Fast Fourier Transforms in IMEX-schemes to price options under Bates model

Chi Zhang
1. Introduction

In the financial market, an option is a contract giving the buyer the right but not the obligation to buy or sell an asset, $S$, at the strike price, $K$, at a maturity date $T$. Two common option types are American and European options.

The most commonly used model is the standard Black-Scholes model, which was introduced by Black-Scholes in [1] and Merton in [2]

$$dS_t = \mu S_t \, dt + \sigma S_t \, dW_t,$$  

(1)

where $S$ is the price of underlying asset, $\mu$ is the drift rate of $S$ and $\sigma$ is the volatility of $S$.

Later, a model was developed by Merton [3] who added log-normally distributed jumps in the asset and by Heston [4] who allowed for stochastic variations in the volatility.

In this paper, we mainly focus on the pricing of European and American options under Bates model [5], which combines Heston’s stochastic volatility model with the Merton’s jump model. Thus this model allows jumps in the underlying asset $S$ and stochastic variance $V$, see Equation (2)

$$dS_t = (r - q - \lambda \xi)S_t \, dt + \sqrt{V_t} S_t \, dW_t^1 + (J - 1)S_t \, dN,$$  

(2)

$$dV_t = \kappa(\theta - V_t) \, dt + \sigma \sqrt{V_t} \, dW_t^2,$$  

where $r$ is the risk free interest rate, $q$ is the dividend yield, $\lambda$ is the intensity of the Poisson process $N$, $W^1$ and $W^2$ are Wiener processes with correlation $\rho$, $\theta$ is the mean level of the jump, and $\kappa$ is the rate of reversion to the mean level of the variance $V$.

The jump size $J$ is a Poisson process with a log-normal distribution

$$f(J) = \frac{1}{\sqrt{2\pi} \delta} \exp \left( - \frac{[ln(J) - (\gamma - \delta^2/2)]^2}{2\delta^2} \right),$$  

(3)

where $\gamma - \delta^2/2$ is the mean of $\log J$, $\delta^2$ is the variance of $\log J$, and $\xi$ is defined by $\xi = e^\gamma - 1$. Let $\tau = T - t$, and we formulate the partial integro-differential equation in forward time
\[
\frac{\partial u(s,v,\tau)}{\partial \tau} = \frac{1}{2} \nu s^2 \frac{\partial^2 u(s,v,\tau)}{\partial s^2} + \frac{1}{2} \nu \sigma^2 \frac{\partial^2 u(s,v,\tau)}{\partial v^2} + \rho \nu s \frac{\partial^2 u(s,v,\tau)}{\partial s \partial v} \\
+ (r - q - \lambda \xi) \frac{\partial u(s,v,\tau)}{\partial s} + \kappa (\theta - v) \frac{\partial u(s,v,\tau)}{\partial v} \\
-(r + \lambda)u(s,v,\tau) + \lambda \int_0^\infty u(js,v,\tau)f(j) dj,
\]

where \( u \) is the price of a European option, \( s \) is the value of the underlying asset and \( v \) is the volatility. The initial condition that is used for (4) is

\[
u(s,v,0) = \Phi(s,v) = max(s - K, 0),
\]

where \( \Phi(s,v) \) is the pay-off function. The boundary conditions are

\[
\begin{aligned}
u(0,v,\tau) &= 0, \\
u(S_{\text{max}},v,\tau) &= S_{\text{max}} e^{(-q \tau)} - Ke^{(-r \tau)}, \\
\frac{\partial u(s,V_{\text{max}},\tau)}{\partial v} &= 0,
\end{aligned}
\]

where \( S_{\text{max}} \) is the maximum value of the underlying asset, and \( V_{\text{max}} \) is the maximum value of the variance.

Numerical pricing of options under Bates model are for instance considered in [6], [7], [8], [9], [10], and [11].

The work in this thesis proceeds from [12] and [13]. In [12] the author used the backward differentiation formula BDF-2 for (4), but this is time-consuming since he has to solve linear systems with full matrices for all time-steps. Then in [13] an IMEX-scheme was used where the full matrices end up in the right-hand side in the time-stepping scheme. In this paper we will take advantage of the Fast Fourier Transform to accelerate the whole pricing process for the IMEX-scheme for both European and American options.

2. Discretization

2.1 Spatial discretization

The detailed discretization of the problem can be found in [12]. As the problem is solved on a two dimensional grid in \( s \) and \( v \), we define \( u \) to be the lexicographical
ordering of the approximation solution of (4)

\[ u = (u_{1,1} \ u_{1,2} \ ... \ u_{1,n} \ ... \ u_{j,1} \ ... \ u_{m,1} \ ... \ u_{m,n})^T. \]

All derivatives in (4) are estimated by second order finite differences

\[
\begin{align*}
    u_{ss} &\approx \frac{u_{i+1,j} - 2u_{ij} + u_{i-1,j}}{h_s^2}, \\
    u_{vv} &\approx \frac{u_{i,j+1} - 2u_{ij} + u_{i,j-1}}{h_v^2}, \\
    u_{sv} &\approx \frac{u_{i+1,j+1} - u_{i+1,j-1} + u_{i-1,j+1} + u_{i-1,j-1}}{4h_s h_v},
\end{align*}
\]

When solving the problem a second time with adaptive step lengths in space, derivative approximations of \( u \) change. This is shown in details in [14] and [15].

The discrete approximation of the integral term in (4) is evaluated by first making a transformation from the original computational grid \( s_i \) to an equidistant grid \( x_i \). This is because we will make use of the Fast Fourier Transforms later. Let

\[
\begin{align*}
    I &= \int_{-\infty}^{\infty} u(s, v, \tau) f(J) dJ = \int_{-\infty}^{\infty} \tilde{u}(x + z, v, \tau) \tilde{f}(z) dz, 
\end{align*}
\]

where \( x = \log s, z = \log J, \tilde{u}(z, v, \tau) = u(e^z, v, \tau) \) and \( \tilde{f}(z) = e^zf(e^z) \). Then we create new variables \( \zeta = z + x \) and we get the estimation of the integral term at \( x_i \)

\[
\begin{align*}
    I_i &= \int_{-\infty}^{\infty} \tilde{u}(\zeta, v, \tau) \tilde{f}(\zeta - x_i) d\zeta = \int_{x_{\text{min}}}^{x_{\text{max}}} \tilde{u}(\zeta, v, \tau) \tilde{f}(\zeta - x_i) d\zeta \\
    &+ \int_{-\infty}^{x_{\text{min}}} \tilde{u}(\zeta, v, \tau) \tilde{f}(\zeta - x_i) d\zeta + \int_{x_{\text{max}}}^{\infty} \tilde{u}(\zeta, v, \tau) \tilde{f}(\zeta - x_i) d\zeta.
\end{align*}
\]

The first part of (9) is evaluated using the trapezoidal quadrature rule on an equidistant grid in \( x \) with spacing \( \Delta x \) and \( m_x \) grid-points in the open interval of \( (x_{\text{min}}, x_{\text{max}}) \) giving
\[ \int_{x_{\min}}^{x_{\max}} \bar{u}(\zeta, v, \tau) \bar{f}(\zeta - x_i) d\zeta \approx \Delta x \sum_{j=1}^{m_X} \bar{u}(\zeta_j, v, \tau) \bar{f}(\zeta_j - x_i) \]

\[ + \frac{\Delta x}{2} \bar{u}(x_{\min}, v, \tau) \bar{f}(x_{\min} - x_i) + \frac{\Delta x}{2} \bar{u}(x_{\max}, v, \tau) \bar{f}(x_{\max} - x_i). \]  

The second part of (9) can be evaluated by

\[ I_{i}^{(2)} = \int_{-\infty}^{x_{\min}} \bar{u}(\zeta, v, \tau) \bar{f}(\zeta - x_i) d\zeta \approx \bar{u}(x_{\min}, v, \tau) \int_{-\infty}^{x_{\min}} \bar{f}(\zeta - x_i) d\zeta \]

\[ = \frac{1}{\sqrt{2\pi} \delta} \bar{u}(x_{\min}, v, \tau) \int_{-\infty}^{x_{\min}} e^{-\frac{(x - y^2/2)^2}{2\delta^2}} ds \]

\[ = \bar{u}(x_{\min}, v, \tau) \frac{1}{\sqrt{\pi}} \int_{-\frac{1}{\sqrt{2}}}^{\infty} e^{-z^2} dz \]

\[ = \frac{1}{2} \bar{u}(x_{\min}, v, \tau) \text{erfc}(-z), \]

where \text{erfc}(\cdot) is the complementary error function and \( z = -\frac{(x_{\min} - (y - \delta^2/2))^2}{\sqrt{2}\delta}. \)

Finally the third part in (9) still needs to be evaluated and we treat it as a vector of boundary conditions. To solve this part we use the approximations

\[ \text{erf}(0 - (y - \delta^2/2)) \approx -1 \text{ and } \text{erf}(0 + (y + \delta^2/2)) \approx 1. \]

\[ I_{i}^{(3)} = \int_{x_{\max}}^{\infty} \bar{u}(\zeta, v, \tau) \bar{f}(\zeta - x_i) d\zeta = \int_{x_{\max}}^{\infty} \Phi(e^{\zeta + x}, v, \tau) \bar{f}(z) dz \]

\[ = \frac{1}{\sqrt{2\pi} \delta} \int_{x_{\max}}^{\infty} \max(e^{\zeta - K e^{-r \tau}}) \bar{f}(z) dz \]  

\[ = \frac{1}{2} s_i e^y [1 + \text{erf}(\frac{y - \delta^2}{\sqrt{2}\delta})] + \frac{1}{2} K e^{-r \tau} [-1 + \text{erf}(\frac{x_{\max} - (y - \delta^2/2)}{\sqrt{2}\delta})]. \]

### 2.2 Time discretization

The discretization of derivatives gives a three-diagonal matrix \( A \) in the equation of (4) while the integral gives a block diagonal matrix \( J \). Due to the different structure of these matrices, we treat \( A \) and \( J \) by differently in the time-stepping scheme.

For the first time step, Euler backwards is used.
\[ u^1 = \Delta t (A + f) u^1 + u^0, \]  

(13)

where \( \Delta t \) is the discretization parameter in time.

For the remaining time-steps, the Bates model is solved with an IMEX-scheme. Specifically, we treat the matrix \( A \) implicitly whereas we treat the matrix \( J \) explicitly in the Crank-Nicholson, Adams-Bashforth scheme. The advantage of this IMEX-scheme is that it is faster than the fully implicit method and more stable than the pure explicit method.

\[ u^{k+1} \approx \Delta t A \frac{u^{k+1} + u^k}{2} + \Delta t J \frac{3u^k - u^{k-1}}{2} + u^k, \text{ for } k=2,\ldots,m-1. \]  

(14)

More details about the IMEX method can be found in [16].

3. Fast Fourier Transforms

In order to increase the efficiency of the multiplication by the \( J \) matrix and the vector \( u \), we will use the Fast Fourier Transforms (FFTs).

Let \( \bar{I}_i = \Delta x \sum_{j=1}^{m_x} u(\zeta_j, v, \tau) \hat{f}(\zeta_j - x_i) \) in (9) and all \( \bar{I}_i, i=1,2,\ldots m_x \) can be computed by

\[ \bar{I} = T_{m_x} \bar{u}, \]  

(15)

where \( \bar{I} = (\bar{I}_1 \bar{I}_2 \ldots \bar{I}_{m_x})^T \), \( \bar{u} = (\bar{u}_1 \bar{u}_2 \ldots \bar{u}_{m_x})^T \), and \( T_{m_x} \) is the Toeplitz matrix

\[
T_{m_x} = \begin{pmatrix}
\hat{f}(0) & \hat{f}(\Delta x) & \ldots & \hat{f}((m_x - 1)\Delta x) \\
\hat{f}(-\Delta x) & \hat{f}(0) & \ldots & \hat{f}((m_x - 2)\Delta x) \\
\vdots & \vdots & \ddots & \vdots \\
\hat{f}(-(m_x - 1)\Delta x) & \hat{f}(-(m_x - 2)\Delta x) & \ldots & \hat{f}(0)
\end{pmatrix}.
\]

As the constant diagonal Toeplitz matrix \( T_{m_x} \) can be embedded into a circulant matrix \( C_{2m_x-1} \) with the first row \( c \)

\[
c = (\hat{f}(0) \hat{f}(\Delta x) \ldots \hat{f}((m_x - 1)\Delta x) 0 0 \ldots 0 \hat{f}(-(m_x - 1)\Delta x) \ldots \hat{f}(-\Delta x)).
\]  

(16)

we can first compute \( \bar{I} = C_{2m_x-1} \bar{u} \), where \( \bar{u} = (\bar{u}_1 \bar{u}_2 \ldots \bar{u}_{m_x} 0 \ldots 0)^T \). \( \bar{I} \) can then be achieved by the first \( m_x \) elements in \( \bar{I} \).

To make use of the FFTs, we must first decompose the matrix \( C_{2m_x-1} \) as

\[
C_{2m_x-1} = F_{2m_x-1}^{-1} A F_{2m_x-1},
\]  

(17)

where \( F_{2m_x-1} \) is a Fourier matrix of order \( 2m_x - 1 \) and \( A \) is a diagonal matrix.
with the eigenvalues of \( C_{2m_x-1} \) on the diagonal. Then \( \tilde{I} \) is given by
\[
\tilde{I} = F_{2m_x-1}^{-1} A F_{2m_x-1} \tilde{u},
\]
which could be accomplished by using two fast Fourier transforms (FFTs) and one inverse fast Fourier transforms (IFFTs). To get an accurate estimation, we choose to embed \( T_{m_x} \) in a circulant matrix \( C_{M_x} \) where \( M_x \) is the smallest power of 2 such that \( M_x \geq 2m_x - 1 \) and the number of grid points in \( x \) is twice as many as those in \( s \). The eigenvalues of \( C_{M_x} \) can be computed once in the beginning to improve the efficiency.

Generally, the computation of the matrix-vector multiplication in (15) follows the algorithm below:

- Interpolate values to new equidistant points \( x_i \) from the \( s_i \) grid-points in Section 2.
- Compute \( T_{m_x} \) and embed it into a circulant matrix \( C_{M_x} \).
- Use the FFTs above to compute \( \tilde{I} \).
- Get \( \tilde{I} \) by taking the first \( m_x \) elements in \( \tilde{I} \).
- Interpolate values of \( \tilde{I} \) back to the original grid-points \( s_i \).

4. Adaptive method

The purpose of making use of the adaptive method is to place grid-points where they are most needed for accuracy reasons, [14], [15]. First of all, we need to solve the problem on a sparse equidistant grid. Then a new, adaptive grid can be constructed where grid points could be distributed more efficiently. In this paper, we consider adaptivity in \( s \) and \( v \).

The adaptive process in \( s \) works like this: \( (v \) is treated analogously): Let \( B \) be the “exact” discrete representation of the right hand side of equation (4). For any smooth solution \( u \), it holds that
\[
B_{h_s h_v} u_{h_s h_v} = Bu + \tau_{h_s h_v},
\]
where \( B_{h_s h_v} u_{h_s h_v} \) is the discrete approximation in space of (4), \( u_{h_s h_v} \) is the vector of the numerical solution, and \( \tau_{h_s h_v} \) is the truncation error of the approximation with step-lengths \( h_s \) and \( h_v \). Use the approximation
\[
\tau_{h_s h_v} \approx h_s^2 \eta_s(s, v) + h_v^2 \eta_v(s, v) = \tau_{h_s} + \tau_{h_v},
\]
and consider first
\[ \tau_{h_s} \approx h_s^2 \eta_s(s, v). \]  \hspace{1cm} (20)

Define
\[ \delta_{h_s} = Bu + \tau_{h_s}, \] \hspace{1cm} (21)
\[ \delta_{2h_s} = Bu + \tau_{2h_s}. \]

The local discretization error of the approximation of order \( p \) on the grid with step size \( h_s \) can be approximated in the coarse grid with the step size \( 2h_s \) by using (19) and (20)

\[ \tau_{h_s} = \frac{1}{3} (\delta_{2h_s} - \delta_{h_s}). \] \hspace{1cm} (22)

The local truncation error \( \tau_{h_s} \) is computed in several points in time. The maximum absolute value of \( \tau_{h_s} \) is chosen in each spatial point to calculate the new grid.

In order to get an approximation for \( \eta_s(s, v) \), we compute a solution with a step length \( \hat{h} \). Then from equation (20), we get

\[ \eta_s(s, v) = \frac{\tau_{h_s}(s, v)}{h_s^2}. \] \hspace{1cm} (23)

To control the local discretization error such that \( \tau_{h_s}(s) \leq \epsilon_s \) for any \( \epsilon_s > 0 \), we obtain the following by combining (20), (21) and (22)

\[ |\tau_{h_s}(s)| = |h_s^2(s)\eta_s(s, v)| \approx |h_s^2(s)\frac{\tau_{h_s}(s, v)}{h_s^2(s)}| \leq \epsilon_s. \] \hspace{1cm} (24)

Using Equation (25)
\[ \tau_{\hat{h}_s}(s) = \max_v |\tau_{\hat{h}_s}(s, v)|, \] \hspace{1cm} (25)
we could obtain

\[ h_s(s) = \hat{h}_s(s)\left(\frac{\epsilon_s}{\epsilon_s \gamma_s + \tau_{\hat{h}_s}(s)}\right)^{1/2}. \] \hspace{1cm} (26)

To prevent the algorithm from taking too big space-steps in \( s \) when the truncation error is small, we introduce a parameter \( \gamma_s \) to the equation (26)

\[ h_s(s) = \hat{h}_s(s)\left(\frac{\epsilon_s}{\epsilon_s \gamma_s + \tau_{\hat{h}_s}(s)}\right)^{1/2}. \] \hspace{1cm} (27)

Finally, a simple linear interpolation could be used to get \( h_s(s) \) between known function values.
5. American Option Pricing

In this paper, we use the operator splitting method [17] for the pricing of American options. For simplicity, we introduce the operator

\[ \mathcal{L}u = \frac{\partial u(s,v,\tau)}{\partial \tau} - \frac{1}{2} vs^2 \frac{\partial^2 u(s,v,\tau)}{\partial s^2} - \frac{1}{2} \nu \sigma^2 \frac{\partial^2 u(s,v,\tau)}{\partial v^2} - \rho \sigma v s \frac{\partial^2 u(s,v,\tau)}{\partial s \partial v} \]

\[ -(r - q - \lambda \xi) \frac{\partial u(s,v,\tau)}{\partial \xi} - \kappa (\theta - v) \frac{\partial u(s,v,\tau)}{\partial v} \]

\[ -\lambda \int_0^\infty u(Js,v,\tau)f(J)\,dJ + (r + \lambda)u(s,v,\tau). \]

As American options can be exercised earlier than the maturity date \( T \), we have to include the following constraint for the option price:

\[ u(s,v,\tau) \geq \Phi(s,v), \quad s \in (0,S_{max}), \quad v \in [0,V_{max}], \quad \tau \in [0,T]. \]  

(29)

This gives that the price of the American option based on the Bates model can be obtained by solving a time dependent linear complementarity problem

\[ \begin{cases} \mathcal{L}u \geq 0, \\ u \geq \Phi, \\ (u - \Phi)\mathcal{L}u = 0, \end{cases} \]  

(30)

for \( (s,v,\tau) \in (0,S_{max}) \times [0,V_{max}] \times [0,T] \) with the initial condition (5) and boundary conditions

\[ \begin{cases} u(0,v,\tau) = 0, \\ u(S_{max},v,\tau) = \max(S_{max}e^{(-q\tau)} - Ke^{(-r\tau)},S_{max} - K), \\ \frac{\partial u(s,V_{max},\tau)}{\partial v} = 0. \end{cases} \]  

(31)

We now present the operator splitting method for the linear complementarity problem (30) based on the following formulation with an auxiliary variable \( \lambda \):

\[ \begin{cases} \mathcal{L}u = \lambda, \\ \lambda \geq 0, u \geq \Phi, \\ (u - \Phi)\lambda = 0, \end{cases} \]  

(31)

with the initial condition (5) and boundary conditions (31).
By using the same space and time discretizations as in Section 2, the operator splitting method [17] can be applied based on two fractional time steps. In the first fractional step, we solve a system of linear equations. Then the solution and the variable \( \lambda \) are updated which satisfy the constraints for them in the second step. Since the operator splitting depends on the underlying time discretization, we describe the operator splitting methods for Euler backwards and the IMEX scheme.

For the first time step, the operator splitting method for the implicit Euler scheme reads

\[
\left( \frac{1}{\Delta t} I - (A + f) \right) \tilde{u}^{k+1} = \left( \frac{1}{\Delta t} I \right) u^k + \lambda^k, \tag{33}
\]

\[
\begin{align*}
\frac{1}{\Delta t} (u^{k+1} - \tilde{u}^{k+1}) - (\lambda^{k+1} - \lambda^k) &= 0, \\
(\lambda^{k+1})^T (u^{k+1} - g) &= 0, \quad u^{k+1} \geq g \text{ and } \lambda^{k+1} \geq 0,
\end{align*} \tag{34}
\]

for \( k=1 \). The solution \( \tilde{u}^{k+1} \) of the linear system of equations (33) is obtained and updates for \( \lambda^{k+1} \) and \( u^{k+1} \) by using Equations (34).

Then for the time step \( k=2,3,\ldots,m-1 \), the operator splitting for IMEX scheme reads

\[
\left( \frac{1}{\Delta t} I - \frac{1}{2} A \right) \tilde{u}^{k+1} = \left( \frac{1}{\Delta t} I + \frac{1}{2} A + \frac{3}{2} f \right) u^k - \frac{1}{2} f u^{k-1} + \lambda^k, \tag{35}
\]

\[
\begin{align*}
\frac{1}{\Delta t} (u^{k+1} - \tilde{u}^{k+1}) - (\lambda^{k+1} - \lambda^k) &= 0, \\
(\lambda^{k+1})^T (u^{k+1} - g) &= 0, \quad u^{k+1} \geq g \text{ and } \lambda^{k+1} \geq 0.
\end{align*} \tag{36}
\]

Again, the solution \( \tilde{u}^{k+1} \) is achieved by solving (35) and updates are performed using Equation (36).

6. Numerical results

In this section we present some computational results for European options and American options using equidistant and adaptive grids.

The implementation is done in MATLAB and the matrix \( A \) is allocated as a sparse matrix. The linear system of equations obtained from Euler backwards and the IMEX-scheme are solved by the iterative method Generalized Minimal Residual Method (GMRES), see [19]. We restart GMRES after six iterations for efficiency. When the residual is smaller than \( 10^{-8} \), we terminate the iteration process.
MATLAB’s built-in, sparse, incomplete LU factorization, ilu has been used as a preconditioner. To improve the efficiency, we only need to calculate L and U once for Euler backwards and once again for the IMEX-scheme.

Parameters are chosen as follows: risk free interest rate \( r = 0.03 \), the dividend yield \( q = 0.05 \), the strike price \( K = 100 \), the maturity date \( T = 0.5 \), the correlation between the two Wiener processes \( \rho = -0.5 \), the mean level of variance \( \theta = 0.04 \), the rate of reversion \( \kappa = 2.0 \), the volatility of the variance \( \sigma = 0.25 \), the intensity rate of the jump \( \lambda = 0.2 \), the parameters of mean and variance of the jump \( \gamma \) and \( \delta \) are \(-0.5\) and \(0.4\) respectively. We use the rule of thumb, \( S_{\text{max}} = 4K \), \( S_{\min} = 0 \), and \( V_{\text{max}} = 1 \).

When using the adaptive method, the same coarse grid has been used, regardless of the number of grid points in the adaptive grid, namely \((n,m,k)=(41,40,40)\). The number of grid points in space has varied in both directions while the number of time steps has been kept constant with \( k = 513 \). The ratio between \( n \) and \( m \) is \( n=2m \), see Table 6.1. The first and the last point in \( S \), i.e. \( S_{\text{max}} \) and \( S_{\min} \) are not included in the grid.

<table>
<thead>
<tr>
<th>Grid</th>
<th>Number of Nodes in s</th>
<th>Number of Nodes in v</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>48</td>
<td>24</td>
</tr>
<tr>
<td>2</td>
<td>64</td>
<td>32</td>
</tr>
<tr>
<td>3</td>
<td>96</td>
<td>48</td>
</tr>
<tr>
<td>4</td>
<td>128</td>
<td>64</td>
</tr>
<tr>
<td>5</td>
<td>192</td>
<td>96</td>
</tr>
<tr>
<td>6</td>
<td>256</td>
<td>128</td>
</tr>
</tbody>
</table>

**Table 6.1**

The grid points in \( s \) and \( v \)

In the adaptive method, \( \gamma_s \) that prevents the function \( h_s(s) \) from taking too big space-steps when using the adaptive method, is set to 0.01. To create the adaptive grid, a coarse grid is first generated and the problem is then solved with the coarse grid, as mentioned in Section 4. Using the same coarse grid, namely \((n,m,k)=(41,40,40)\), parameter \( \varepsilon_s \) and \( \varepsilon_v \) are adjusted in order to get the desired number of grid points in the adaptive grid, see Table 6.2.
When applying the adaptive method, \( u_{ss} \) in \( \tau_s = h^2 \ast u_{ss} \) is supposed to be smooth. But at the free boundary for the American option, this does not hold, which leads to estimations of the local truncation errors that are locally quite large, see Figure 6.1. To improve the efficiency of the adaptive method, for each \( v \), if the errors are too large at the free boundary, see Figure 6.1, we artificially set them to be the first error at the free boundary smaller than that at the strike, i.e. make errors at position 2 and position 3 equal to the error at position 1 in Figure 6.2. We also ignore the local truncation errors at the corner and boundary of \( S_{max}, V_{max} \) for both European and American option due to the effects from the boundary conditions used there.

**FIG 6.1** An example of 2-D local truncation errors for the American option

**FIG 6.2** An example of 1-D local truncation errors for the American option for \( v=0.8718 \)
In the FFTs method, $x_{max} = log(S_{max})$, $x_{min} = log(100/1024)$. To improve the accuracy of FFTs, the number of grid points in $x$ is approximately twice as many as those in $s$, see Section 3.

The error is measured in the $l_2$-norm and maximum-norm with respect to the solution on a fine, equidistant grid with 512 points in the $s$-dimension, 256 points in the $v$-dimension and 513 points in time. This solution can be regarded as a good estimation of $u(s,v,T)$. Errors are only measured in the domain below because this is where we consider it most likely that values of the stock and the volatility are located, i.e. this is where we need to have an accurate solution.

$$\Omega_K = \left\{ s \mid \frac{1}{3} K \leq s \leq \frac{5}{3} K, 0 < v \leq 0.5 \right\}. \quad (39)$$

Reference values for the European option come from a Monte-Carlo method from [18], and reference values for the American options are from Jari Toivanen’s experiments in the paper [8], see Table 6.3. For $s=80$, the reference price of European option is larger than that of American option. This is most likely due to the fact that we are using different methods to compute these prices.

<table>
<thead>
<tr>
<th>Grid</th>
<th>European Call</th>
<th>American Call</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>(0.087, 0.045)</td>
<td>(0.135, 0.032)</td>
</tr>
<tr>
<td>2</td>
<td>(0.048, 0.023)</td>
<td>(0.073, 0.017)</td>
</tr>
<tr>
<td>3</td>
<td>(0.021, 0.01)</td>
<td>(0.0315, 0.007)</td>
</tr>
<tr>
<td>4</td>
<td>(0.0116, 0.0054)</td>
<td>(0.0175, 0.0038)</td>
</tr>
<tr>
<td>5</td>
<td>(0.0051, 0.00235)</td>
<td>(0.0077, 0.00165)</td>
</tr>
<tr>
<td>6</td>
<td>(0.00285, 0.0013)</td>
<td>(0.00432, 0.0009)</td>
</tr>
</tbody>
</table>

**Table 6.2**

_Tolerance for adaptive method_
Table 6.3

Reference values at \( S_0 = \{80, 90, 100, 110, 120\} \)

Table 6.4 and Table 6.5 present absolute errors for the European call option and the American call option under Bates model respectively using equidistant and adaptive grids compared to reference values in Table 6.3. The computed values seem to converge to the reference values.

<table>
<thead>
<tr>
<th>Method</th>
<th>Gri d</th>
<th>( S_0=80 )</th>
<th>( S_0=90 )</th>
<th>( S_0=100 )</th>
<th>( S_0=110 )</th>
<th>( S_0=120 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>Adaptive</td>
<td>1</td>
<td>0.01199</td>
<td>0.03682</td>
<td>0.04254</td>
<td>0.02228</td>
<td>0.00916</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>0.00620</td>
<td>0.02022</td>
<td>0.02505</td>
<td>0.01369</td>
<td>0.00506</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>0.00439</td>
<td>0.00374</td>
<td>0.00613</td>
<td>0.00276</td>
<td>0.00041</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>0.00160</td>
<td>0.00382</td>
<td>0.00504</td>
<td>0.00235</td>
<td>0.00056</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>0.00070</td>
<td>0.00140</td>
<td>0.00171</td>
<td>0.00054</td>
<td>0.00011</td>
</tr>
<tr>
<td></td>
<td>6</td>
<td>0.00027</td>
<td>0.00086</td>
<td>0.00083</td>
<td>0.00009</td>
<td>0.00025</td>
</tr>
<tr>
<td>Equidistant</td>
<td>1</td>
<td>0.08869</td>
<td>0.01225</td>
<td>0.13525</td>
<td>0.01342</td>
<td>0.02640</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>0.05402</td>
<td>0.02283</td>
<td>0.04627</td>
<td>0.00389</td>
<td>0.00644</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>0.02588</td>
<td>0.01100</td>
<td>0.01502</td>
<td>0.00219</td>
<td>0.00211</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>0.01484</td>
<td>0.00462</td>
<td>0.01040</td>
<td>0.00205</td>
<td>0.00191</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>0.00657</td>
<td>0.00191</td>
<td>0.00396</td>
<td>0.00084</td>
<td>0.00099</td>
</tr>
<tr>
<td></td>
<td>6</td>
<td>0.00358</td>
<td>0.00105</td>
<td>0.00248</td>
<td>0.00042</td>
<td>0.00065</td>
</tr>
</tbody>
</table>

Table 6.4

Absolute errors at \( S_0 = \{80, 90, 100, 110, 120\} \) for the European call option under Bates model
TABLE 6.5

<table>
<thead>
<tr>
<th>Method</th>
<th>Grid</th>
<th>S0=80</th>
<th>S0=90</th>
<th>S0=100</th>
<th>S0=110</th>
<th>S0=120</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>0.03500</td>
<td>0.01001</td>
<td>0.01866</td>
<td>0.00641</td>
<td>0.00187</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>0.01955</td>
<td>0.00477</td>
<td>0.00759</td>
<td>0.00424</td>
<td>0.00385</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>0.00865</td>
<td>0.00276</td>
<td>0.00268</td>
<td>0.00233</td>
<td>0.00495</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>0.00393</td>
<td>0.00076</td>
<td>0.00385</td>
<td>0.00370</td>
<td>0.00646</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>0.00186</td>
<td>0.00006</td>
<td>0.00149</td>
<td>0.00269</td>
<td>0.00647</td>
</tr>
<tr>
<td></td>
<td>6</td>
<td>0.00088</td>
<td>0.00037</td>
<td>0.00117</td>
<td>0.00267</td>
<td>0.00665</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>0.08893</td>
<td>0.01202</td>
<td>0.13545</td>
<td>0.01153</td>
<td>0.02163</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>0.05426</td>
<td>0.02257</td>
<td>0.04644</td>
<td>0.00178</td>
<td>0.00084</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>0.02610</td>
<td>0.01085</td>
<td>0.01538</td>
<td>0.00469</td>
<td>0.00433</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>0.01492</td>
<td>0.00488</td>
<td>0.01133</td>
<td>0.00513</td>
<td>0.00519</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>0.00687</td>
<td>0.00149</td>
<td>0.00396</td>
<td>0.00316</td>
<td>0.00574</td>
</tr>
<tr>
<td></td>
<td>6</td>
<td>0.00384</td>
<td>0.00074</td>
<td>0.00264</td>
<td>0.00288</td>
<td>0.00624</td>
</tr>
</tbody>
</table>

**Table 6.5**

*Absolute errors at $S_0 = \{80, 90, 100, 110, 120\}$ for the American call option under Bates model*

Figure 6.3 shows the log-log plot of the $l2$-norm errors and maximum-norm errors against number of grid points for the European call option and the American call option in both equidistant grid and adaptive grid. All slopes are around -1 which corresponds to the optimal convergence rate, since the x-axis in all four cases is the total number of grid points, i.e. (the number of grid points in s) $\times$ (the number of grid points in v).
In Figure 6.4, the CPU time is plotted as a function of $l_2$-norm errors and maximum-norm errors respectively for the European call option and the American call option in both equidistant grid and adaptive grid. It is obvious that the adaptive method gives more accurate estimations under a certain CPU time.
FIG 6.4 CPU times against absolute errors in equidistant grid and adaptive grid:
l2-norm errors in European call options (Top Left), maximum-norm errors in European call options 
(Top Right), l2-norm errors in American call options (Bottom Left), maximum-norm errors in American call options (Bottom Right)

In Figure 6.5 we compared methods with and without using FFTs in both equidistant and adaptive grid. When the number of grid points is small, the method without FFTs, \textit{i.e.} computing the integral term in \( s \), is more efficient because the number of grid points is larger in \( x \) and interpolation processes related to the FFTs take up a large proportion of the whole CPU time. But when the number of grid points is large enough, the method with FFTs is equally fast or faster than that without. From the plot we notice that the slope of the method with FFTs is smaller, which indicates that the CPU time increases more slowly, and we can expect a larger gain using FFTs for even larger problems.
7. Conclusions

In this thesis we used adaptive finite differences and an IMEX time-stepping scheme to price European and American options under Bates model. It is clear from the numerical results that the adaptive method generates a more accurate solution than the equidistant one by using the same number of grid-points. Both methods are 2nd order accurate for European options and American options.

Giving a certain error, the adaptive method is more efficient than the equidistant method since for a certain size of the discretization error, a smaller system of equations needs to be solved when considering the adaptive method and therefore the
CPU time is reduced.

Using FFTs in the computation of the integrals is more efficient than standard matrix-vector multiplication for both European and American option pricing if we solve large problems.
References


[18] J. Toivanen, Private communication, KME, Stanford University, e-mail: Jari.Toivanen@mit.jyu.