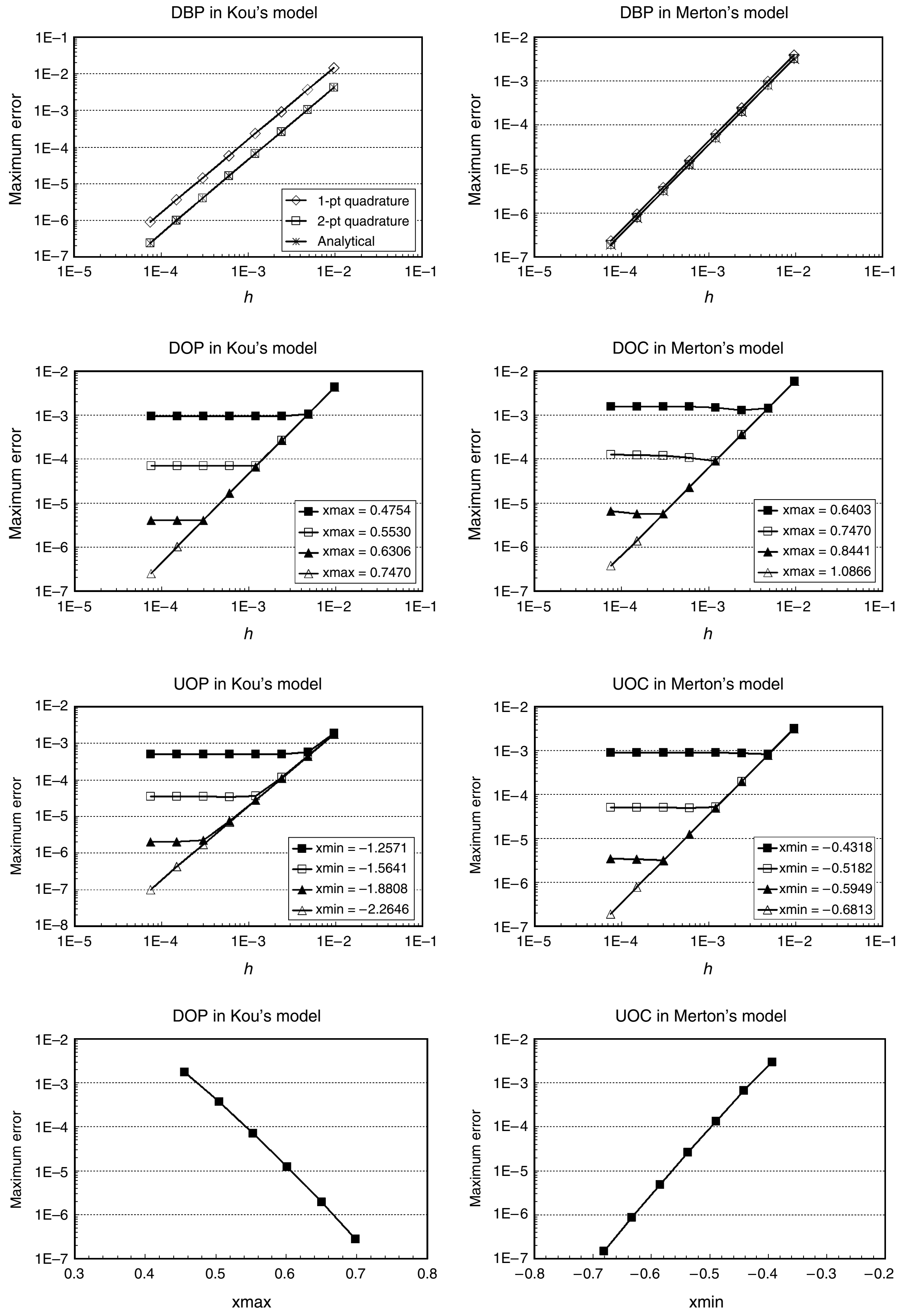
$$(\mathbb{M} + k\mathbb{A})\mathbf{u}^{i+1} = (\mathbb{M} - k\mathbb{A})\mathbf{u}^{i-1} + 2k\mathbb{B}\mathbf{u}^i - 2k\mathbf{F}, \qquad i = 1, 2, \ldots, N,$$

$$\mathbf{u}^N = \tfrac{1}{2}(\mathbf{u}^{N-1} + \mathbf{u}^{N+1}).$$

Table 1. Parameter values used in numerical experiments.

Kou	$\sigma = 0.1$, $\lambda = 3$, $p = 0.3$, $\eta_1 = 40$, $\eta_2 = 12$
Merton	$\sigma = 0.1$, $\lambda = 3$, $m = -0.05$, $s = 0.086$
DEVG	$\sigma = 0.1$, $s = 0.16$, $\nu = 0.1$, $\theta = -0.2$
SVCJ	$\lambda = 4$, $\nu = 0.02$, $m = -0.04$, $s = 0.06$, $\rho_D = -0.5$, $\rho_J = -0.5$, $\xi = 0.1$, $\kappa = 4$, $\theta = 0.04$
Other parameters	$K = 100$, $L = 80$, $U = 120$, $r = 5\%$, $q = 2\%$

Figure 1. Maximum norm pricing errors in Kou's (2002) and Merton's (1976) models: Spatial discretization and localization.



It starts with one IMEX Euler step, continues on with midpoint steps in which the stiff diffusion part is treated implicitly and the nonstiff jump part is treated explicitly, and ends with a smoothing step. The error of this scheme is shown to be $O(k^2)$ (Bader and Deuflhard 1983, Theorem 2).

Figure 2 illustrates convergence of the temporal discretization. We consider a double-barrier put option in Kou's (2002) model and a down-and-out call option in Merton's (1976) model, both with one year to maturity. Similar results were obtained for other types of options. The spatial discretization step size h was taken small enough to guarantee spatial discretization error less than 10^{-5}, our target accuracy. For this fixed h, the benchmark values of the solution of the system of ODEs are computed using small enough time steps. For the extrapolation scheme, two basic steps of six months each were taken ($H = 0.5$). Time discretization errors are plotted for the first-order IMEX Euler scheme, the second-order IMEX midpoint scheme, and our extrapolation scheme for both examples. The two plots in the first row illustrate the $O(k)$ convergence of the IMEX Euler scheme, the $O(k^2)$ convergence of the IMEX midpoint scheme, and faster than polynomial convergence of our extrapolation scheme. For the one-year double-barrier put option, it takes our extrapolation scheme a total of 72 time steps (in 0.05 seconds on a Dell Xeon 3.06 GHz PC) to achieve our target accuracy of 10^{-5}, while it takes the IMEX midpoint scheme 768 time steps (in 0.49 seconds), and the IMEX Euler scheme about 1.3 million time steps (in 873 seconds) to achieve the same accuracy. For the one-year down-and-out call option, it takes the extrapolation scheme a total of 110 time steps (in 0.49 seconds) to achieve 2×10^{-6} accuracy, while it takes 1,536 steps for the IMEX midpoint scheme (in 5.82 seconds) and about 5.24 millions steps (in 5.58 hours) for the IMEX Euler scheme to achieve the same accuracy (note that computation times for the down-and-out call are longer than for the double-barrier call with the same number of time steps due to the larger computational domain needed to accurately localize the down-and-out call problem, which is originally posed on the unbounded domain). The two plots in the second row experimentally illustrate that our scheme exhibits error decay $O(\exp(-c\mathcal{N}^{1/2}\ln\mathcal{N}))$ in the number of time steps $\mathcal{N}$, consistent with our discussion in §4. The basic step size $H = 0.5$ is sufficient to achieve our 10^{-5} accuracy goal in both examples.

To further illustrate the extrapolation scheme, an extrapolation tableau example is given in Table 2. Here, we consider a three-month down-and-out at-the-money put option in Kou's model ($S = K = 100$, $T = 0.25$ year). Consider a target accuracy goal of 10^{-5}. The benchmark price is 1.8595375 (computed with a fine enough finite-element mesh and a large enough number of time steps). IMEX Euler schemes with $n_1 = 1$ and $n_2 = 2$ steps produce the first two numbers in column 2 of Table 2 (T_{11} and T_{21}, respectively). Next, the value T_{22} is produced by extrapolation and the subdiagonal error estimate is computed (given in the $\mathscr{E}$-column). Because the subdiagonal error estimate is greater than 10^{-5}, the IMEX Euler scheme with $n_3 = 3$

Figure 2. Maximum norm pricing errors in Kou's (2002) and Merton's (1976) models: Time discretization.

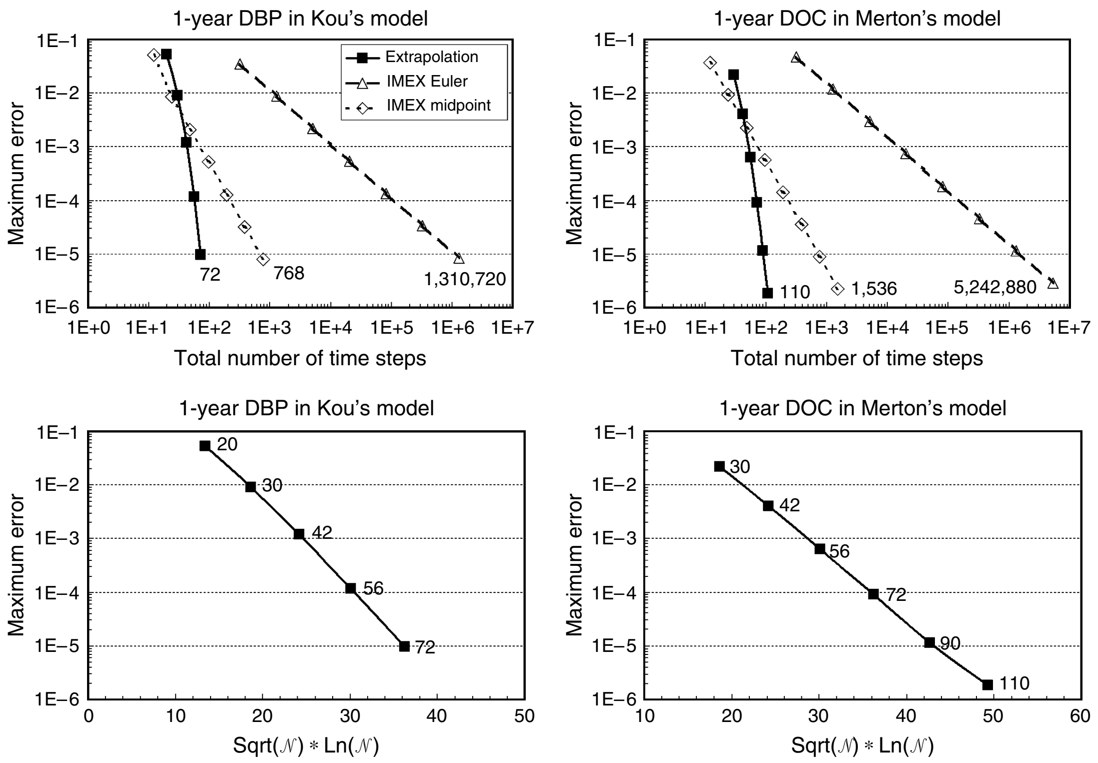


Table 2. Extrapolation based on the IMEX Euler scheme.

n_i	Euler	Extrapolation							$\mathscr{E}$	Error
1	1.771739									
2	1.881621	1.991503							1.1E−01	1.3E−01
3	1.898934	1.933560	1.904588						2.9E−02	4.5E−02
4	1.900516	1.905261	1.876963	1.867754					9.2E−03	8.2E−03
5	1.898484	1.890355	1.867995	1.862016	1.860582				1.4E−03	1.0E−03
6	1.895680	1.881658	1.864266	1.860537	1.859797	1.859640			1.6E−04	1.0E−04
7	1.892892	1.876169	1.862445	1.860017	1.859628	1.859560	1.859546		1.3E−05	8.8E−06
8	1.890342	1.872490	1.861452	1.859796	1.859574	1.859543	1.859537	1.859536	1.3E−06	2.0E−06

Notes. n_i: number of IMEX Euler steps, $\mathscr{E}$: subdiagonal error estimate. The column labeled Error gives the error between the approximation and the benchmark solution.

steps is used to produce T_{31}. Next, the extrapolation algorithm produces T_{32} and T_{33}, and the next subdiagonal error estimate is computed. We continue down the extrapolation tableau until the subdiagonal error estimate is below 10^{-5}, as shown in the last row of Table 2. We obtain the option price within our accuracy goal of 10^{-5} in seven extrapolation stages with a total of $\mathcal{N} = 1 + 2 + \cdots + 8 = 36$ time steps. The actual errors between the computed and the benchmark solution are given in the last column.

Figure 3 illustrates the valuation of Bermudan options. Here, we consider a one-year monthly monitored put option in Kou's (2002) and Merton's (1976) jump-diffusion models, $T = 1$, $N_B = 12$. The two plots in the first row show the option value functions of the Bermudan options, as well as the option payoff functions. The corresponding European put option value functions are also plotted for comparison. The two plots in the second row show the $O(h^2)$ convergence of the finite-element spatial discretization errors for both the European puts and Bermudan puts. The two plots in the third row show the convergence of the IMEX Euler, the IMEX midpoint, and the extrapolation time-stepping schemes. In these plots, the step size h is taken so that the spatial discretization error is about 10^{-5}. For the Bermudan put in Kou's (2002) model, it takes the extrapolation scheme only 252 time steps to achieve 3×10^{-6} accuracy (in 1.34 seconds), while it takes the second-order IMEX midpoint scheme 6,144 steps (in 28.67 seconds), and the first-order IMEX Euler scheme more than 3 million steps (in 3.36 hours) to achieve the same accuracy. Similarly, for the Bermudan put in Merton's (1976) model, it takes the extrapolation scheme 252 time steps (in 0.53 seconds) to achieve 2×10^{-6} accuracy in contrast to 7,680 steps for the IMEX midpoint scheme (in 12.09 seconds) and 4.7 million steps for the IMEX Euler scheme (in 2.09 hours). As these results demonstrate, the extrapolation method is also very efficient for Bermudan options in jump-diffusion models. In the special case of Merton's (1976) jump-diffusion model, Broadie and Yamamoto's (2005) method based on the fast Gauss transform (FGT) is remarkably fast and accurate for the pricing of Bermudan options. We have compared our numerical results with Broadie and Yamamoto's and found agreement to 10^{-6}. In the special case of Merton's (1976) model, Broadie and Yamamoto's (2003) method is more efficient than our method, because it uses analytical transition densities between exercise dates, while we solve the ODE between exercise dates. However, our method applies to general jump-diffusion models, while the FGT method is restricted to the processes with transition densities given by sums of Gaussians.

Figure 4 shows the application of our method to the diffusion-extended VG model. We consider a double-barrier knock-out put option. The option value as a function of the underlying asset price is shown on the left. The plot on the right shows the convergence of the three time discretization schemes. While it takes 364 steps (in 1.59 seconds) for the extrapolation scheme to achieve 10^{-5} accuracy in time discretization, it takes the second-order IMEX midpoint scheme 8,192 steps (in 38.77 seconds) and the IMEX Euler scheme 655,360 steps (in 2,979 seconds) to achieve the same accuracy. As expected, our extrapolation approach also performs well for the diffusion-extended VG model with infinite activity jumps.

Figure 5 shows option pricing in the SVCJ model. We consider a double-barrier put option and a down-and-out put option in the SVCJ model, both with six months to expiration. The two plots in the first row show the option prices as functions of the underlying stock price and volatility. The two plots in the second row verify the second-order convergence of the finite-element spatial discretization. They also show that using 1×1 Gaussian quadrature to compute the jump matrix (an $n \times n$ Gaussian quadrature rule denotes a tensor product of n-point Gaussian quadrature rules in each of the two state variables) is sufficient to guarantee the second-order accuracy of the finite-element discretization. The 2×2 GQ rule improves the constant C in the Ch^2 error estimate, without affecting the order. The two plots in the third row show the convergence of the time discretization schemes. The spatial step size is taken so that the spatial discretization error is less than 2×10^{-4}. Maximum norm errors of the IMEX Euler scheme, the IMEX midpoint scheme, and the extrapolation scheme with the basic step sizes $H = T = 0.5$ year are plotted. The extrapolation scheme is remarkably fast and accurate for this two-dimensional application with bivariate jumps. It takes only 66 time steps (in 55 seconds) to achieve 2×10^{-5} accuracy in time discretization, while it takes the IMEX

Figure 3. Bermudan put options in Kou's (2002) and Merton's (1976) jump-diffusion models.

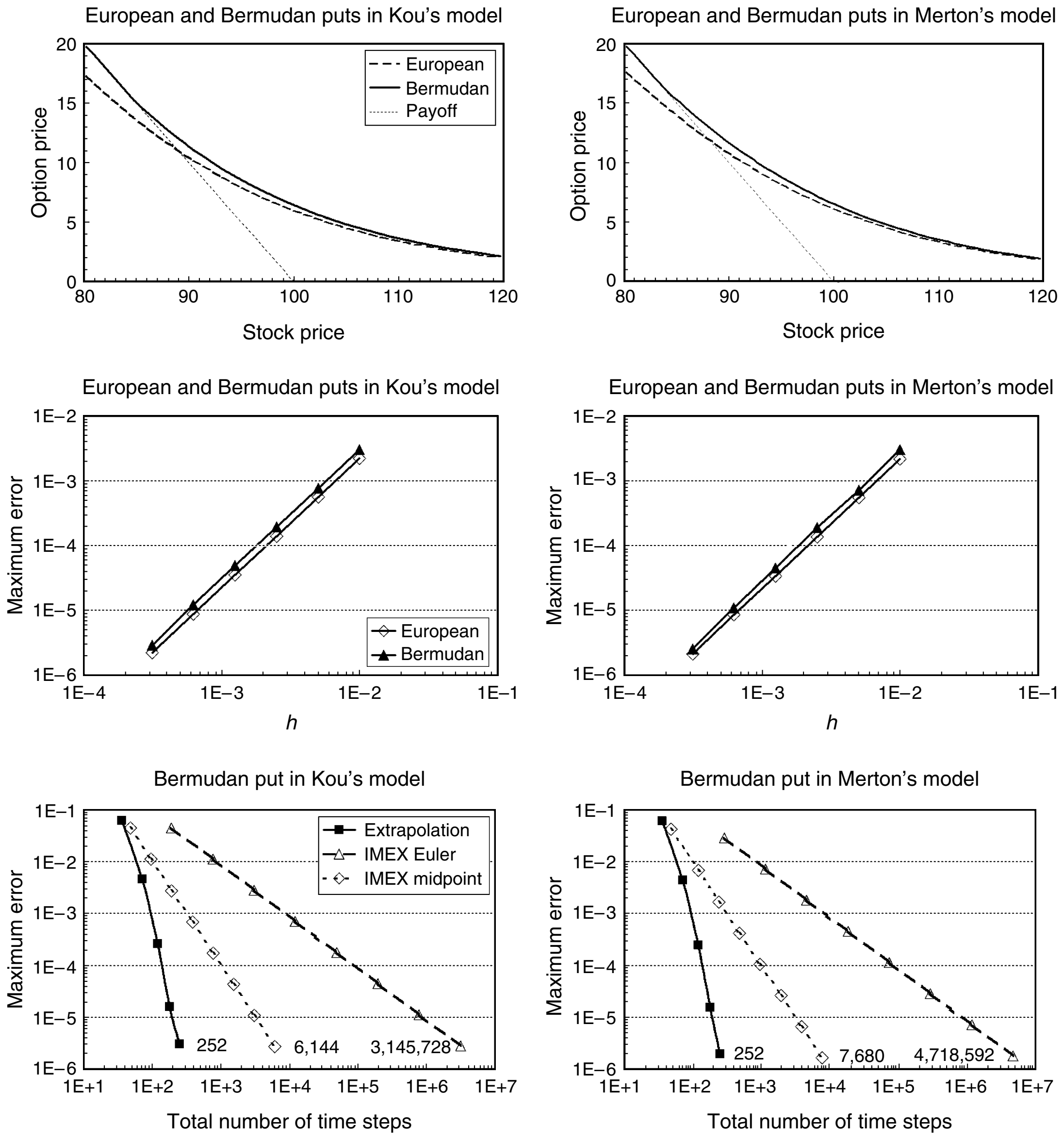


Figure 4. Double barrier knock-out puts in the diffusion extended variance gamma model.

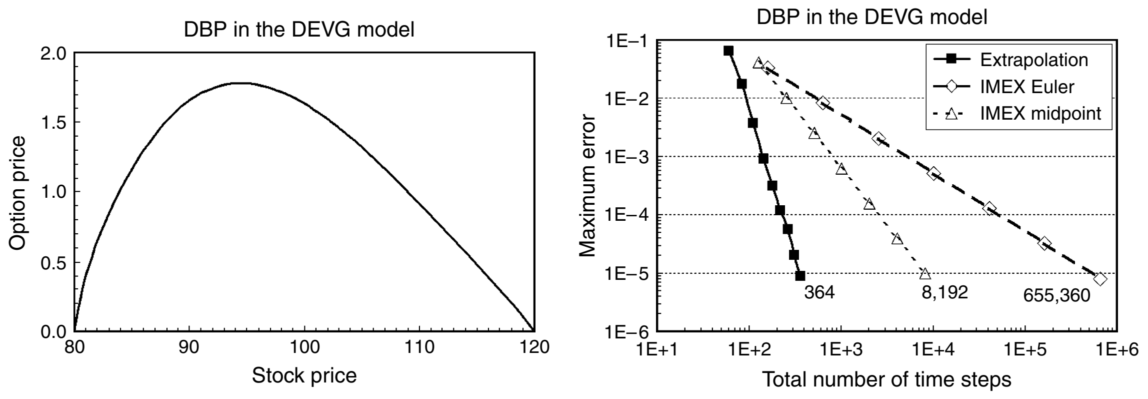
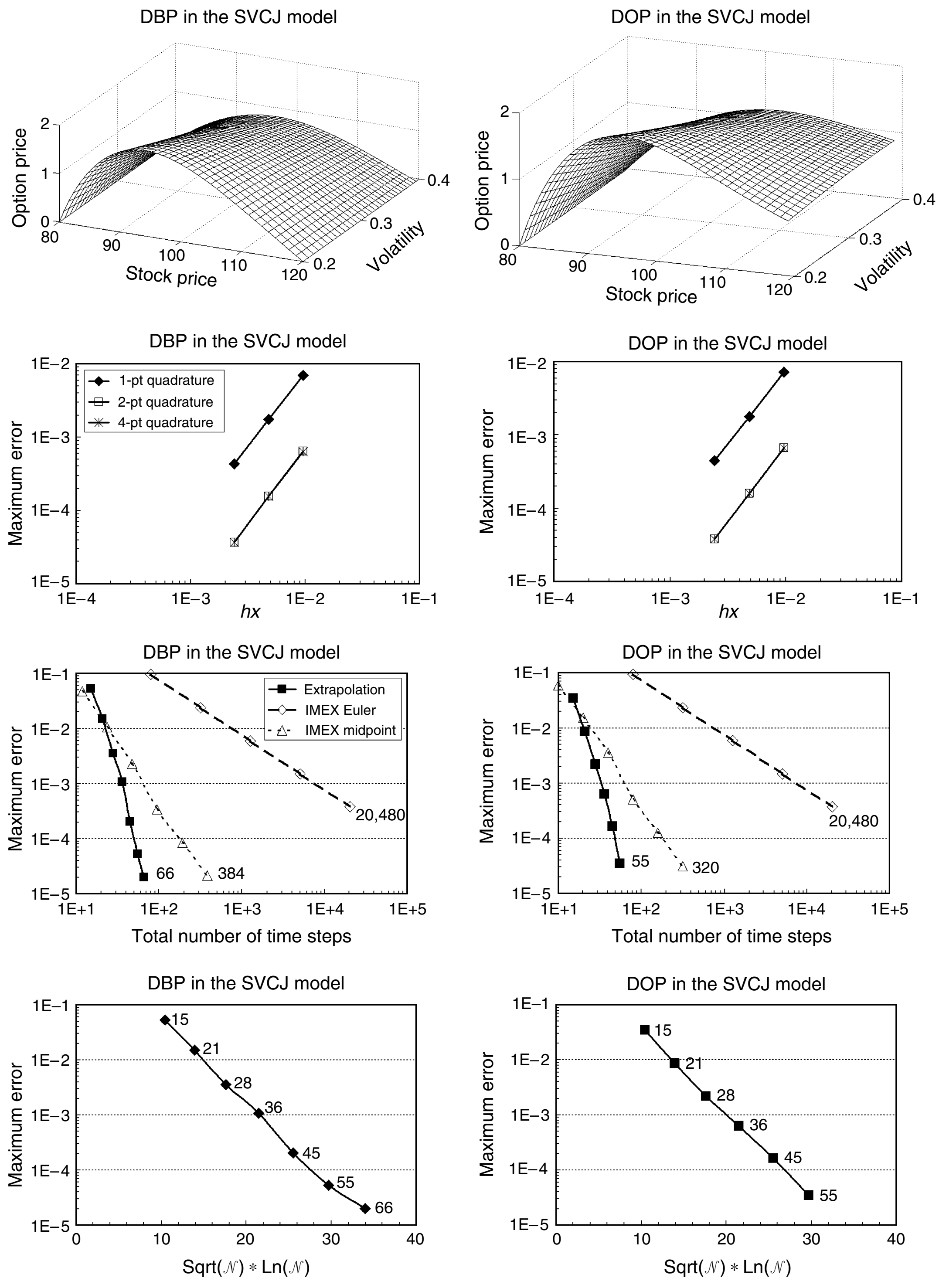


Figure 5. Option pricing in the SVCJ model.



midpoint scheme 384 steps (in 137 seconds) to achieve the same accuracy. Similarly, for the down-and-out put option, it takes 55 steps for the extrapolation scheme (in 118 seconds) to achieve 3×10^{-5} accuracy in time, while it takes the IMEX midpoint scheme 320 time steps (in 273 seconds) to achieve the same accuracy. The first-order IMEX Euler scheme is very slow and inefficient in this two-dimensional application. The last two plots experimentally illustrate the $O(\exp(-c\mathcal{N}^{1/2}\ln\mathcal{N}))$ error decay in the total number of time steps $\mathcal{N}$ in the SVCJ model.

Table 3 presents sample prices for one-year double-barrier knock-out puts, down-and-out puts, up-and-out puts in Kou's (2002) model (DBP-K, DOP-K, and UOP-K, respectively), double-barrier knock-out calls, down-and-out calls,

Table 3. Knock-out barrier and Bermudan option prices in one-dimensional and two-dimensional jump-diffusion models.

	$S_0 = 85$	$S_0 = 90$	$S_0 = 95$	$S_0 = 100$	$S_0 = 105$	$S_0 = 110$	$S_0 = 115$
DBP-K	1.76406	1.90376	1.69610	1.37753	1.02413	0.66641	0.32168
DOP-K	1.76965	1.92246	1.74565	1.48584	1.22547	0.99347	0.79752
UOP-K	13.63634	10.41784	7.82736	5.75775	4.06491	2.61282	1.29118
DBC-M	1.01355	1.56686	1.92151	1.96473	1.69452	1.20480	0.61221
DOC-M	1.99844	3.76860	6.10282	8.97505	12.30970	16.02425	20.03950
UOC-M	1.16174	1.65662	1.97628	1.99620	1.71238	1.21482	0.61707
DBP-DEVG	1.16193	1.65360	1.77859	1.63371	1.31929	0.91583	0.47745
BerP-K	15.06947	11.36619	8.54786	6.41713	4.82248	3.63468	2.75053
BerP-M	15.29539	11.66226	8.76732	6.51385	4.79963	3.51690	2.56802
DBP-SVCJ	1.15530	1.65210	1.75470	1.59510	1.27580	0.87230	0.43580
DOP-SVCJ	1.16370	1.67710	1.81480	1.72020	1.50580	1.25130	1.00390

and up-and-out calls in Merton's (1976) model (DBC-M, DOC-M, and UOC-M, respectively), one-year double-barrier knock-out puts (DBP-DEVG) in the diffusion-extended VG model, one-year Bermudan puts in Kou's (2002) and Merton's (1976) models (BerP-K and BerP-M), and six-month double-barrier knock-out puts and down-and-out puts in the SVCJ model (DBP-SVCJ and DOP-SVCJ) for seven different underlying asset prices. Model parameters are given in Table 1. The prices are computed with 10^{-5} accuracy for the one-dimensional models and 10^{-4} accuracy for the SVCJ.

7. Conclusions

This paper proposes a new computational method for the valuation of options in jump-diffusion models. The option value function for European and barrier options satisfies a PIDE. This PIDE is commonly integrated in time by IMEX time discretization schemes, where the differential (diffusion) term is treated implicitly, while the integral (jump) term is treated explicitly. In particular, the popular IMEX Euler scheme is first-order accurate. Second-order accuracy can be achieved by using the IMEX midpoint scheme. In contrast to the above approaches, we propose a new high-order time discretization scheme for the PIDE based on the extrapolation approach to the solution of ODEs, that also treats the diffusion term implicitly and the jump term explicitly. The scheme is simple to implement, can be added to any existing PIDE solver based on the IMEX Euler scheme, and is remarkably fast and accurate. We demonstrate our approach on the examples of Merton's (1976) and Kou's (2002) jump-diffusion models, the diffusion-extended VG model, as well as the two-dimensional Duffie-Pan-Singleton model with correlated and contemporaneous jumps in the asset return and in volatility. By way of example, pricing a one-year double-barrier option in Kou's (2002) jump-diffusion model, our extrapolation scheme attains accuracy of 10^{-5} in 72 time steps (in 0.05 seconds). In contrast, it takes the first-order IMEX Euler scheme more than 1.3 million time steps (in 873 seconds) and the second-order IMEX midpoint scheme 768 time steps (in 0.49 seconds) to attain the same accuracy. Combining simplicity of implementation and remarkable gains in computational efficiency, we expect this method to be very attractive to financial engineers.

8. Electronic Companion

An electronic companion to this paper is available as part of the online version that can be found at http://or.journal.informs.org/.

Endnote

1. Note that we include the convection term λU in the PIDE in the definition of the operator $\mathscr{A}$ to be treated implicitly, so only the integral is included in the definition of the operator $\mathscr{B}$ to be treated explicitly. The resulting IMEX scheme is proved to be unconditionally stable by d'Halluin et al. (2005). This is in contrast with the IMEX scheme in Zhang (1997), who treats the convection term explicitly as well, resulting in the stability condition of the form $\Delta t \leqslant C$ with some constant C.

Acknowledgment

This research was supported by the National Science Foundation under grants DMS-0223354 and DMI-0422937.

References

Amin, K. I. 1993. Jump diffusion option valuation in discrete-time. *J. Finance* **48** 1833–1863.

Andersen, L., J. Andreasen. 2000. Jump-diffusion processes: Volatility smile fitting and numerical methods for option pricing. *Rev. Derivatives Res.* **4** 231–262.

Asmussen, S., J. Rosinski. 2001. Approximations of small jumps of Lévy processes with a view towards simulation. *J. Appl. Probab.* **38** 482–493.

Bader, G., P. Deuflhard. 1983. A semi-implicit mid-point rule for stiff systems of ordinary differential equations. *Numerische Mathematik* **41** 373–398.

Bates, D. 1996. Jumps and stochastic volatility: Exchange rate processes implicit in Deutsche mark options. *Rev. Financial Stud.* **9**(1) 69–107.

Bates, D. 2000. Post-'87 crash fears in the S&P 500 futures option market. *J. Econometrics* **94** 181–238.

Bensoussan, A., J. L. Lions. 1984. *Impulse Control and Quasi-Variational Inequalities*. Gauthier-Villars, Paris, France.

Broadie, M., J. Detemple. 1996. American option valuation: New bounds, approximations, and a comparison of existing methods. *Rev. Financial Stud.* **9**(4) 1211–1250.

Broadie, M., O. Kaya. 2006. Exact simulation of stochastic volatility and other affine jump diffusion processes. *Oper. Res.* **54**(2) 217–231.

Broadie, M., Y. Yamamoto. 2003. Application of the fast Gauss transform to option pricing. *Management Sci.* **49**(8) 1071–1088.

Broadie, M., Y. Yamamoto. 2005. A double-exponential fast Gauss transform for pricing discrete path-dependent options. *Oper. Res.* **53**(5) 764–779.

Carr, P., A. Hirsa. 2003. Why be backward? Forward equations for American options. *Risk* **16**(1) 103–107.

Carr, P., D. Madan. 1999. Option valuation using the fast Fourier transform. *J. Comput. Finance* **2**(4) 61–73.

Ciarlet, P. G. 1978. *The Finite Element Method for Elliptic Problems*. North-Holland, Amsterdam, The Netherlands.

Cont, R., P. Tankov. 2004. *Financial Modeling with Jump Processes*. Chapman & Hall/CRC, Boca Raton, FL.

Cont, R., E. Voltchkova. 2005. A finite-difference scheme for option pricing in jump diffusion and exponential Lévy processes. *SIAM J. Numer. Anal.* **43**(4) 1596–1626.

Davis, P. 1994. *Circulant Matrices*, 2nd ed. Chelsea Publishing Company, New York.

Deuflhard, P. 1985. Recent progress in extrapolation methods for ordinary differential equations. *SIAM Rev.* **27**(4) 505–535.

Deuflhard, P., F. Bornemann. 2002. *Scientific Computing with Ordinary Differential Equations*. Springer, Berlin, Germany.

d'Halluin, Y., P. A. Forsyth, G. Labahn. 2004. A penalty method for American options with jump diffusion processes. *Numerische Mathematik* **97**(2) 321–352.

d'Halluin, Y., P. A. Forsyth, K. R. Vetzal. 2005. Robust numerical methods for contingent claims under jump diffusion processes. *IMA J. Numer. Anal.* **25** 87–112.

Duffie, D., J. Pan, K. Singleton. 2000. Transform analysis and asset pricing for affine jump-diffusions. *Econometrica* **68**(6) 1343–1376.

Eraker, B., M. Johannes, N. Polson. 2003. The impact of jumps in volatility and returns. *J. Finance* **58**(3) 1269–1300.

Eydeland, A. 1994. A fast algorithm for computing integrals in function spaces: Financial applications. *Comput. Econom.* **7**(4) 277–285.

Feng, L., V. Linetsky. 2007. Pricing discretely monitored barrier options and defaultable bonds in Lévy process models: A Hilbert transform approach. *Math. Finance.* Forthcoming.

Glasserman, P., N. Merener. 2003. Numerical solution of jump-diffusion LIBOR market models. *Finance Stochastics* **7** 1–27.

Hairer, E., G. Wanner. 1996. *Solving Ordinary Differential Equations II: Stiff and Differential-Algebraic Problems*, 2nd ed. Springer, Berlin, Germany.

Heston, S. 1993. A closed-form solution for options with stochastic volatility with applications to bond and currency options. *Rev. Financial Stud.* **6**(2) 327–343.

Hilber, H., A.-M. Matache, C. Schwab. 2005. Sparse wavelet methods for option pricing under stochastic volatility. *J. Comput. Finance* **8**(4) 1–42.

Hirsa, A., D. B. Madan. 2003. Pricing American options under variance gamma. *J. Comput. Finance* **7**(2) 63–80.

Horn, R. A., C. R. Johnson. 1994. *Topics in Matrix Analysis*. Cambridge University Press, Cambridge, UK.

Hundsdorfer, W., J. G. Verwer. 2003. *Numerical Solution of Time-Dependent Advection-Diffusion-Reaction Equations*. Springer, Berlin, Germany.

Ibanez, A. 2003. Robust pricing of the American put option: A note on Richardson extrapolation and the early exercise premium. *Management Sci.* **49**(9) 1210–1228.

Johnson, C. 1987. *Numerical Solution of Partial Differential Equations by the Finite Element Method*. Cambridge University Press, Cambridge, UK.

Kangro, R., R. Nicolaides. 2000. Far field boundary conditions for Black-Scholes equations. *SIAM J. Numer. Anal.* **38**(4) 1357–1368.

Kou, S. G. 2002. A jump-diffusion model for option pricing. *Management Sci.* **48**(8) 1086–1101.

Kou, S. G., H. Wang. 2003. First passage times of a jump diffusion process. *Adv. Appl. Probab.* **35** 504–531.

Kou, S. G., H. Wang. 2004. Option pricing under a double exponential jump-diffusion model. *Management Sci.* **50** 1178–1192.

Larsson, S., V. Thomée. 2003. *Partial Differential Equations with Numerical Methods*. Springer, Berlin, Germany.

Madan, D. B., F. Milne. 1991. Option pricing with V. G. Martingale components. *Math. Finance* **1**(4) 39–55.

Madan, D. B., E. Seneta. 1990. The variance gamma (V. G.) model for share market returns. *J. Bus.* **63**(4) 511–524.

Madan, D. B., P. Carr, E. Chang. 1998. The variance gamma process and option pricing. *Eur. Finance Rev.* **2** 79–105.

Marcozzi, M. 2001. On the approximation of optimal stopping problems with application to financial mathematics. *SIAM J. Sci. Comput.* **22**(5) 1865–1884.

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

Matache, A.-M., C. Schwab, T. P. Wihler. 2006. Linear complexity solution of parabolic integro-differential equations. *Numerische Mathematik*. Forthcoming.

Matache, A.-M., C. Schwab, T. P. Wihler. 2005b. Fast numerical solution of parabolic integro-differential equations with applications in finance. *SIAM J. Sci. Comput.* **27**(2) 369–393.

Matache, A.-M., T. von Petersdorff, C. Schwab. 2004. Fast deterministic pricing of options on Lévy driven assets. *Math. Model. Numer. Anal.* **38**(1) 37–72.

Merton, R. 1976. Option pricing when underlying stock returns are discontinuous. *J. Financial Econom.* **3** 125–144.

Petrella, G., S. G. Kou. 2004. Numerical pricing of discrete barrier and lookback options via Laplace transforms. *J. Comput. Finance* **8** 1–37.

Quarteroni, A., A. Valli. 1997. *Numerical Approximation of Partial Differential Equations*. Springer, Berlin, Germany.

Tavella, D., C. Randall. 2000. *Pricing Financial Instruments: The Finite Difference Method*. Wiley, New York.

Thomée, V. 1997. *Galerkin Finite Element Methods for Parabolic Problems*. Springer, Berlin, Germany.

Van Loan, C. 1992. *Computational Frameworks for the Fast Fourier Transform*. SIAM, Philadelphia, PA.

Vogel, C. R. 2002. *Computational Methods for Inverse Problems*. SIAM, Philadelphia, PA.

Zhang, X.-L. 1997. Numerical analysis of American option pricing in a jump-diffusion model. *Math. Oper. Res.* **22**(3) 668–690.