SIAM J. SCI. COMPUT.
Vol. 36, No. 5, pp. B817–B834

AN IMEX-SCHEME FOR PRICING OPTIONS UNDER STOCHASTIC VOLATILITY MODELS WITH JUMPS*

SANTTU SALMI†, JARI TOIVANEN‡, AND LINA VON SYDOW§

Abstract. Partial integro-differential equation (PIDE) formulations are often preferable for pricing options under models with stochastic volatility and jumps, especially for American-style option contracts. We consider the pricing of options under such models, namely the Bates model and the so-called stochastic volatility with contemporaneous jumps (SVCJ) model. The nonlocality of the jump terms in these models leads to matrices with full matrix blocks. Standard discretization methods are not viable directly since they would require the inversion of such a matrix. Instead, we adopt a two-step implicit-explicit (IMEX) time discretization scheme, the IMEX-CNAB scheme, where the jump term is treated explicitly using the second-order Adams–Bashforth (AB) method, while the rest is treated implicitly using the Crank–Nicolson (CN) method. The resulting linear systems can then be solved directly by employing LU decomposition. Alternatively, the systems can be iterated under a scalable algebraic multigrid (AMG) method. For pricing American options, LU decomposition is employed with an operator splitting method for the early exercise constraint or, alternatively, a projected AMG method can be used to solve the resulting linear complementarity problems. We price European and American options in numerical experiments, which demonstrate the good efficiency of the proposed methods.

Key words. option pricing, stochastic volatility model, jump-diffusion model, finite difference method, implicit-explicit time discretization

AMS subject classifications. 35K85, 65M06, 65M55, 65Y20, 91B25

DOI. 10.1137/130924905

1. Introduction.

Since the pioneering papers by Black and Scholes [6] and Merton [33], the shortcomings of their original model have become clear. It is well known that fitting empirically observed option prices into the Black–Scholes model typically implies a volatility distribution with a smile-like shape. This volatility smile becomes more pronounced near the maturity date. The usual modifications of the Black–Scholes model to explain such implied volatility patterns include models with jumps and/or stochastic volatility. The underlying asset price can be completely modeled by an infinite activity model such as the Carr–Geman–Madan–Yor (CGMY) model [8]. Over long time intervals the component behaving like the geometric Brownian motion becomes the dominant part in the model. For options with long maturities, the stochastic volatility models, for example the Heston model [24], are often regarded as more appropriate. For options with short maturities, however, jumps become increasingly important as a purely geometric Brownian motion driven process would require extremely high levels of volatility to explain the pronounced volatility smile pattern. Well-known jump-diffusion models in the literature include the Merton [34]

*Submitted to the journal's Computational Methods in Science and Engineering section October 25, 2013; accepted for publication (in revised form) June 13, 2014; published electronically October 2, 2014.
http://www.siam.org/journals/sisc/36-5/94290.html

†Department of Mathematical Information Technology, P.O. Box 35 (Agora), FI-40014 University of Jyväskylä, Jyväskylä, Finland (santtu.salmi@jyu.fi).
‡Institute for Computational and Mathematical Engineering, Stanford University, Stanford, CA 94305 (toivanen@stanford.edu), and Department of Mathematical Information Technology, P.O. Box 35 (Agora), FI-40014 University of Jyväskylä, Jyväskylä, Finland (jari.toivanen@jyu.fi).
§Department of Information Technology, Uppsala University, Box 337, 751 05 Uppsala, Sweden (lina@it.uu.se).

and Kou [30] models.

Bates proposed a more complete model in [5] with stochastic volatility and jumps in returns, essentially a combination of the Heston and Merton models. Duffie, Pan, and Singleton [15] followed up with arguably a more realistic model with stochastic volatility and jumps in both returns and volatility. They considered independently arriving jumps (SVIJ) and contemporaneously arriving jumps (SVCJ). These models were further investigated by Eraker, Johannes, and Polson in [17]. They concluded that SVIJ and SVCJ provide the best fit, but SVCJ is easier to estimate since jumps in returns and volatility are simultaneous.

In this paper we consider the numerical pricing of options under the Bates model (also known as the SVJ model) and the SVCJ model. A partial integro-differential equation (PIDE) can be derived for the price of a European option under the Bates and SVCJ models. Similarly, the price of an American option can be obtained by formulating a linear complementarity problem (LCP) with the same operator. Under the Heston model, numerical methods based on partial differential formulations for option pricing have been considered in the 1990s, for example, in [10, 54]. More recently similar methods have been generalized for the Bates model in [4, 9, 51, 37, 48, 41] and pricing European options under the SVJC model have been studied in [18, 53].

The finite difference and finite element methods are the most common ways to discretize the spatial operator; see [1]. For example, in [4, 18, 37, 41, 53] finite elements have been employed, while in [9, 51, 48, 36, 39, 35, 40] finite differences are used. The discretization of the integral operator leads to matrices with full matrix blocks; for simplicity such matrices are called full matrices in the following. Standard implicit time discretization schemes lead to systems of equations with these full matrices. Efficient iterative methods can be used in combination with implicit discretization schemes, such as in [50, 45], for example. A more attractive approach is to employ special implicit-explicit (IMEX) time discretization schemes which treat the jump term explicitly and the rest implicitly. A first-order accurate IMEX-Euler scheme for option pricing under jump-diffusion models was proposed in [12]. Second-order accurate IMEX schemes, for example, in [3, 20] have been applied for option pricing in [18, 31, 32, 46]. In [47] a family of IMEX time discretization schemes was analyzed in the context of option pricing under jump-diffusion models. In particular the IMEX-CNAB scheme was found to be fast and accurate, while having desirable stability properties. Here we employ this time discretization scheme for the Bates and SVCJ models.

The implicit treatment of the partial differential operator leads to block tridiagonal systems which are of LCP form for American options. While the projected successive over-relaxation (PSOR) method [13] can be easily employed to solve these systems, its efficiency deteriorates, i.e., the number of required iterations grows when discretizations are refined. A scalable alternative is a well-designed multigrid method which requires a constant number of iterations also with refined discretizations. The first such method for LCPs was the projected full approximation scheme (PFAS) multigrid method introduced by Brandt and Cryer in [7]. This method was used to price American options by Clarke and Parrott in [10], and Oosterlee in [38]. The projected multigrid (PMG) method for LCPs introduced by Reisinger and Wittum in [43] resembles more closely multigrid methods for linear systems. The PFAS and PMG methods are so-called geometrical multigrid methods which require a sequence of hierarchical grids. An easier-to-use alternative is algebraic multigrid (AMG) methods

[44, 49] which automatically generate a sequence of coarser problems. The projected algebraic multigrid (PAMG) method proposed in [52] is a generalization of these algebraic methods for LCPs. This is one of the methods that we employ for the solution of LCPs.

The other method that we will use is the operator splitting method proposed in [25] and employed for the Heston model in [27, 28]. This method approximates the LCP as a system of linear equations and a set of simple decoupled linear complementarity problems. A popular alternative approximation is the penalty method which was considered for the Heston model by Zvan, Forsyth, and Vetzal in [54] and since then by many authors for various option pricing models. For the block tridiagonal systems of linear equations many different methods can employed. In this paper, we will show that these systems can be solved very efficiently using a modern sparse direct solver when the time-step is constant. In this case, an LU decomposition of the coefficient-matrix needs to be formed only once. For two-dimensional problems like the underlying one, George showed in his 1973 paper [22] that the decomposition can be formed using $\mathcal{O}(m^3)$ operations and the solution each time-step requires $\mathcal{O}(m^2 \log m)$ operations, where m is of the same order as the number of grid-points in both directions. As the number of time-steps is usually also of the same order, i.e., $\sim m$, the computational complexity of the time-stepping is greater than the one required by forming the LU decomposition.

Explicit treatment of the jumps leads to matrix-vector multiplications with a full matrix. These multiplications can be performed efficiently by employing FFT based implementations. For a logarithmically uniform grid, the jump matrix under the SVJ model is a Toeplitz matrix, which can be embedded into a circulant matrix, as in [2], for example. The multiplication can then be computed by an FFT and an inverse FFT, which require only $\mathcal{O}(m \log m)$ operations for each grid line. This approach is more involved for the SVCJ model, where FFTs need to be performed in two directions. Again under the SVCJ model the jump matrix can be embedded into a block circulant matrix with circulant blocks (BCCB); see [18], for example. The FFTs in both directions require $\mathcal{O}(m^2 \log m)$ operations. Here we describe and employ this approach with the SVCJ model.

2. Option pricing model.

2.1. Governing equations. First, we consider a model with stochastic volatility and jumps in returns described by Bates [5]. Under this model the behavior of the asset value s and its variance v is described by the coupled stochastic differential equations

$$\begin{aligned} ds &= \mu_B s dt + \sqrt{v} s dw_1 + s dJ, \\ dv &= \kappa(\theta - v)dt + \sigma\sqrt{v} dw_2. \end{aligned} \tag{2.1}$$

Here μ_B is the drift rate defined by $\mu_B = r - q - \lambda\xi_B$, where $r \geq 0$ is the risk-free interest rate and $q \geq 0$ is the continuous dividend yield. The jump process J is a compound Poisson process with intensity $\lambda > 0$ and $J + 1$ has a log-normal distribution $p(y)$ with the mean in $\log y$ being γ and the variance in $\log y$ being δ^2, i.e., the probability density function is given by $p(y) = \frac{1}{\sqrt{2\pi}y\delta} e^{-\frac{(\log y - \gamma)^2}{2\delta^2}}$. The parameter ξ_B is defined by $\xi_B = e^{\gamma + \delta^2/2} - 1$. The variance v has mean level θ, κ is the rate of reversion to the mean level of v, and σ is the volatility of the variance v. The two Wiener processes w_1 are w_2 have the correlation ρ.

By combining derivative pricing arguments from [11, 19] for the Bates model, we can obtain the PIDE (formulated in forward time)

$$\begin{aligned}\frac{\partial u}{\partial \tau} &= \frac{1}{2}vs^2\frac{\partial^2 u}{\partial s^2} + \rho\sigma vs\frac{\partial^2 u}{\partial s\partial v} + \frac{1}{2}\sigma^2 v\frac{\partial^2 u}{\partial v^2} + (r-q-\lambda\xi_B)s\frac{\partial u}{\partial s} + \kappa(\theta - v)\frac{\partial u}{\partial v} \\ &\quad - (r+\lambda)u + \lambda\int_0^\infty u(sy, v, \tau)p(y)dy =: L_D^B u + L_I^B u,\end{aligned} \tag{2.2}$$

where u is the price of a European option and $\tau = T - t$ is the time to expiry. The operators L_D^B and L_I^B are defined as the differential part (including the term $-(r+\lambda)u$) and the integral part of (2.2), respectively. The initial condition for (2.2) is defined by

$$u = g(s), \tag{2.3}$$

where g is the pay-off function which gives the value of the option at the expiry.

Next, we allow for jumps in the volatility and study SVCJ. Then we have

$$\begin{aligned} ds &= \mu_S s dt + \sqrt{v}s dw_1 + s dJ^s, \\ dv &= \kappa(\theta - v)dt + \sigma\sqrt{v}dw_2 + dJ^v. \end{aligned} \tag{2.4}$$

Now $\mu_S = r - q - \lambda\xi_S$, where ξ_S is defined by $\xi_S = e^{\gamma+\delta^2/2}(1-\nu\rho_J)^{-1} - 1$ and ρ_J defines the correlation between jumps in returns and variance. The two-dimensional jump process (J^s, J^v) is an $\mathbb{R}\times\mathbb{R}^+$-valued compound Poisson process with intensity $\lambda > 0$. The 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, $J^s + 1$ has a log-normal distribution $p(z^s, z^v)$ with the mean in $\log z^s$ being $\gamma + \rho_J z^v$. This gives a bivariate probability density function defined by $p(z^s, z^v) = \frac{1}{\sqrt{2\pi}z^s\delta\nu}e^{-\frac{z^v}{\nu} - \frac{(\log z^s - \gamma - \rho_J z^v)^2}{2\delta^2}}$.

As in [15, 18], we assume that the price of a European option under the SVCJ model can be obtained as the solution to the PIDE

$$\begin{aligned}\frac{\partial u}{\partial \tau} &= \frac{1}{2}vs^2\frac{\partial^2 u}{\partial s^2} + \rho\sigma vs\frac{\partial^2 u}{\partial s\partial v} + \frac{1}{2}\sigma^2 v\frac{\partial^2 u}{\partial v^2} + (r-q-\lambda\xi_S)s\frac{\partial u}{\partial s} + \kappa(\theta - v)\frac{\partial u}{\partial v} \\ &\quad - (r+\lambda)u + \lambda\int_0^\infty\int_0^\infty u(s\cdot z^s, v + z^v, \tau)p(z^s, z^v)dz^v dz^s =: L_D^S u + L_I^S u.\end{aligned} \tag{2.5}$$

The initial condition is defined by (2.3).

For computational reasons we truncate the unbounded domain to $(s, v, \tau) \in (0, s_{\max})\times(0, v_{\max})\times(0, T]$. We impose the boundary conditions $u(0, v, \tau) = e^{-r\tau}g(0)$ and $\frac{\partial u}{\partial v}(s, v_{\max}, \tau) = 0$. For put options we impose $u(s_{\max}, v, \tau) = 0$. Furthermore, for integrations we extend u for $v > v_{\max}$ as $u(s, v, \tau) = u(s, v_{\max}, \tau)$. For put options, we extend u for $s > s_{\max}$ as $u(s, v, \tau) = 0$. On the boundary $v = 0$, the PIDEs (2.2) and (2.5) themselves can be posed as a boundary condition; see [16] for a discussion on this boundary and its treatment.

2.2. Linear complementarity problem for American options. In the following, L_D and L_I denote the differential operator L_D^B or L_D^S, and the integral operator L_I^B or L_I^S, respectively. With this notation both (2.2) and (2.5) can be expressed as $\frac{\partial u}{\partial \tau} - L_D u - L_I u =: \mathcal{L}u = 0$. The price u of an American option satisfies the LCP defined by

$$\begin{cases} \mathcal{L}u \geq 0, \quad u \geq 0, \\ (\mathcal{L}u)(u - g) = 0. \end{cases} \tag{2.6}$$

We impose the same boundary conditions as for (2.2) and (2.5) except on the boundary $s = 0$ the boundary condition is without discounting, i.e., $u(0, v, \tau) = g(0)$.

3. Discretization.

3.1. Computational grid. We will use a computational grid that is uniform in τ and nonuniform in s and v. The spatial grid is denoted (s_i, v_j), $i = 1, \ldots, m_s$, $j = 1, \ldots, m_v$. We employ grid generating functions $s : [0, 1] \to [0, s_{\max}]$ and $v : [0, 1] \to [0, v_{\max}]$ to define the grid-points as $s_i = s((i-1)/(m_s - 1))$ and $v_j = v((j-1)/(m_v - 1))$.

There are many ways to choose the functions s and v. Here we use the quadratic functions

$$s(p) = ap^2 + bp, \quad p \in [0, 1], \qquad v(q) = cq^2 + dq, \quad q \in [0, 1], \tag{3.1}$$

with the coefficients a, b, c, and d defined by the following conditions. These quadratic functions offer one of the simplest ways to construct refined grids. Also other grid generating functions can be used equally well with the numerical methods described in the following. In order to have the end points at $s_{\max}$ and $v_{\max}$ the conditions $s(1) = s_{\max}$ and $v(1) = v_{\max}$ have to hold. We choose s to map the point p_K to the strike price K, i.e., $s(p_K) = K$. When this point satisfies $p_K < K/s_{\max}$ the grid is refined in the interval $[0, K]$. A similar condition can be defined also for v, but instead we require the grid to be α times finer at 0 than at $v_{\max}$. This leads to the condition $v'(0) = \alpha v'(1)$. These four conditions define the coefficients a, b, c, and d.

At a grid point $s_i = s((i-1)/(m_s - 1))$ the grid step to the left is denoted by $\Delta s_i^- = s_i - s_{i-1}$ and the grid step to the right is denoted by $\Delta s_i^+ = s_{i+1} - s_i$. The grid steps around v_j are denoted in the same way.

3.2. IMEX-discretization in time. In [47] a number of IMEX-discretization methods for option pricing problems under jump-diffusion models are examined. The authors found that the IMEX-CNAB method produced the smallest error among the methods they studied. This method can be seen as a modification of the popular Crank–Nicolson method with a second-order accurate explicit treatment for the integral operator. For these reasons, we will employ and promote this method here.

We start by taking a small even number $2\tilde{N}$ of Rannacher-style smoothing Euler steps [42, 23] with the time-step $\Delta\tau/2$ given by

$$\left(I - \frac{\Delta\tau}{2} L_D\right) u^{(n+1)/2} = \left(I + \frac{\Delta\tau}{2} L_I\right) u^{n/2}, \quad n = 0, \ldots, 2\tilde{N} - 1. \tag{3.2}$$

This is followed by IMEX-CNAB steps defined by

$$\left(I - \frac{\Delta\tau}{2} L_D\right) u^{n+1} = \left(I + \frac{\Delta\tau}{2} L_D + \frac{3\Delta\tau}{2} L_I\right) u^n - \frac{\Delta\tau}{2} L_I u^{n-1}, \quad n = \tilde{N}, \ldots, N. \tag{3.3}$$

3.3. Discretization of the differential operator. Here we construct a seven-point finite difference discretization for the differential operator

$$\begin{aligned} L_D u = {} & \frac{1}{2} v s^2 \frac{\partial^2 u}{\partial s^2} + \rho\sigma v s \frac{\partial^2 u}{\partial s \partial v} + \frac{1}{2}\sigma^2 v \frac{\partial^2 u}{\partial v^2} \\ & + (r - q - \lambda\xi) s \frac{\partial u}{\partial s} + \kappa(\theta - v) \frac{\partial u}{\partial v} - (r + \lambda) u. \end{aligned} \tag{3.4}$$

We start with the mixed derivative and assume that the correlation ρ is negative. A similar seven-point finite difference discretization can also be constructed for a positive ρ; see [26, 27], for example. An alternative approach to construct finite difference stencils is considered in [29]. A Taylor-expansion of $u(s_{i+1}, v_{j-1}, \tau)$ and $u(s_{i-1}, v_{j+1}, \tau)$ around (s_i, v_j, τ) leads to, omitting the argument (s_i, v_j, τ) in the expressions,

$$
\begin{aligned}
u(s_{i+1}, v_{j-1}, \tau) \approx u &+ \Delta s_i^+ \frac{\partial u}{\partial s} - \Delta v_j^- \frac{\partial u}{\partial v} \\
&+ \frac{(\Delta s_i^+)^2}{2} \frac{\partial^2 u}{\partial s^2} - \Delta s_i^+ \Delta v_j^- \frac{\partial^2 u}{\partial s \partial v} + \frac{(\Delta v_j^-)^2}{2} \frac{\partial^2 u}{\partial v^2}, \\
u(s_{i-1}, v_{j+1}, \tau) \approx u &- \Delta s_i^- \frac{\partial u}{\partial s} + \Delta v_j^+ \frac{\partial u}{\partial v} \\
&+ \frac{(\Delta s_i^-)^2}{2} \frac{\partial^2 u}{\partial s^2} - \Delta s_i^- \Delta v_j^+ \frac{\partial^2 u}{\partial s \partial v} + \frac{(\Delta v_j^+)^2}{2} \frac{\partial^2 u}{\partial v^2},
\end{aligned} \tag{3.5}
$$

which gives

$$
\begin{aligned}
\frac{\partial^2 u}{\partial s \partial v} &\approx \frac{-u(s_{i+1}, v_{j-1}, \tau) + u + \Delta s_i^+ \frac{\partial u}{\partial s} - \Delta v_j^- \frac{\partial u}{\partial v} + \frac{(\Delta s_i^+)^2}{2} \frac{\partial^2 u}{\partial s^2} + \frac{(\Delta v_j^-)^2}{2} \frac{\partial^2 u}{\partial v^2}}{\Delta s_i^+ \Delta v_j^-}, \\
\frac{\partial^2 u}{\partial s \partial v} &\approx \frac{-u(s_{i-1}, v_{j+1}, \tau) + u - \Delta s_i^- \frac{\partial u}{\partial s} + \Delta v_j^+ \frac{\partial u}{\partial v} + \frac{(\Delta s_i^-)^2}{2} \frac{\partial^2 u}{\partial s^2} + \frac{(\Delta v_j^+)^2}{2} \frac{\partial^2 u}{\partial v^2}}{\Delta s_i^- \Delta v_j^+}.
\end{aligned} \tag{3.6}
$$

Using a weighted average of the approximations in (3.6) we obtain

$$
\begin{aligned}
\frac{\partial^2 u}{\partial s \partial v} \approx\ & w \frac{-u(s_{i+1}, v_{j-1}, \tau) + u + \Delta s_i^+ \frac{\partial u}{\partial s} - \Delta v_j^- \frac{\partial u}{\partial v} + \frac{(\Delta s_i^+)^2}{2} \frac{\partial^2 u}{\partial s^2} + \frac{(\Delta v_j^-)^2}{2} \frac{\partial^2 u}{\partial v^2}}{\Delta s_i^+ \Delta v_j^-} \\
&+ (1 - w) \frac{-u(s_{i-1}, v_{j+1}, \tau) + u - \Delta s_i^- \frac{\partial u}{\partial s} + \Delta v_j^+ \frac{\partial u}{\partial v} + \frac{(\Delta s_i^-)^2}{2} \frac{\partial^2 u}{\partial s^2} + \frac{(\Delta v_j^+)^2}{2} \frac{\partial^2 u}{\partial v^2}}{\Delta s_i^- \Delta v_j^+}.
\end{aligned} \tag{3.7}
$$

Using (3.7) in (3.4), we obtain the approximation

$$
\begin{aligned}
L_D u \approx\ & c_{ss} \frac{\partial^2 u}{\partial s^2} + c_{vv} \frac{\partial^2 u}{\partial v^2} + c_s \frac{\partial u}{\partial s} + c_v \frac{\partial u}{\partial v} \\
&+ \left(-(r + \lambda) + \rho \sigma v s \left(\frac{w}{\Delta s_i^+ \Delta v_j^-} + \frac{1 - w}{\Delta s_i^- \Delta v_j^+} \right) \right) u \\
&- \frac{w}{\Delta s_i^+ \Delta v_j^-} u(s_{i+1}, v_{j-1}, \tau) - \frac{1 - w}{\Delta s_i^- \Delta v_j^+} u(s_{i-1}, v_{j+1}, \tau),
\end{aligned} \tag{3.8}
$$

where the coefficients are

$$
\begin{aligned}
c_{ss} &= \frac{1}{2}\left(vs^2 + \rho\sigma vs\left(\frac{w\Delta s_i^+}{\Delta v_j^-} + \frac{(1-w)\Delta s_i^-}{\Delta v_j^+}\right)\right),\\
c_{vv} &= \frac{1}{2}\left(\sigma^2 v + \rho\sigma vs\left(\frac{w\Delta v_j^-}{\Delta s_i^+} + \frac{(1-w)\Delta v_j^+}{\Delta s_i^-}\right)\right),\\
c_s &= (r - q - \lambda\xi)s + \rho\sigma vs\left(\frac{w}{\Delta v_j^-} - \frac{1-w}{\Delta v_j^+}\right),\\
c_v &= \kappa(\theta - v) + \rho\sigma vs\left(-\frac{w}{\Delta s_i^+} + \frac{1-w}{\Delta s_i^-}\right).
\end{aligned}
\tag{3.9}
$$

Second-order finite difference discretizations of the second-derivatives are defined by

$$
\begin{aligned}
\frac{\partial^2 u}{\partial s^2}(s_i, v_j, \tau) &\approx 2\frac{\frac{1}{\Delta s_i^+}u(s_{i+1}, v_j, \tau) - \left(\frac{1}{\Delta s_i^+} + \frac{1}{\Delta s_i^-}\right)u(s_i, v_j, \tau) + \frac{1}{\Delta s_i^-}u(s_{i-1}, v_j, \tau)}{\Delta s_i^- + \Delta s_i^+},\\
\frac{\partial^2 u}{\partial v^2}(s_i, v_j, \tau) &\approx 2\frac{\frac{1}{\Delta v_j^+}u(s_i, v_{j+1}, \tau) - \left(\frac{1}{\Delta v_j^+} + \frac{1}{\Delta v_j^-}\right)u(s_i, v_j, \tau) + \frac{1}{\Delta v_j^-}u(s_i, v_{j-1}, \tau)}{\Delta v_j^- + \Delta v_j^+}.
\end{aligned}
\tag{3.10}
$$

Similarly, second-order finite differences for the first-order derivatives are defined by

$$
\begin{aligned}
\frac{\partial u}{\partial s}(s_i, v_j, \tau) &\approx \frac{\frac{\Delta s_i^-}{\Delta s_i^+}u(s_{i+1}, v_j, \tau) - \left(\frac{\Delta s_i^-}{\Delta s_i^+} - \frac{\Delta s_i^+}{\Delta s_i^-}\right)u(s_i, v_j, \tau) - \frac{\Delta s_i^+}{\Delta s_i^-}u(s_{i-1}, v_j, \tau)}{\Delta s_i^- + \Delta s_i^+},\\
\frac{\partial u}{\partial v}(s_i, v_j, \tau) &\approx \frac{\frac{\Delta v_j^-}{\Delta v_j^+}u(s_i, v_{j+1}, \tau) - \left(\frac{\Delta v_j^-}{\Delta v_j^+} - \frac{\Delta v_j^+}{\Delta v_j^-}\right)u(s_i, v_j, \tau) - \frac{\Delta v_j^+}{\Delta v_j^-}u(s_i, v_{j-1}, \tau)}{\Delta v_j^- + \Delta v_j^+}.
\end{aligned}
\tag{3.11}
$$

When a finite difference method is employed, spurious oscillations might occur when the discretization matrix of $-L_D$ does not lead to a so-called M-matrix. Sufficient conditions for an M-matrix are that it is strictly diagonally dominant with positive diagonal elements and it has nonpositive off-diagonal elements. To ensure that the off-diagonals are to be nonpositive, we add artificial diffusion when necessary. With the second-order differences (3.10) and (3.11), this leads to the modified diffusion coefficients

$$
\begin{aligned}
\tilde{c}_{ss} &= \max\left\{\tilde{c}_{ss},\ -\frac{1}{2}c_s\Delta s_i^-,\ \frac{1}{2}c_s\Delta s_i^+\right\},\\
\tilde{c}_{vv} &= \max\left\{\tilde{c}_{vv},\ -\frac{1}{2}c_v\Delta v_i^-,\ \frac{1}{2}c_v\Delta v_i^+\right\}.
\end{aligned}
\tag{3.12}
$$

When the original coefficients c_{ss} and c_{vv} are positive, this addition of artificial diffusion is equivalent to using one-sided first-order finite differences to discretize a part of the first-order derivatives.

In order not to add excessive diffusion to the discretization, it is desirable that the coefficients c_{ss} and c_{vv} are nonnegative. For a general weight $w \in [0,1]$, this leads to the bounds

$$-\frac{\rho}{\sigma} s_i \Delta v_j^+ \le \Delta s_i^- \le -\frac{1}{\rho\sigma} s_i \Delta v_j^+ \quad \text{and} \quad -\frac{\rho}{\sigma} s_i \Delta v_j^- \le \Delta s_i^+ \le -\frac{1}{\rho\sigma} s_i \Delta v_j^- \tag{3.13}$$

for the grid step sizes Δs_i^- and Δs_i^+. These bounds can be quite stringent when the correlation ρ approaches -1.

Note that the diagonal element of the discretization matrix is $r + \lambda$ minus the sum of the off-diagonal elements. Thus, the matrix is strictly diagonally dominant with positive diagonals when $r + \lambda > 0$ and the off-diagonal elements are nonpositive.

3.4. Discretization of the integral operator.

3.4.1. Bates' model. The integral term can be written as

$$I = \int_0^\infty u(sy, v, \tau) p(y) dy = \int_{-\infty}^\infty \bar{u}(z + x, v, \tau) \bar{p}(z) dz, \tag{3.14}$$

where $x = \log s$, $z = \log y$, $\bar{u}(z, v, \tau) = u(e^z, v, \tau)$, and $\bar{p}(z) = p(e^z)e^z$. Now, make the change of variable $\zeta = z + x$ and study the value of the integral at the point x_i,

$$\begin{aligned} I_i &= \int_{-\infty}^\infty \bar{u}(\zeta, v, \tau) \bar{p}(\zeta - x_i) d\zeta = \int_{x_{\min}}^{x_{\max}} \bar{u}(\zeta, v, \tau) \bar{p}(\zeta - x_i) d\zeta \\ &\quad + \int_{-\infty}^{x_{\min}} \bar{u}(\zeta, v, \tau) \bar{p}(\zeta - x_i) d\zeta + \int_{x_{\max}}^\infty \bar{u}(\zeta, v, \tau) \bar{p}(\zeta - x_i) d\zeta. \end{aligned} \tag{3.15}$$

The interval $(x_{\min}, x_{\max})$ is chosen to be so large that the impact of the two last integrals of (3.15) is negligible. For put options $x_{\max} = \log s_{\max}$ is a natural choice as $\bar{u}(\zeta, v, \tau) = 0$ for $\zeta \ge x_{\max}$ due to the extension of u for $s > s_{\max}$. The first part of the integral is evaluated using the trapezoidal quadrature rule on an equidistant grid in x with spacing Δx and m_x grid-points in $(x_{\min}, x_{\max})$ giving

$$I_i \approx \int_{x_{\min}}^{x_{\max}} \bar{u}(\zeta, v, \tau) \bar{p}(\zeta - x_i) d\zeta \approx \Delta x \sum_{j=1}^{m_x} \bar{u}(\zeta_j, v, \tau) \bar{p}(\zeta_j - x_i) = \bar{I}_i. \tag{3.16}$$

Note that the above approximation includes the additional terms $\frac{\Delta x}{2} \bar{u}(x_{\min}, v, \tau) \bar{p}(x_{\min} - x_i)$ and $\frac{\Delta x}{2} \bar{u}(x_{\max}, v, \tau) \bar{p}(x_{\max} - x_i)$, which are also assumed to be negligible. Computing all $\bar{I}_i$, $i = 1, \ldots, m_x$, can be accomplished by the matrix-vector multiplication defined by

$$\bar{\mathbf{I}} = \mathbf{T}_{m_x} \bar{\mathbf{u}},$$

where

$$\bar{\mathbf{I}} = \begin{pmatrix} \bar{I}_1 & \bar{I}_2 & \cdots & \bar{I}_{m_x - 1} & \bar{I}_{m_x} \end{pmatrix}^T, \qquad \bar{\mathbf{u}} = \begin{pmatrix} \bar{u}_1 & \bar{u}_2 & \cdots & \bar{u}_{m_x - 1} & \bar{u}_{m_x} \end{pmatrix}^T,$$

$$\mathbf{T}_{m_x} = \begin{pmatrix} \bar{p}(0) & \bar{p}(\Delta x) & \cdots & \bar{p}((m_x - 2)\Delta x) & \bar{p}((m_x - 1)\Delta x) \\ \bar{p}(-\Delta x) & \bar{p}(0) & \bar{p}(\Delta x) & \cdots & \bar{p}((m_x - 2)\Delta x) \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ \bar{p}(-(m_x - 2)\Delta x) & \cdots & \bar{p}(-\Delta x) & \bar{p}(0) & \bar{p}(\Delta x) \\ \bar{p}(-(m_x - 1)\Delta x) & \bar{p}(-(m_x - 2)\Delta x) & \cdots & \bar{p}(-\Delta x) & \bar{p}(0) \end{pmatrix}.$$

The matrix $\mathbf{T}_{m_x}$ is a so-called Toeplitz matrix with constant diagonals. Such matrices can be embedded in circulant matrices which for $\mathbf{T}_{m_x}$ yields

$$\mathbf{C}_{2m_x-1} = \left(\begin{array}{ccc|ccc} \bar{p}(0) & \cdots & \bar{p}((m_x-1)\Delta x) & \cdots & \bar{p}(-\Delta x) \\ \vdots & \ddots & & \ddots & \vdots \\ \bar{p}(-(m_x-1)\Delta x) & & \bar{p}(0) & & \bar{p}((m_x-1)\Delta x) \\ \hline \vdots & \ddots & & \ddots & \vdots \\ \bar{p}(\Delta x) & \cdots & \bar{p}(-(m_x-1)\Delta x) & \cdots & \bar{p}(0) \end{array}\right).$$

The computation of $\bar{\mathbf{I}} = \mathbf{T}_{m_x}\bar{\mathbf{u}}$ can be performed by first computing $\tilde{\mathbf{I}} = \mathbf{C}_{2m_x-1}\tilde{\mathbf{u}}$, where $\tilde{\mathbf{u}} = \begin{pmatrix} \bar{u}_1 & \bar{u}_2 & \cdots & \bar{u}_{m_x-1} & \bar{u}_{m_x} & 0 & \cdots & 0 \end{pmatrix}^T$. Then $\bar{\mathbf{I}}$ is given by the m_x first elements in $\tilde{\mathbf{I}}$.

The circulant matrix $\mathbf{C}_{2m_x-1}$ can be decomposed as $\mathbf{C}_{2m_x-1} = \mathbf{F}^{-1}_{2m_x-1}\mathbf{\Lambda}\mathbf{F}_{2m_x-1}$, where $\mathbf{F}_{2m_x-1}$ is a Fourier matrix of order $2m_x - 1$ and $\mathbf{\Lambda}$ is a diagonal matrix with the eigenvalues of $\mathbf{C}_{2m_x-1}$ on the diagonal. These eigenvalues can be computed by $\mathbf{F}_{2m_x-1}\mathbf{c}$, where $\mathbf{c}$ is the first column vector in $\mathbf{C}_{2m_x-1}$. From this we conclude that the multiplication of a vector $\mathbf{w}$ by the matrix $\mathbf{C}_{2m_x-1}$ can be computed as $\mathbf{F}^{-1}_{2m_x-1}\mathbf{\Lambda}\mathbf{F}_{2m_x-1}\mathbf{w}$. This can then be accomplished using two discrete Fourier transforms (DFTs) and one inverse discrete Fourier transform (IDFT). By embedding $\mathbf{T}_{m_x}$ in a circulant matrix $\mathbf{C}_{M_x}$ where M_x is the smallest power of 2 such that $M_x \geq 2m_x - 1$, the DFTs and IDFTs can be computed using fast Fourier transforms (FFTs) requiring $\mathcal{O}(M_x \log_2 M_x)$ arithmetic operations. If there are repeated matrix-vector multiplications with the matrix $\mathbf{C}_{M_x}$, then the eigenvalues of $\mathbf{C}_{M_x}$ can be precomputed once in the beginning.

We summarize the computation of the integral in (3.14) as follows:

- Interpolate values from the s_i grid-points to equidistant points x_i between $x_{\min}$ and $x_{\max}$.
- Compute and embed the matrix $\mathbf{T}_{m_x}$ into a circulant matrix $\mathbf{C}_{M_x}$.
- Compute $\bar{\mathbf{I}}$ using the algorithm described above using FFTs.
- Interpolate $\bar{I}_i$ to the original grid-points s_i.

3.4.2. SVCJ model. The integral term can be written as

$$\begin{aligned} I &= \int_0^\infty \int_0^\infty u(sz^s, v + z^v, \tau)p(z^s, z^v)dz^v dz^s \\ &= \int_{-\infty}^\infty \int_0^\infty \bar{u}(x + z^x, v + z^v, \tau)\bar{p}(z^x, z^v)dz^v dz^x, \end{aligned} \tag{3.17}$$

where $x = \log s$, $z^x = \log z^s$, $\bar{u}(x, v, \tau) = u(e^x, v, \tau)$, and $\bar{p}(z^x, z^v) = p(e^{z^x}, z^v)e^{z^x}$.

Now, make the changes of variables $\zeta = x + z^x$ and $\eta = v + z^v$ and study the value of the integral at the point (x_i, v_j),

$$I_{i,j} = \int_{-\infty}^\infty \int_{v_j}^\infty \bar{u}(\zeta, \eta, \tau)\bar{p}(\zeta - x_i, \eta - v_j)d\eta d\zeta. \tag{3.18}$$

Similarly to the treatment of the integral in the Bates model, we choose a rectangle $(x_{\min}, x_{\max}) \times (v_j, \bar{v}_{\max})$, which is large enough so that integrating over it gives a sufficiently good approximation for (3.18). The first part of the integral is evaluated

using the two-dimensional generalization of the trapezoidal rule on an equidistant grid in x and in v with spacing Δv and $\bar{m}_v$ grid-points giving

(3.19)
$$\int_{x_{\min}}^{x_{\max}} \int_{v_j}^{\bar{v}_{\max}} \bar{u}(\zeta,\eta,\tau)\bar{p}(\zeta - x_i, \eta - v_j)d\eta d\zeta \approx \frac{\Delta x \Delta v}{2} \sum_{k=1}^{m_x} \bar{u}(\zeta_k,\eta_j,\tau)\bar{p}(\zeta_k - x_i, 0)$$
$$+ \Delta x \Delta v \sum_{k=1}^{m_x} \sum_{\ell=j+1}^{\bar{m}_v} \bar{u}(\zeta_k,\eta_\ell,\tau)\bar{p}(\zeta_k - x_i, \eta_\ell - v_j) = \bar{I}_{i,j}.$$

By defining $\bar{\mathbf{I}} = \begin{pmatrix} \bar{\mathbf{I}}_1 & \bar{\mathbf{I}}_2 & \cdots & \bar{\mathbf{I}}_{m_x-1} & \bar{\mathbf{I}}_{m_x} \end{pmatrix}^T$, $\bar{\mathbf{u}} = \begin{pmatrix} \bar{\mathbf{u}}_1 & \bar{\mathbf{u}}_2 & \cdots & \bar{\mathbf{u}}_{m_x-1} & \bar{\mathbf{u}}_{m_x} \end{pmatrix}^T$, $\bar{\mathbf{I}}_\ell = \begin{pmatrix} \bar{I}_{1,\ell} & \bar{I}_{2,\ell} & \cdots & \bar{I}_{\bar{m}_v-1,\ell} & \bar{I}_{\bar{m}_v,\ell} \end{pmatrix}^T$, $\bar{\mathbf{u}}_\ell = \begin{pmatrix} \bar{u}_{1,\ell} & \bar{u}_{2,\ell} & \cdots & \bar{u}_{\bar{m}_v-1,\ell} & \bar{u}_{\bar{m}_v,\ell} \end{pmatrix}^T$, we can compute $\bar{\mathbf{I}}$ by $\bar{\mathbf{I}} = \mathbf{T}_{\bar{m}_v m_x}\bar{\mathbf{u}}$, where $\mathbf{T}_{\bar{m}_v m_x}$ is a block-Toeplitz matrix with Toeplitz blocks (BTTB-matrix) defined by

(3.20)
$$\mathbf{T}_{\bar{m}_v m_x} = \begin{pmatrix} \mathbf{T}_0 & \mathbf{T}_1 & \cdots & \mathbf{T}_{\bar{m}_v-1} \\ \mathbf{T}_{-1} & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \mathbf{T}_1 \\ \mathbf{T}_{-(\bar{m}_v-1)} & \cdots & \mathbf{T}_{-1} & \mathbf{T}_0 \end{pmatrix},$$
$$\mathbf{T}_\ell = \begin{pmatrix} T_{0,\ell} & T_{1,\ell} & \cdots & T_{m_x-1,\ell} \\ T_{-1,\ell} & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & T_{1,\ell} \\ T_{-(m_x-1),\ell} & \cdots & T_{-1,\ell} & T_{0,\ell} \end{pmatrix},$$

where

$$T_{k,\ell} = \begin{cases} 0, & \ell < 0, \\ \frac{1}{2}\Delta x \Delta v \bar{p}(k\Delta x, 0), & \ell = 0, \\ \Delta x \Delta v \bar{p}(k\Delta x, \ell \Delta v), & \ell > 0. \end{cases}$$

For a general block-circulant matrix with circulant blocks (BCCB-matrix) $\mathbf{C}_{M_v M_x}$ of order $M_v M_x$ defined by

$$\mathbf{C}_{M_v M_x} = \begin{pmatrix} \mathbf{C}_0 & \mathbf{C}_1 & \cdots & \mathbf{C}_{M_v-1} \\ \mathbf{C}_{M_v-1} & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \mathbf{C}_1 \\ \mathbf{C}_1 & \cdots & \mathbf{C}_{M_v-1} & \mathbf{C}_0 \end{pmatrix},$$
$$\mathbf{C}_\ell = \begin{pmatrix} C_{0,\ell} & C_{1,\ell} & \cdots & C_{M_x-1,\ell} \\ C_{M_x-1,\ell} & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & C_{1,\ell} \\ C_{1,\ell} & \cdots & C_{M_x-1,\ell} & C_{0,\ell} \end{pmatrix},$$

it holds that $\mathbf{C}_{M_v M_x} = (\mathbf{F}_{M_v} \otimes \mathbf{F}_{M_x})^{-1}\mathbf{\Lambda}(\mathbf{F}_{M_v} \otimes \mathbf{F}_{M_x})$. Following the discussion in the previous section we conclude that the multiplication of a vector $\mathbf{w}$ by a BCCB-matrix can be accomplished using two $2d$ DFTs and one $2d$ IDFT. Again, the DFTs

and IDFT can be computed efficiently using FFTs requiring $\mathcal{O}(M_v M_x \log_2(M_v M_x))$ arithmetic operations.

The BTTB-matrix $\mathbf{T}_{\bar{m}_v m_x}$ defined in (3.20) can be embedded in a BCCB-matrix $\mathbf{C}_{M_v M_x}$ where M_x is the smallest power of 2 such that $M_x \geq 2m_x - 1$ and M_v is the smallest power of 2 such that $M_v \geq 2\bar{m}_v - 1$. A multiplication of a vector $\mathbf{w}$ by the matrix $\mathbf{T}$ can then be replaced by the multiplication of the vector $\tilde{\mathbf{w}}$ by the matrix $\mathbf{C}$. Here $\tilde{\mathbf{w}}$ is defined as the extension of $\mathbf{w}$ by appending $M_x - m_x$ zeros to each block in $\mathbf{w}$ and $M_v - \bar{m}_v$ zero vectors of dimension M_x.

We summarize the computation of the integral in (3.18) as follows:

- Interpolate values from grid-points in (s, v) to equidistant points x_i between $x_{\min}$ and $x_{\max}$ and equidistant points v_j between 0 and $\bar{v}_{\max}$.
- Compute the BTTB-matrix $\mathbf{T}_{\bar{m}_v m_x}$ and embed this matrix in a BCCB-matrix $\mathbf{C}_{M_v M_x}$.
- Compute $\bar{\mathbf{I}}$ using the algorithm described above using FFTs.
- Interpolate $\bar{I}_{i,j}$ to the original grid in (s, v).

3.5. Numerical experiments. In this section we price European and American *put* options under the Bates and the SVCJ models. We compare two alternative approaches to solving the discretized systems: the algebraic multigrid (AMG) method, and the LU decomposition method. The operator splitting (OS) method [25, 28] is employed to enable the LU decomposition method for American options as well. Similarly, the projected algebraic multigrid (PAMG) method is adopted instead of the AMG for American options. These combinations lead to a total of eight different cases.

The used AMG implementation is based on the Ruge–Stüben algorithm [44] and the PAMG method is described in [52]. This algorithm generates automatically the coarse grid problems using only the left-hand side matrix. In the considered option prices problems, each coarsening reduces the size of the system to be between one-third and one-half. For the used grids the number of levels generated by the multigrid methods varies between 7 and 16. The AMG methods employ a multigrid V-cycle. The smoother for moving downwards and upwards is one (projected) Gauss–Seidel iteration over all points, followed by one more iteration over the so-called fine-points (F-points). For the LU decompositions, we employed the UMFPACK version 5.6.1 [14]. For FFTs, we employed FFTW version 3.3.2 [21]. We performed the experiments on a Mac laptop with 2 GHz Intel Core i7 processor and 8 GB of memory.

The shared model parameters employed in the numerical experiments are listed in Table 1. In addition, under the SVCJ model the correlation between jumps is set as $\rho_J = -0.5$, and the mean of the exponentially distributed jump sizes in variance is set as $\nu = 0.2$. The truncation boundary in the s-direction is $s_{\max} = 4K = 400$. The truncation boundary in the v-direction for the Bates model is $v_{\max} = 0.5$ and for the SVCJ model $v_{\max} = 3$. These truncations are chosen experimentally so that the error caused by them should not be larger than the discretization errors on the considered grids. The coefficients for the grid generating function s in (3.1) are defined by the parameter $p_K = 7/16$. The coefficients for the grid generating function v are defined by $\alpha = 2$ for the Bates model and $\alpha = 4$ for the SVCJ model. The truncation boundaries for the integrations are defined by $x_{\min} = \log s_2 - \frac{1}{8}(\log s_{\max} - \log s_2)$, $x_{\max} = \log s_{\max}$, and $\bar{v}_{\max} = \frac{5}{4} v_{\max}$. The number of grid-points in the equidistant integration grid is twice as many in each direction compared to the computational grid. Reference prices, listed in Table 2, were computed using the (P)AMG method on fine grids defined by $m_s = 4097$, $m_v = 2049$, $N = 513$ for the Bates model and by

TABLE 1
Shared parameters employed in numerical experiments for the Bates and the SVCJ model.

Parameter	Value
Brownian correlation ρ	-0.5
Risk-free interest rate r	0.03
Dividend yield q	0
Volatility of variance σ	0.25
Rate of mean reversal κ	2
Variance mean level θ	0.04
Strike price K	100
Jump arrival rate λ	0.2
Expiry time T	0.5
Jump size log-variance δ^2	0.16
Jump size log-mean γ	-0.5

TABLE 2
Reference prices at $s = \{90, 100, 110\}$ *and* $v = 0.04$.

Option type	Price at 90	Price at 100	Price at 110
European put (Bates)	11.302917	6.589881	4.191455
European put (SVCJ)	11.475480	6.928637	4.641829
American put (Bates)	11.619920	6.714240	4.261583
American put (SVCJ)	11.778543	7.046400	4.707443

$m_s = 4097$, $m_v = 4097$, $N = 513$ for the SVCJ model.

Figure 1 illustrates that the pricing errors of the (P)AMG and LU(+OS) approaches are almost identical. Option prices, ratios of consecutive errors, average iterations, and CPU times for the European options under the Bates and SVCJ models are listed in Tables 3 and 5 for the AMG approach, and in Tables 4 and 6 for the LU decomposition approach. Similarly, the numerical results for American options under the Bates and SVCJ models are reported in Tables 7 and 9 for the PAMG approach, and in Tables 8 and 10 for the LU+OS approach. The seven refinements in the plots of Figure 1 correspond to the seven grids described in Tables 3–10.

The ratios of consecutive errors are computed using the l_2 norm with respect to the reference prices. The ratios are, on average, of second order. Thus, this clearly suggests that the IMEX scheme is also second-order accurate as expected. With the LU+OS approach the accuracy is slightly better for the American option prices at 90 and 100 on the finest grids. The operator splitting method yields more accurate prices near the early exercise boundary in our experiments. Due to this the ratios for the LU+OS approach are better on the finest grids and it seems to converge faster on the refinement levels 6 and 7 in Figure 1.

As the grid is refined, the LU decomposition approach is slightly faster than the (P)AMG method. In both cases, however, penny-accurate prices can be obtained in a fraction of a second under the Bates model, and in about six seconds under the SVCJ model. On finer grids computing prices under the SVCJ model is up to 15 times more time consuming than under the Bates model. The two main reasons for this is the need to perform $2d$ FFTs instead of $1d$ FFTs and larger computational domain/grids for the SVCJ model. Under the SVCJ model, performing the integrations require an order of magnitude more time than solving the discretized PDEs or associated LCPs. Thus, the proposed IMEX scheme is much faster than any iterative procedure requiring multiple integrations per time step.

Figure 1 and Tables 2–10 show that the prices of American options are slightly

less accurate than for the European options on the same grid. The ratio of the l_2 errors of American and European options is larger on finer grids. For the (P)AMG and LU(+OS) approaches these ratios are less than 2 and 1.3, respectively. On this respect the methods behave very similarly under the Bates and SVCJ models. Thus, the discretizations are also rather accurate for American options. The CPU times for pricing European and American options are essentially the same in the tables. Thus, the PAMG and LU+OS methods offer very efficient way to solve the LCPs resulting from American options.

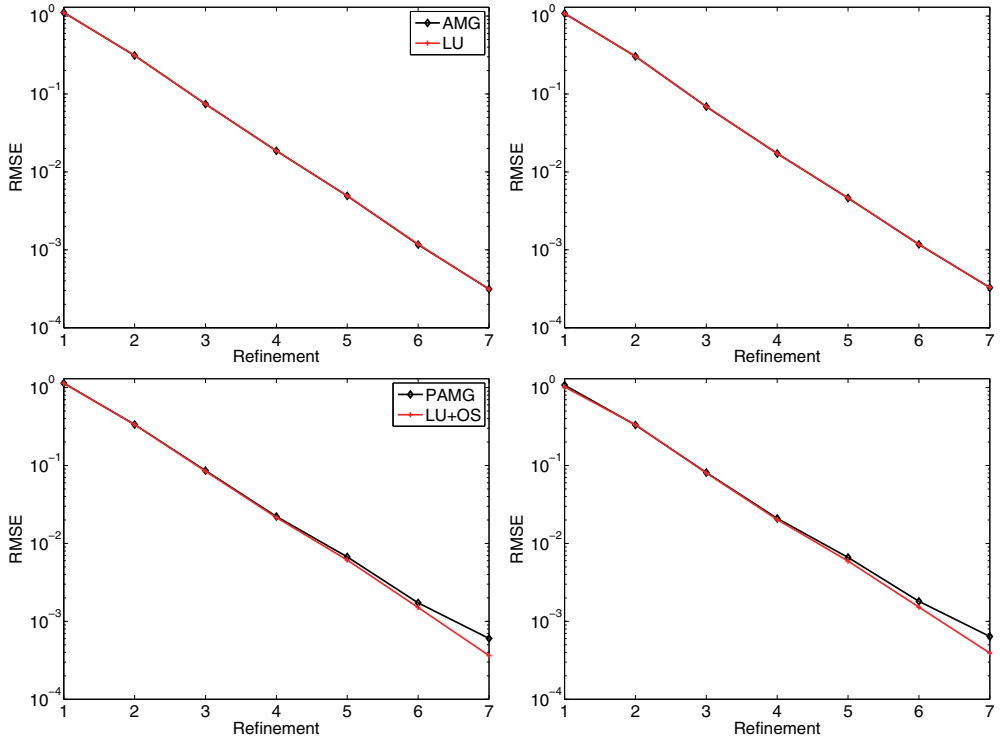

FIG. 1. *Root mean square errors under the Bates and SVCJ models computed with the (P)AMG and LU decomposition approaches: European option under the Bates model (top left), European option under the SVCJ model (top right), American option under the Bates model (bottom left), and American option under the SVCJ model (bottom right).*

TABLE 3

Average iterations, CPU times, ratios of convergence, and option prices at $s = \{90, 100, 110\}$ *and* $v = 0.04$ *for the European put option under the Bates model computed with AMG.*

Grid $m_s \times m_v \times N$	Price at 90	Price at 100	Price at 110	Avg. itns.	Ratio	CPU time(s)
$17 \times 9 \times 2$	12.3849	5.0124	3.9946	2.0		0.001
$33 \times 17 \times 4$	11.3310	6.0693	4.0525	2.0	3.56	0.002
$65 \times 33 \times 8$	11.3077	6.4645	4.1630	2.1	4.20	0.016
$129 \times 65 \times 16$	11.3103	6.5589	4.1862	3.2	3.99	0.137
$257 \times 129 \times 32$	11.3029	6.5821	4.1880	2.9	3.79	0.984
$513 \times 257 \times 64$	11.3035	6.5880	4.1908	2.2	4.22	6.746
$1025 \times 513 \times 128$	11.3026	6.5894	4.1914	2.8	3.72	64.584

TABLE 4

Average iterations, CPU times, ratios of convergence, and option prices at $s = \{90, 100, 110\}$ and $v = 0.04$ for the European put option under the Bates model computed with LU decomposition.

Grid $m_s \times m_v \times N$	Price at 90	Price at 100	Price at 110	Avg. itns.	Ratio	CPU time(s)
$17 \times 9 \times 2$	12.3849	5.0124	3.9946	–		0.001
$33 \times 17 \times 4$	11.3310	6.0693	4.0525	–	3.56	0.003
$65 \times 33 \times 8$	11.3077	6.4645	4.1630	–	4.20	0.013
$129 \times 65 \times 16$	11.3103	6.5589	4.1862	–	3.99	0.081
$257 \times 129 \times 32$	11.3029	6.5821	4.1880	–	3.78	0.586
$513 \times 257 \times 64$	11.3034	6.5880	4.1908	–	4.17	4.556
$1025 \times 513 \times 128$	11.3026	6.5894	4.1914	–	3.78	40.946

TABLE 5

Average iterations, CPU times, ratios of convergence, and option prices at $s = \{90, 100, 110\}$ and $v = 0.04$ for the European put option under the SVCJ model computed with AMG.

Grid $m_s \times m_v \times N$	Price at 90	Price at 100	Price at 110	Avg. itns.	Ratio	CPU time(s)
$17 \times 17 \times 2$	12.8806	5.6953	4.6756	3.0		0.002
$33 \times 33 \times 4$	11.5047	6.4238	4.4946	3.0	3.55	0.011
$65 \times 65 \times 8$	11.4806	6.8124	4.6159	3.0	4.41	0.083
$129 \times 129 \times 16$	11.4832	6.9001	4.6373	3.0	3.99	0.693
$257 \times 257 \times 32$	11.4755	6.9213	4.6386	3.1	3.73	6.548
$513 \times 513 \times 64$	11.4759	6.9268	4.6412	3.0	3.94	65.261
$1025 \times 1025 \times 128$	11.4752	6.9282	4.6417	3.0	3.58	650.509

TABLE 6

Average iterations, CPU times, ratios of convergence, and option prices at $s = \{90, 100, 110\}$ and $v = 0.04$ for the European put option under the SVCJ model computed with LU decomposition.

Grid $m_s \times m_v \times N$	Price at 90	Price at 100	Price at 110	Avg. itns.	Ratio	CPU time(s)
$17 \times 17 \times 2$	12.8806	5.6953	4.6756	–		0.004
$33 \times 33 \times 4$	11.5047	6.4238	4.4946	–	3.55	0.011
$65 \times 65 \times 8$	11.4806	6.8124	4.6159	–	4.42	0.072
$129 \times 129 \times 16$	11.4832	6.9001	4.6373	–	3.99	0.630
$257 \times 257 \times 32$	11.4755	6.9213	4.6386	–	3.73	5.842
$513 \times 513 \times 64$	11.4759	6.9268	4.6412	–	3.94	59.000
$1025 \times 1025 \times 128$	11.4752	6.9282	4.6417	–	3.58	616.475

TABLE 7

Average iterations, CPU times, ratios of convergence, and option prices at $s = \{90, 100, 110\}$ and $v = 0.04$ for the American put option under the Bates model computed with PAMG.

Grid $m_s \times m_v \times N$	Price at 90	Price at 100	Price at 110	Avg. itns.	Ratio	CPU time(s)
$17 \times 9 \times 2$	12.7091	5.0795	4.0487	2.0		0.001
$33 \times 17 \times 4$	11.5471	6.1571	4.1120	2.0	3.40	0.003
$65 \times 33 \times 8$	11.5941	6.5718	4.2277	2.1	3.91	0.017
$129 \times 65 \times 16$	11.6189	6.6767	4.2537	3.2	3.88	0.140
$257 \times 129 \times 32$	11.6169	6.7040	4.2569	2.9	3.29	1.015
$513 \times 257 \times 64$	11.6195	6.7115	4.2605	2.3	3.90	6.778
$1025 \times 513 \times 128$	11.6193	6.7135	4.2613	2.8	2.86	65.024

TABLE 8

Average iterations, CPU times, ratios of convergence, and option prices at $s = \{90, 100, 110\}$ and $v = 0.04$ for the American put option under the Bates model computed with the LU+OS approach.

Grid $m_s \times m_v \times N$	Price at 90	Price at 100	Price at 110	Avg. itns.	Ratio	CPU time(s)
$17 \times 9 \times 2$	12.6810	5.0757	4.0446	–		0.001
$33 \times 17 \times 4$	11.5537	6.1576	4.1115	–	3.38	0.003
$65 \times 33 \times 8$	11.5992	6.5736	4.2282	–	3.97	0.013
$129 \times 65 \times 16$	11.6217	6.6778	4.2541	–	3.92	0.080
$257 \times 129 \times 32$	11.6187	6.7047	4.2572	–	3.52	0.592
$513 \times 257 \times 64$	11.6204	6.7119	4.2606	–	4.06	4.582
$1025 \times 513 \times 128$	11.6197	6.7137	4.2614	–	4.14	41.348

TABLE 9

Average iterations, CPU times, ratios of convergence, and option prices at $s = \{90, 100, 110\}$ and $v = 0.04$ for the American put option under the SVCJ model computed with the PAMG.

Grid $m_s \times m_v \times N$	Price at 90	Price at 100	Price at 110	Avg. itns.	Ratio	CPU time(s)
$17 \times 17 \times 2$	13.1177	5.7494	4.7195	3.0		0.002
$33 \times 33 \times 4$	11.6958	6.5020	4.5477	3.0	3.25	0.011
$65 \times 65 \times 8$	11.7480	6.9127	4.6755	3.0	4.07	0.081
$129 \times 129 \times 16$	11.7759	7.0110	4.7001	3.0	3.88	0.708
$257 \times 257 \times 32$	11.7750	7.0365	4.7029	3.1	3.17	6.735
$513 \times 513 \times 64$	11.7779	7.0436	4.7063	3.0	3.66	65.655
$1025 \times 1025 \times 128$	11.7778	7.0456	4.7072	3.0	2.82	652.219

TABLE 10

Average iterations, CPU times, ratios of convergence, and option prices at $s = \{90, 100, 110\}$ and $v = 0.04$ for the American put option under the SVCJ model computed with the LU+OS approach.

Grid $m_s \times m_v \times N$	Price at 90	Price at 100	Price at 110	Avg. itns.	Ratio	CPU time(s)
$17 \times 17 \times 2$	12.9690	5.7407	4.7116	–		0.003
$33 \times 33 \times 4$	11.6884	6.4991	4.5455	–	2.47	0.010
$65 \times 65 \times 8$	11.7515	6.9138	4.6758	–	4.24	0.071
$129 \times 129 \times 16$	11.7787	7.0120	4.7005	–	3.95	0.643
$257 \times 257 \times 32$	11.7767	7.0372	4.7032	–	3.20	5.848
$513 \times 513 \times 64$	11.7788	7.0440	4.7065	–	3.94	59.131
$1025 \times 1025 \times 128$	11.7782	7.0458	4.7072	–	4.24	613.418

4. Conclusions. In this work we have considered the numerical pricing of options under the Bates model and the SVCJ model. For the time discretization we employed the second-order accurate IMEX-CNAB scheme which treats the differential operator implicitly and the integral term explicitly. This way we avoid having to solve systems of equations with dense matrices. The matrix-vector multiplications arising from the explicit jump terms can be computed efficiently using FFTs.

European options lead to linear systems of equations, which under the IMEX-CNAB scheme can be solved efficiently by employing AMG method or LU decomposition. For American options a common approach is to formulate the pricing problem as an LCP. Here we solve these LCPs by employing either PAMG or operator splitting method in combination with LU decomposition. Both these methods were employed to price European/American options under the Bates/SVCJ model, rendering a total setup of eight combinations.

Numerical experiments show that the (P)AMG and LU(+OS) approaches produce almost identical prices. Since the operator splitting method does not essentially reduce the accuracy in the solution and the LU decomposition can be precomputed prior to the time-stepping, the LU(+OS) methodology turns out to be surprisingly accurate and fast. With the proposed methods pricing American options is essentially equally fast and only a bit less accurate than pricing European options. Thus, the methods are especially efficient for pricing American options. To the best of our knowledge, this is the first paper where American options under the SVCJ model are priced.

REFERENCES

[1] Y. Achdou and O. Pironneau, *Computational Methods for Option Pricing*, Frontiers Appl. Math. 30, SIAM, Philadelphia, 2005.

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

[3] U. M. Ascher, S. J. Ruuth, and B. T. R. Wetton, *Implicit-explicit methods for time-dependent partial differential equations*, SIAM J. Numer. Anal., 32 (1995), pp. 797–823.

[4] L. V. Ballestra and C. Sgarra, *The evaluation of American options in a stochastic volatility model with jumps: An efficient finite element approach*, Comput. Math. Appl., 60 (2010), pp. 1571–1590.

[5] D. S. Bates, *Jumps and stochastic volatility: Exchange rate processes implicit Deutsche mark options*, Review Financial Stud., 9 (1996), pp. 69–107.

[6] F. Black and M. Scholes, *The pricing of options and corporate liabilities*, J. Political Economy, 81 (1973), pp. 637–654.

[7] A. Brandt and C. W. Cryer, *Multigrid algorithms for the solution of linear complementarity problems arising from free boundary problems*, SIAM J. Sci. Statist. Comput., 4 (1983), pp. 655–684.

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

[9] C. Chiarella, B. Kang, G. H. Meyer, and A. Ziogas, *The evaluation of American option prices under stochastic volatility and jump-diffusion dynamics using the method of lines*, Int. J. Theor. Appl. Finance, 12 (2009), pp. 393–425.

[10] N. Clarke and K. Parrott, *Multigrid for American option pricing with stochastic volatility*, Appl. Math. Finance, 6 (1999), pp. 177–195.

[11] R. Cont and P. Tankov, *Financial Modelling with Jump Processes*, Chapman & Hall/CRC, Boca Raton, FL, 2004.

[12] R. Cont and E. Voltchkova, *A finite difference scheme for option pricing in jump diffusion and exponential Lévy models*, SIAM Numer. Anal., 43 (2005), pp. 1596–1626.

[13] C. W. Cryer, *The solution of a quadratic programming problem using systematic overrelaxation*, SIAM J. Control, 9 (1971), pp. 385–392.

[14] T. A. Davis, *Algorithm* 832: *UMFPACK* V4.3—*an unsymmetric-pattern multifrontal method*, ACM Trans. Math. Software, 30 (2004), pp. 196–199.

[15] D. Duffie, J. Pan, and K. Singleton, *Transform analysis and asset pricing for affine jump-diffusions*, Econometrica, 68 (2000), pp. 1343–1376.
[16] E. Ekström and J. Tysk, *The Black-Scholes equation in stochastic volatility models*, J. Math. Anal. Appl., 368 (2010), pp. 498–507.
[17] B. Eraker, M. Johannes, and N. Polson, *The impact of jumps in volatility and returns*, J. Finance, 58 (2003), pp. 1269–1300.
[18] L. Feng and V. Linetsky, *Pricing options in jump-diffusion models: An extrapolation approach*, Oper. Res., 56 (2008), pp. 304–325.
[19] J.-P. Fouque, G. Papanicolaou, and K. R. Sircar, *Derivatives in Financial Markets with Stochastic Volatility*, Cambridge University Press, Cambridge, UK, 2000.
[20] J. Frank, W. Hundsdorfer, and J. G. Verwer, *On the stability of implicit-explicit linear multistep methods*, Appl. Numer. Math., 25 (1997), pp. 193–205.
[21] M. Frigo and S. G. Johnson, *The design and implementation of FFTW3*, in Proceedings of the IEEE, Vol. 93, IEEE, Piscataway, NJ, 2005, pp. 216–231.
[22] A. George, *Nested dissection of a regular finite element mesh*, SIAM J. Numer. Anal., 10 (1973), pp. 345–363.
[23] M. B. Giles and R. Carter, *Convergence analysis of Crank-Nicolson and Rannacher time-marching*, J. Comput. Finance, 9 (2006), pp. 89–112.
[24] S. Heston, *A closed-form solution for options with stochastic volatility with applications to bond and currency options*, Rev. Financial Stud., 6 (1993), pp. 327–343.
[25] S. Ikonen and J. Toivanen, *Operator splitting methods for American option pricing*, Appl. Math. Lett., 17 (2004), pp. 809–814.
[26] S. Ikonen and J. Toivanen, *Componentwise splitting methods for pricing American options under stochastic volatility*, Int. J. Theor. Appl. Finance, 10 (2007), pp. 331–361.
[27] S. Ikonen and J. Toivanen, *Efficient numerical methods for pricing American options under stochastic volatility*, Numer. Methods Partial Differential Equations, 24 (2008), pp. 104–126.
[28] S. Ikonen and J. Toivanen, *Operator splitting methods for pricing American options under stochastic volatility*, Numer. Math., 113 (2009), pp. 299–324.
[29] K. Ito and J. Toivanen, *Lagrange multiplier approach with optimized finite difference stencils for pricing American options under stochastic volatility*, SIAM J. Sci. Comput., 31 (2009), pp. 2646–2664.
[30] S. G. Kou, *A jump-diffusion model for option pricing*, Management Sci., 48 (2002), pp. 1086–1101.
[31] Y. Kwon and Y. Lee, *A second-order finite difference method for option pricing under jump-diffusion models*, SIAM J. Numer. Anal., 49 (2011), pp. 2598–2617.
[32] Y. Kwon and Y. Lee, *A second-order tridiagonal method for American options under jump-diffusion models*, SIAM J. Sci. Comput., 33 (2011), pp. 1860–1872.
[33] Y. Kwon and Y. Lee, *Theory of rational option pricing*, Bell J. Econom. Management Sci., 4 (1973), pp. 141–183.
[34] Y. Kwon and Y. Lee, *Option pricing when underlying stock returns are discontinuous*, J. Financial Econ., 3 (1976), pp. 125–144.
[35] G. Linde, J. Persson, and L. von Sydow, *A highly accurate adaptive finite difference solver for the Black-Scholes equation*, Int. J. Comput. Math., 86 (2009), pp. 2104–2121.
[36] P. Lötstedt, J. Persson, L. von Sydow, and J. Tysk, *Space-time adaptive finite difference method for European multi-asset options*, Comput. Math. Appl., 53 (2007), pp. 1159–1180.
[37] E. Miglio and C. Sgarra, *A finite element discretization method for option pricing with the Bates model*, SeMA J., 55 (2011), pp. 23–40.
[38] C. W. Oosterlee, *On multigrid for linear complementarity problems with application to American-style options*, Electron. Trans. Numer. Anal., 15 (2003), pp. 165–185.
[39] J. Persson and L. von Sydow, *Pricing European multi-asset options using a space-time adaptive FD-method*, Comput. Vis. Sci., 10 (2007), pp. 173–183.
[40] J. Persson and L. von Sydow, *Pricing American options using a space-time adaptive finite difference method*, Math. Comput. Simulation, 80 (2010), pp. 1922–1935.
[41] N. Rambeerich, D. Y. Tangman, M. R. Lollchund, and M. Bhuruth, *High-order computational methods for option valuation under multifactor models*, European J. Oper. Res., 224 (2013), pp. 219–226.
[42] R. Rannacher, *Finite element solution of diffusion problems with irregular data*, Numer. Math., 43 (1984), pp. 309–327.
[43] C. Reisinger and G. Wittum, *On multigrid for anisotropic equations and variational inequalities: Pricing multi-dimensional European and American options*, Comput. Vis. Sci., 7 (2004), pp. 189–197.

[44] J. W. Ruge and K. Stüben, *Algebraic multigrid*, in Multigrid Methods, Frontiers Appl. Math 3, SIAM, Philadelphia, 1987, pp. 73–130.
[45] S. Salmi and J. Toivanen, *An iterative method for pricing American options under jump-diffusion models*, Appl. Numer. Math., 61 (2011), pp. 821–831.
[46] S. Salmi and J. Toivanen, *Comparison and survey of finite difference methods for pricing American options under finite activity jump-diffusion models*, Int. J. Comput. Math., 89 (2012), pp. 1112–1134.
[47] S. Salmi and J. Toivanen, *IMEX schemes for pricing options under jump-diffusion models*, Appl. Numer. Math., 84 (2014), pp. 33–45.
[48] S. Salmi, J. Toivanen, and L. von Sydow, *Iterative methods for pricing American options under the Bates model*, in Proceedings of the 2013 International Conference on Computational Science, Procedia Computer Science Series 18, Elsevier, Amsterdam, 2013, pp. 1136–1144.
[49] K. Stüben, *Algebraic Multigrid: An Introduction with Applications*, in Multigrid, Academic Press Inc., San Diego, 2001.
[50] D. Tavella and C. Randall, *Pricing Financial Instruments: The Finite Difference Method*, John Wiley & Sons, Chichester, 2000.
[51] J. Toivanen, *A componentwise splitting method for pricing American options under the Bates model*, in Applied and Numerical Partial Differential Equations, Comput. Methods Appl. Sci. 15, Springer, New York, 2010, pp. 213–227.
[52] J. Toivanen and C. W. Oosterlee, *A projected algebraic multigrid method for linear complementarity problems*, Numer. Math. Theory Methods Appl., 5 (2012), pp. 85–98.
[53] Y.-Y. Zhang, H.-K. Pang, L. Feng, and X.-Q. Jin, *Quadratic finite element and preconditioning for options pricing in the SVCJ model*, J. Comput. Finance, 17 (2014), pp. 3–30.
[54] R. Zvan, P. A. Forsyth, and K. R. Vetzal, *Penalty methods for American options with stochastic volatility*, J. Comput. Appl. Math., 91 (1998), pp. 199–218.