A Numerical Study of Radial Basis Function Based Methods for Options Pricing under the One Dimension Jump-diffusion Model

Ron T.L. Chan\textsuperscript{a,*}, S. Hubbert\textsuperscript{b}

\textsuperscript{a}Department of Economics, Mathematics and Statistics, Birkbeck, University of London, Malet Street, London WC1E 7HX.
\textsuperscript{b}Department of Economics, Mathematics and Statistics, Birkbeck, University of London, Malet Street, London WC1E 7HX.

Abstract

The aim of this chapter is to show how option prices in jump-diffusion models can be computed using meshless methods based on Radial Basis Function (RBF) interpolation. The RBF technique is demonstrated by solving the partial integro-differential equation (PIDE) in one-dimension for the American put and the European vanilla call/put options on dividend-paying stocks in the Merton and Kou jump-diffusion models. The radial basis function we select is the Cubic Spline. We also propose a simple numerical algorithm for finding a finite computational range of an improper integral term in the PIDE so that the accuracy of approximation of the integral can be improved. Moreover, the solution functions of the PIDE are approximated explicitly by RBFs which have exact forms so we can easily compute the global integral by any kind of numerical quadrature. Finally, we will not only show numerically that our scheme is second order accurate in both spatial and time variables in a European case but also second order accurate in spatial variables and first order accurate in time variables in an American case.

Keywords: Lévy Processes, the Jump-diffusion model, Partial-Integro Differential Equation, Radial Basis Function, Cubic Spline, European Option, American Option.

\textsuperscript{*}Corresponding author

Email addresses: rchan@ems.bbk.ac.uk (Ron T.L. Chan), s.hubbert@bbk.ac.uk (S. Hubbert)
1. Introduction

In this paper we show how to compute European and American option prices in the Jump-diffusion model using Radial Basis Function (RBF) interpolation techniques. RBF methods have recently been proposed for numerically solving initial value and free boundary problems for the classical Black and Scholes equation, both in the one and in the multiple asset case (Faussauer et al., 2004a,b; Hon and Mao, 1999; Larsson et al., 2008). The new feature of the present paper is that in the Jump-diffusion model, as in general Lévy type models, the Black and Scholes PDE is replaced by a Partial Integro-Differential Operator or PIDE, involving a global term in the form of an integral operator. The PIDE has a form:

\[
\partial_{\tau} u(x, \tau) = \frac{1}{2} \sigma^2 \partial_x^2 u + \left( r - q - \frac{1}{2} \sigma^2 - \eta \right) \partial_x u - (r + \lambda) u + \lambda \int_{\mathbb{R}} u(x + y, \tau) f(y) dy
\]

(cf. Cont and Tankov, 2004; Schoutens, 2003, 2006). Our main contribution is to show how to numerically solve (1) in an efficient way using RBFs, both for initial value and free boundary problems (as for American options). We have chosen the Jump-diffusion model as a typical case on which to test the present RBF methodology. Our method extends however without problems to other contexts in which the basic pricing equation is a PIDE, like that of Lévy-type models such as Carr-Geman-Madan-Yor (CGGMY) (Carr et al., 2002) or Variance Gamma (VG) (Carr et al., 1998; Madan and Milne, 1991). These will be treated in a future paper.

Currently, PIDEs such as the Merton Model (Merton, 1976) and the Kou Model (Kou, 2002; Kou and Wang, 2001), one have mostly been treated by a traditional Finite Difference Method (FDM) or Finite Elements Method (FEM). In FDM, the idea is to simply fully discretize the PIDE on an equidistant grid, after having (artificially) localized the equations to some bounded interval/domain in \( \mathbb{R} \). The global integral term can be computed by numerical quadrature or by using the Fast Fourier Transform (FFT) (see, Almendral, 2005; Almendral and Oosterlee, 2004, 2005a, 2006, 2007; Andersen and Andreasen, 2000; Briani et al., 2007; Cont and Voltchkova, 2005; d’Halluin et al., 2004, 2005; Hirsa and Madan, 2004; Wang et al., 2007). By contrast, FEM is defined as piecewise polynomial functions or wavelet functions on regular triangularizations. This technique is used to approximate
solutions of the partial differential terms as well as of the integral term (cf. Almendral and Oosterlee, 2005b; Matache et al., 2003, 2005).

In general, there are a number of problems which arise with these current approaches. First, some of the literature, (e.g., Briani et al., 2007; Cont and Voltchkova, 2005), plays down the importance of pricing American and European vanilla option values when time to maturity is less than 3 months. The reason is that for short times-to-maturity the numerical methods used to price the option tend to be inaccurate near the strike price where a singularity (kink) exists. A singularity is defined as a point at which the function, or its derivative, is discontinuous. The payoff functions of vanilla call and put options have such a singularity. As a result, standard numerical methods such as FDM cannot ensure accuracy and suffer a reduced rate of convergence when used to price options at a very short time to maturity. This can be explained by Forysth and Vetzal’s heuristic findings in (Forsyth and Vetzal, 2002) when they solve an American option under a Brownian case. By their heuristic analysis of the behavior of the solution near the exercise boundary in an American option, Forysth and Vetzal finds that the convergence rate of the solution appears only at the rate less than second order convergence when a standard FDM with implicit method and constant timesteps has been applied. Forysth et al. shed light on this kind of problem (d’Halluin et al., 2005) by suggesting Rannacher’s time stepping method (Rannacher, 1984). This is a mixture of implicit and Crank-Nicolson methods. They demonstrate this technique by approximating an option price whose maturity is a quarter of a year. This method gives second order rates of convergence when pricing European options but not American ones. By using the same idea and combining it with a penalty method and a modified form of a timestep selector suggested in (Johnson, 1987), Forysth et al. (d’Halluin et al., 2004) show how to achieve second order convergence for pricing American options. Although their methods can yield second order convergence, the necessary calculations can be quite complex. Moreover, the papers (Briani et al., 2007; Cont and Voltchkova, 2005; d’Halluin et al., 2005, 2004) implement an implicit-explicit numerical scheme to price European or American options under the Lévy model. These papers treat the convection (hyperbolic) term $\partial_x u(x, \tau)$ of (1) explicitly by implementing the upwind scheme and the diffusion (elliptic) term $\partial_{xx} u(x, \tau)$ of (1) implicitly. As a result, restrictive stability conditions are necessary for the convection term when the upwind scheme is implemented. A final but fundamental problem with FDM and FEM is that these are, in practice, restricted to
problems of two or three space dimensions; however, most applications easily require many more, e.g. when pricing basket options.

Our RBF-method will circumvent many of these disadvantages. This paper is divided into five sections, including this introduction. Section 2 is a brief review of both the Merton and Kou Jump-diffusion models. In section 3 we first explain and then define our RBF algorithm for solving PIDEs, which we implement the Jump-diffusion model. Section 4 contains our numerical results for both European and American call and put options, including an analysis of the root-mean-square error and rate of convergence and also a comparison the accuracy of our solution with that of FDM and FEM. Section 5 concludes.

2. PIDE Option Pricing Formula in Jump-diffusion Market

In this short section we will focus on the Merton and the Kou Jump-diffusion Models which are a general Lévy processes consisting of Brownian motion and compound Possion jumps. By using this model we can describe the price dynamics of the underlying risky asset, \((S_t)_{t \geq 0}\). The evolution of \((S_t)_{t \geq 0}\) is driven by a diffusion process, punctuated by jumps which describe rare events such as crashes and/or drawdowns at random intervals. As a market model, it is an example of an incomplete market. We will skirt around the hedging issue by working directly with the risk-neutral probability measure \(Q\), as is customary. The stock price process, \((S_t)_{t \geq 0}\), is then given by

\[
S_t = S_0 e^{L_t}
\]

where \(S_0\) is the stock price at time zero and \(L_t\) is defined by:

\[
L_t := \gamma_c t + \sigma W_t + \sum_{i=1}^{N_t} Y_i,
\]

here, \(\gamma_c\) is a drift term, \(\sigma\) is a volatility, \(W_t\) is a Brownian motion, \(N_t\) is a Possion process with intensity \(\lambda\), \(Y_i\) is an i.i.d. sequence of random variables. The risk-neutral Lévy triplet of these two processes can be described as follows:

\[
(\gamma_c, \sigma, \nu)
\]

where

\[
\gamma_c = r - q - \frac{1}{2} \sigma^2 - \lambda \eta + \int_{\mathbb{R}} x \nu(dx),
\]

\[
\nu(dx) = \int_{\mathbb{R}} x \eta(dx)
\]
Here we focus on the case where the Lévy measure is associated to the pure-jump component and hence the Lévy measure \( \nu(dx) \) can be written as \( \lambda f(x) \, dx \), where the weight function \( f(x) \) can take two forms:

1. In the classical Merton model, for any \( i \in \{1, 2, \ldots\} \), \( Y_i \) are log-normally distributed variables with \( Y_i \sim N(\mu_J, \sigma_J^2) \) and as a result,
   \[
   f(x) := \frac{1}{\sqrt{2\pi\sigma_J}} e^{(x-\mu_J)^2/2\sigma_J^2}.
   \] (5)

2. In the Kou model,
   \[
   f(x) = p\alpha_1 e^{-\alpha_1 x} 1_{x \geq 0} + (1 - p)\alpha_2 e^{\alpha_2 x} 1_{x \leq 0}.
   \] (6)

**Remark 1.** In the Merton Jump-diffusion model, one should notice that \( Y_i \) is i.i.d so for each \( i \in \{1, 2, 3, \ldots\} \), \( Y_i \) has the same mean and variance. For the sake of simplicity, we use \( \mu_J \) and \( \sigma_J^2 \) to represent the mean and variance of each \( Y_i \) respectively.

Also in (4), \( \eta = \int_{\mathbb{R}} (e^x - 1) f(x) \, dx \) represents the expected relative price change due to a jump. Since we have defined the Lévy density function \( f(x) \) for both Jump-diffusion processes, \( \eta \) can be computed as:

1. In the Merton model,
   \[
   \eta = e^{\mu_J + \sigma_J^2/2} - 1.
   \] (7)

2. In the Kou model,
   \[
   \eta = \frac{p\alpha_1}{\alpha_1 - 1} + \frac{(1 - p)\alpha_2}{\alpha_2 + 1} - 1.
   \] (8)

This is found by integrating \( e^x \) over the real line by setting \( \alpha_1 > 1 \) and \( \alpha_2 > 0 \).

For the details of the computation of (7) and (8), we shall refer the reader to [Cont and Tankov, 2004; Boyarchenko and Levendorskiı, 2002].

The drift-term \( \gamma_c \) in (3) assumes that \( e^{-rt} S_t \) is a martingale with respect to the natural filtration. We let \( \tau = T - t \), the time-to-maturity, where \( T \) is the maturity of the financial option under consideration and we introduce \( x = \log S_t \), the underlying asset’s log-price. If \( u(x, \tau) \) denotes the values of some (American and European) contingent claim on \( S_t \) when \( \log S_t = x \) and
\( \tau = T - t \), then it is well-known, see for example, (Cont and Tankov, 2004) that \( u \) satisfies the following PIDE in the non-exercise region:

\[
\partial_{\tau}u(x, \tau) = \frac{1}{2} \sigma^2 \partial^2_x u + \left( r - q - \frac{1}{2} \sigma^2 - \eta \right) \partial_x u - (r + \lambda)u + \lambda \int_{\mathbb{R}} u(x + y, \tau)f(y)dy,
\]

\( =: \mathcal{L}[u](x, \tau). \) (9)

with initial value

\( u(x, 0) = g(x) := G(e^x) = \begin{cases} \max\{e^x - K, 0\}, & \text{call option} \\ \max\{K - e^x, 0\}, & \text{put option} \end{cases} \) (10)

For an American put, we have to take into account the possibility of early exercise (e.g., Cont and Tankov, 2004; Schoutens, 2003, 2006). As a result, the highest value of American option can be achieved by maximizing over all allowed exercise strategies:

\[
u(x, \tau) = \operatorname{ess\,sup}_{\tau^* \in \Gamma(t, T)} E^Q_t [e^{-r(\tau^* - t)}G(e^{x_{\tau^*}})] \] (11)

where \( \Gamma(t, T) \) denotes the set of non-anticipating exercise times \( \tau^* \), satisfying \( t \leq \tau^* \leq T \). To actually compute the \( u(x, \tau) \) of the American put, one can solve the following linear complementarity problem (Cont and Tankov, 2004; Schoutens, 2003, 2006):

\[
\partial_{\tau}u(\tau, x) - \mathcal{L}u(x, \tau) \geq 0, \text{ in } (0, T) \times \mathbb{R} \]

\[
\begin{align}
\partial_{\tau}u(\tau, x) - \mathcal{L}u(x, \tau) &\geq 0, \text{ a.e. in } (0, T) \times \mathbb{R} \quad (13) \\
(u(x, \tau) - G(e^x)) (\partial_{\tau}u(\tau, x) - \mathcal{L}u(x, \tau)) &= 0, \text{ in } (0, T) \times \mathbb{R} \quad (14) \\
u(x, 0) &= G(e^x), \end{align}
\]

Since we only deal with a jump-diffusion model with \( \sigma > 0 \) and finite jump intensity in this paper, we know that by Pham (Pham, 1997), the smooth pasting condition,

\[
\frac{\partial u(x_{\tau^*}, \tau^*)}{\partial x} = -1
\]

is valid at time of exercise \( \tau^* \). Therefore the value of an American put option is continuously differentiable with respect to the underlying on \((0, T) \times \mathbb{R}\); in particular the derivative is continuous across the exercise boundary.
3. Meshfree Numerical Approximation Method

Meshfree radial basis function (RBF) interpolation is a well-known technique for reconstructing an unknown function from scattered data. It has numerous applications in different fields, such as terrain modeling in geology, surface reconstruction in imaging, and the numerical solution of partial differential equations in applied mathematics. In particular, RBFs have recently been used to solve the PDEs of quantitative finance. A number of authors, including Faussauer et al. (Faussauer et al., 2004a,b), Larsson et al. (Larsson et al., 2008), Pettersson et al. (Pettersson et al., 2008) and Hon and Mao (Hon and Mao, 1999), have suggested RBFs as a tool for solving Black-Scholes equations for European as well as American options. This numerical scheme for the estimation of partial derivatives using RBFs was originally proposed by Kansa (Kansa, 1990a), resulting in a new method for solving partial differential equations (Kansa, 1990b). The aim here is to obtain a RBF approximation of the initial value or pay-off of the option. Once we are disposition of such an RBF-interpolant, we implement an RBF-scheme to solve the PIDE with this RBF-interpolant as initial value. The general idea of the proposed numerical scheme is to approximate the unknown function $u(x, \tau)$ by an RBF-interpolant using the interpolation points found for the initial value using the RBF-scheme, and derive a system of linear constant coefficient ODE by requiring that the PIDE (9) be satisfied in the chosen RBF-interpolation points. After picking interpolation points $x_j \in \mathbb{R}$, we approximate, for any fixed time-to-maturity $\tau$, the solution $u(x, \tau)$ in (9) by its RBF-interpolant:

$$u(x, \tau) \simeq \sum_{j=1}^{N} \rho_j(\tau) \phi(||x - x_j||_2) =: \tilde{u}(x, \tau), \quad (16)$$

Since the radial basis function does not depend on time, the time derivative of $\tilde{u}(x, \tau)$ in equation (16) is simply:

$$\frac{\partial \tilde{u}(x, \tau)}{\partial \tau} = \sum_{j=1}^{N} \frac{d\rho_j(\tau)}{d\tau} \phi(||x - x_j||), \quad (17)$$
Moreover, the first and second partial derivatives of \( \tilde{u}(x, \tau) \) with respect to \( x \) are

\[
\frac{\partial \tilde{u}(x, \tau)}{\partial x} = \sum_{j=1}^{N} \rho_j(\tau) \frac{\partial \phi(|x - x_j|)}{\partial x},
\]

(18)

\[
\frac{\partial^2 \tilde{u}(x, \tau)}{\partial x^2} = \sum_{j=1}^{N} \rho_j(\tau) \frac{\partial^2 \phi(|x - x_j|)}{\partial x^2},
\]

(19)

where for the particular case when \( \phi \) is the Cubic Spline,

\[
\frac{\partial \phi(|x - x_j|)}{\partial x} = \begin{cases} 
3(|x - x_j|)^2 & \text{if } x - x_j > 0, \\
-3(|x - x_j|)^2 & \text{if } x - x_j < 0,
\end{cases}
\]

\[
\frac{\partial^2 \phi(|x - x_j|)}{\partial x^2} = 6(|x - x_j|),
\]

(20)

In this research we choose the Cubic Spline rather than the most popular ones, MQ and IMQ as a basis function because its simplicity and accuracy. Although there exists a substantial literature on choosing "optimal" shape parameters in MQ and IMQ, e.g. (Fasshauer and Zhang [2007], Fornberg and Wright, 2004) and (Kansa and Carlson, 1992), it is still an open question and there is no theoretical proof for selecting an optimal shape parameter \( c \) (cf. Wendland, 2005) in IMQ and MQ. To avoid this complexity, Cubic Spline will prevail in our RBF-approximation scheme.

3.1. Transforming PIDE to A System of ODEs by RBF

Given a set of interpolation points \( x_1, \ldots, x_j, \ldots, x_N \) in \( \mathbb{R} \), and an RBF \( \phi \), we can construct \( N \times N \) matrices \( A, A_x \) and \( A_{xx} \) defined by \( \phi(|x_i - x_j|) \) for \( 1 \leq i, j \leq N \). \( \phi'(|x_i - x_j|) \) and \( \phi''(|x_i - x_j|) \) respectively. Note in case the \( x_j \)'s are chosen according to the Equally Spacing Method, ESM, used in (Fausshauer et al., 2004a; Hon and Mao, 1999). In brief, Equally Spacing Method is the way to choose equally spaced points in a finite interval. In the ESM, we determine an interval \([x_{\min}, x_{\max}]\) outside of which we can neglect the contribution of \( u(x, \tau) \) to the global integral term of a PIDE [9], and for given \( N = 0, 1, 2, \ldots, \) simply put

\[
x_j := x_j^\Delta = x_{\min} + j \Delta x, \quad j = 0, 1, 2, \ldots, N - 1
\]

(21)
where \( \Delta x = (x_{\text{max}} - x_{\text{min}})/(N - 1) \). We also define a matrix-valued function 
\( y \rightarrow \mathbf{A}(y) \) by \( \phi(|x_i + y - x_j|) \) \( 1 \leq i, j \leq N \). If we substitute \( \tilde{u}(x, \tau) \) for \( u(x, \tau) \) in (9)
and require the PIDE to be satisfied in the interpolation points \( x_j \), we arrive at the following system of ODEs for the vector \( \boldsymbol{\rho}(\tau) := (\rho_1(\tau), \ldots, \rho_N(\tau)) \):

\[
\mathbf{A}\boldsymbol{\rho}_\tau = \frac{\sigma^2}{2} \mathbf{A}_{xx}\boldsymbol{\rho} + \left( r - q - \frac{\sigma^2}{2} - \lambda \eta \right) \mathbf{A}_x\boldsymbol{\rho} + (r + \lambda) \mathbf{A}\boldsymbol{\rho} + \\
\lambda \left( \int_{-\infty}^{\infty} \mathbf{A}(y)f(y)\,dy \right) \boldsymbol{\rho},
\]

where \( \rho_\tau := \frac{\partial \rho}{\partial \tau} \), and where we recall that \( f(y) \) is the probability density of the jump \( Y_i \sim \mathcal{N}(\mu_J, \sigma_J^2) \) : \( f(y) = (\sigma_J\sqrt{2\pi})^{-1} \exp \left( -\frac{(y - \mu_J)^2}{2\sigma_J^2} \right) \) in the Merton model, or \( f(y) = p\alpha_1 e^{-\alpha_1 x} 1_{x \geq 0} + (1 - p)\alpha_2 e^{\alpha_2 x} 1_{x \leq 0} \) in the Kou model. Before applying a suitable numerical integration algorithm to the integral terms in (22), we truncate the integrals from an infinite computational range to a finite one. Briani et al. (Briani et al., 2007), Cont and Voltchkova (Cont and Voltchkova, 2005), Tankov and Voltchkova (Tankov and Voltchkova, 2009) and d’Halluin et al. (d’Halluin et al., 2004, 2005) have provided different numerical techniques to find out a finite computational range so as to reduce the numerical approximation errors when doing this truncation. In this thesis we shall adopt Briani et al.’s numerical technique to truncate the integral domain of our PIDE (cf. Briani et al., 2007) in both the Merton and Kou model. See Appendix A for a proof.

Supposed \( \epsilon > 0 \), a formula of selecting a bounded interval \([y_-, y_\epsilon]\) for the set of points \( y \) in the Merton case is:

\[
y_\epsilon = \sqrt{-2\sigma_J^2 \log(\epsilon\sigma_J\sqrt{2\pi}/2)} + \mu_J, \quad \forall y \geq 0 \tag{23}
y_- = -y_\epsilon, \quad \forall y < 0 \tag{24}
\]

In the Kou model we have

\[
y_\epsilon = \log \left( \epsilon/p \right)/(1 - \alpha_1), \quad \forall y \geq 0 \tag{25}
y_- = -\log \left( \epsilon/(1 - p) \right)/(1 - \alpha_2), \quad \forall y < 0 \tag{26}
\]

We therefore transform equation (22) into

\[
\mathbf{A}\boldsymbol{\rho}_\tau = \frac{\sigma^2}{2} \mathbf{A}_{xx}\boldsymbol{\rho} + \left( r - q - \frac{\sigma^2}{2} - \lambda \eta \right) \mathbf{A}_x\boldsymbol{\rho} + (r + \lambda) \mathbf{A}\boldsymbol{\rho} + \\
\lambda \left( \int_{y_-}^{y_\epsilon} \mathbf{A}(y)f(y)\,dy \right) \boldsymbol{\rho}. \tag{27}
\]
We use matlab’s adaptive Gauss-Kronrod quadrature to evaluate the matrix of the integrals in (27): this amounts to approximating
\[
\int_{y_{m}}^{y_{n}} \phi(|x_i + y - x_j|) f(y) \, dy \approx \sum_{k=1}^{m} w_k \phi(|x_i + y_k - x_j|) f(y_k),
\]
where \( w_k \) and \( y_k \) are suitable quadrature weights and quadrature points; cf. [Shampine, 2008] for details. To simplify notations, we set
\[
F(x_i - x_j) = \sum_{k=1}^{m} w_k \phi(|x_i + y_k - x_j|) f(y_k).
\]
Then the integrals in equation (27) will be approximated by
\[
\int_{y_{m}}^{y_{n}} A(y) f(y) \, dy \approx \begin{bmatrix}
F(x_1 - x_1) & F(x_1 - x_2) & \cdots & F(x_1 - x_N) \\
F(x_2 - x_1) & F(x_2 - x_2) & \cdots & F(x_2 - x_N) \\
\cdots & \cdots & \cdots & \cdots \\
F(x_N - x_1) & F(x_N - x_2) & \cdots & F(x_N - x_N)
\end{bmatrix} = C(y).
\]
Substituting (29) into equation (27), we arrive at the new approximate equation:
\[
A \rho = \frac{\sigma^2}{2} A_{xx} \rho + \left( r - q - \frac{\sigma^2}{2} - \lambda \eta \right) A_x \rho + (r + \lambda) A \rho + \lambda C(y) \rho.
\]
As we have known the Cubic Spline is strictly conditionally positive definite function of order 2, the invertibility of \( A \) is not assumed without adding a real-valued polynomial of degree at most 1 in (16) (cf. [Wendland, 2005]). Nevertheless, Bos and Salkauskas proved that \( A \) is non-singular in a univariate case (cf. [Bos and Salkauskas, 1987], Theorem 5.1). As a result, the invertibility of \( A \) is still guaranteed.

We perform Gaussian elimination with partial pivoting to calculate \( A^{-1} \). Then, we multiply both sides of (30) by \( A^{-1} \) and obtain the following homogeneous system of ODEs with constant coefficients:
\[
\rho = A^{-1} \left( \frac{\sigma^2}{2} A_{xx} + \left( r - q - \frac{\sigma^2}{2} - \lambda \eta \right) A_x + (r + \lambda) A + \lambda C(y) \right) \rho \\
\equiv \Theta \rho
\]
(31)
where \( \Theta \) is defined by the left hand side. After some numerical experimentation, we found that the matrix \( \Theta \) is very stiff. To explain why \( \Theta \) is stiff, we shall use the following example to illustrate it. Suppose we select our maximum and minimum logarithm price \( x_{\text{min}} \) and \( x_{\text{max}} \) in (21) equal to \(-10\) and \(10\) respectively, then we use (21) to generate a list of 100 interpolation points. Based on the procedures and the ideas we have mentioned above we can get a \( 100 \times 100 \) matrix \( \Theta \) in (31). Then we measure the stiffness ratio of \( \Theta \). The stiffness ratio is the quotient of the largest and the smallest eigenvalues of the Jacobian matrix \( \Theta \). The ratio we have is \(1.2864 \times 10^5\). This implies that (31) is a stiff ODE and therefore we have to solve the ODEs by an implicit method, e.g. backward differentiation formulas (BDFs), a modified Rosenbrock formula of order 2, the trapezoidal rule or TR-BDF2, an implicit Runge-Kutta formula with a first stage that is a trapezoidal rule step and a second stage that is a backward differentiation formula of order two. In this paper we use former one.

4. Numerical Results

4.1. European Vanilla Options

In this section we present the numerical results of our Cubic Spline approximation scheme and compare these with Merton and Kou’s analytical option price formula for both puts and calls. Beside this, we also compare the results of our Cubic Spline approximation scheme with those of Briani et al.’s finite difference method (FD) with implicit and explicit (IMEX) scheme in [Briani et al., 2007] and Almendral et al.’s finite element method (FE) with backward differentiation formulas of order two (BDF2) and FD with BDF2 in [Almendral and Oosterlee, 2005b]. To measure the accuracy of our RBF-approximation, we use a set of evaluation points \( \hat{x}_i^{\Delta x} \), for which we will simply take the grid points

\[
\hat{x}_i := \hat{x}_i^{\Delta x} = \hat{x}_{\text{min}} + j \Delta x, \quad j = 0, 1, 2, \ldots, N_{\text{eval}} - 1.
\]

Here \( \Delta x = (\hat{x}_{\text{max}} - \hat{x}_{\text{min}})/(N_{\text{eval}} - 1) \) with \( x_{\text{min}} \leq \hat{x}_{\text{min}} \leq \hat{x}_{\text{max}} \leq x_{\text{max}} \) and \( N_{\text{eval}} \) is the number of the evaluation points chosen. To define our evaluation points. We set \( \hat{x}_{\text{min}} = K - 0.7 \times 10^{-\varsigma-1} \) and \( \hat{x}_{\text{max}} = K + 0.7 \times 10^{-\varsigma-1} \) where \( K \) is a strike price and \( \varsigma \) is the total number of the digits of \( K \). We will use the following two different measures for the errors,
the root-mean-square (rms) error:

$$E_2 = \sqrt{\frac{1}{N_{\text{eval}}} \sum_{0 \leq i \leq N_{\text{eval}}} |V(e^{\hat{x}_i}, t) - \tilde{u}(\hat{x}_i)|^2}, \quad (33)$$

and the relative error:

$$E_{\text{rel}}(\hat{x}, t) = \frac{|V(e^{\hat{x}}, t) - \tilde{u}(\hat{x}, t)|}{V(e^{\hat{x}}, t)}. \quad (34)$$

where $V(e^{\hat{x}}, t)$ and $\tilde{u}(\hat{x}, t)$ are the exact value and approximate value at the point $(\hat{x}, t)$ respectively. We also calculate the rate of convergence by using $E_2(\hat{x}_i, T)$. We define the formula:

$$E_2(\hat{x}_i, T) = C(1/N)^{R_2} \quad (35)$$

where $N$ is the number of interpolation points, $C$ is a constant number and $R_2$ is the rate of convergence which represent linear when they are equal to one or quadratic when they are equal to two.

It is known [Merton, 1976] that the analytical price of a European call/put option in the Merton Jump-diffusion model is given by

$$V_{\text{MJ}}(S_t, T - t, K, r, q, \sigma)$$

$$= \sum_{k=0}^{\infty} e^{-\lambda(1+\eta)(T-t)} \frac{(\lambda(1+\eta)(T-t))^k}{k!} V_{\text{BS}}(S_t, T - t, K, r_k, \sigma_k, q) \quad (36)$$

where $T - t$ is the time to maturity, $\eta = e^{\mu_J + \sigma^2_J/2} - 1$ represents the expected percentage change in the stock price originating from a jump, $\sigma^2_k = \sigma^2 + \frac{k\sigma^2_J}{T-t}$ the observed volatility, $r_k = r - \lambda \eta + k \log(1 + \eta)/(T - t)$, $q$ is the dividend and $V_{\text{BS}}$ the Black-Scholes price of a call and put, computed as

$$V_{\text{BS}}(S_t, T - t, K, r_k, \sigma_k, q)$$

$$= \begin{cases} 
Se^{-q(T-t)}\Phi(d_{+,k}) - Ke^{-r_k(T-t)}\Phi(d_{-,k}) & \text{call option,} \\
Ke^{-r_k(T-t)}\Phi(-d_{-,k}) - Se^{-q(T-t)}\Phi(-d_{+,k}) & \text{put option,}
\end{cases}$$

where $\Phi(\cdot)$ is the cumulative normal distribution and

$$d_{+,k} = \frac{\log(S/K) + (r_k - q + \sigma^2_k/2)(T-t)}{\sigma_k \sqrt{T-t}}, \quad d_{-,k} = d_{+,k} - \sigma_k \sqrt{T-t}. \quad (36)$$
For the derivation of $V_{MJ}(S_t, T - t, K, r, q, \sigma)$, we shall refer to the reader to (Merton, 1976; Cont and Tankov, 2004).

In general, for models where the characteristic function of the Lévy process is known, an analytical solution of PIDE (9) may be found using Fourier analysis (Carr and Madan, 1999; Lewis, 2001). For the sake of simplicity and accuracy we propose Jackson et al.’s Fourier Space Time-Stepping method rather than Carr-Madan’s Fast Fourier Transform (FFT) method (Carr and Madan, 1999) and Lewis’s FFT method (Lewis, 2001). In brief, the idea of this method is based on the Fourier transform of the PIDE. By making use of FFT and inverse Fast Fourier transform (FFT$^{-1}$), European Option price can be determined. The pricing formula of evaluating European option can be expressed as follows:

$$V_{\text{Kou}}(S, \tau, K, r, q) = \text{FFT}^{-1}[\text{FFT} [V_{\text{Kou}}(S, T)] e^{\psi \tau}],$$ (37)

where $\psi(z)$ is the characteristic function of the Kou model which can be defined as:

$$-\frac{\sigma^2 z^2}{2} + iz \gamma_c + \lambda(\frac{p\alpha_1}{\alpha_1 - iz} + \frac{(1-p)\alpha_2}{\alpha_2 + iz} - 1),$$

and $V_{\text{Kou}}(S, T)$ is the payoff function (10). For more details of this method, we shall refer the reader to (Jackson et al., 2008). This method has been reported to have second order convergence in space in European cases.

Our RBF-algorithm for numerically solving (9) with initial condition (10) runs as follows:

1. Find the RBF-approximation to the initial value $u(x, 0)$ using ESM (see [21]). This will provide us with a set of interpolation points $x_1, \ldots, x_n$, together with an initial vector $\mathbf{p}(0) = (p_1(0), \ldots, p_N(0))$.
2. Then use $\mathbf{p}(0)$ as initial value for the system (31). By using any stiff ODE solver, we find out the $\mathbf{p}(T)$ at time $T$.
3. Finally, substitute $\mathbf{p}(T)$ back into $\sum_{j=1}^{N} p_j(T) \phi(|x - x_j|)$ to get an approximate value of $u(x, T)$.

In our numerical experiment we implement the algorithm in MATLAB R2007b. We select our maximum and minimum logarithm price $x_{\text{min}}(\log(S_{\text{min}}))$ and $x_{\text{max}}(\log(S_{\text{max}}))$, as before, equal to $-10$ and $10$ respectively. Because of achieving more accurate approximation of the integral in (27), we also set $\epsilon$ in both [23] and [25] to be $3.72 \times 10^{-40}$ for finding a finite computational interval $[y_{-\epsilon}, y_{\epsilon}]$. Moreover, we use function quadgk which implements adaptive
Gauss-Kronrod quadrature for computing equation (28) as well as function
ode15s which implements backward differentiation formulas (BDFs) of order
two for the calculation of equation (31). The main reason of choosing it is the
following: According to (Iserles, 2009) BDFs of orders 1 and 2 are A-stable
(the stability region includes the entire left half complex plane). Since (31)
is stiff, according to Theorem 4.11 (The Dahlquist second barrier) of (Iserles,
2009), the highest order of an A stable multistep method\(^1\) such as BDFs,
is only two. We therefore conclude that our solution is second order conver-
gence in time. This conclusion is in line with the finding of (Pettersson et al.,
2008). In (Pettersson et al., 2008) Pettersson et al. show that second order
in time can be achieved in a European case due to the second order time-
stepping scheme, BDFs of order 2. Although they solve Black Schole PDE
rather than PIDE in their paper, an similar approach of solving European
option like our approximation scheme is applied.

All the parameters of all the tables except Table 3 and 6 are chosen from
different literatures. The parameter \(\sigma = 1\) in Table 3 and 6 is selected to
stress our numerical algorithm. From Table 1 to 6 \(E_2\) falls down when the
number of the interpolation points \(N\) increases. Our Cubic Spline approxi-
mation scheme can get second order convergence in space. This is due to the
limited smoothness of the Cubic Spline which has second order of convergence
(cf. (Wendland, 2005)). In Table 7 we compare the results of the FD used in
Briani et al.’s paper (Briani et al., 2007) with those using our Cubic Spline
approximation scheme. Our numerical approximation scheme can achieve
lower \(E_{rel}(\log S, T)\) than ARS-233 scheme and Explicit scheme. Table 7 and
9 are other comparisons of the accuracy between our Cubic Spline approxi-
mation scheme and Almendral and Oosterlee’s FD and FE with BDF2. To
illustrate a fair comparison, we set our maximum and minimum logarithm
price \(x_{\text{min}}\) and \(x_{\text{max}}\) same as Almendral and Oosterlee proposed in their nu-
merical experiments. Hence we set \([x_{\text{min}} \ x_{\text{max}}]\) equal to \([-4 \ 4]\) and \([-6 \ 6]\) in
the Merton model (Table 8) and the Kou model (Table 9) respectively. Our
Cubic Spline approximation scheme can attain lower \(E_{rel}(\log S, T)\) than FD
and FE with BDF2 in both the Merton and Kou cases.

\(^1\)Multistep methods are used for the numerical solution of ordinary differential equa-
tions. Conceptually, a numerical method starts from an initial point and then takes a
short step forward in time to find the next solution point. The process continues with
subsequent steps to map out the solution.
<table>
<thead>
<tr>
<th>N</th>
<th>$E_2(\hat{x}_i, T)$</th>
<th>$R_2$</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>4.421249E-03</td>
<td>N/A</td>
</tr>
<tr>
<td>600</td>
<td>1.192379E-04</td>
<td>2.016</td>
</tr>
<tr>
<td>1100</td>
<td>3.550438E-05</td>
<td>1.999</td>
</tr>
<tr>
<td>1600</td>
<td>1.678138E-05</td>
<td>2.000</td>
</tr>
<tr>
<td>2100</td>
<td>9.741214E-06</td>
<td>2.000</td>
</tr>
<tr>
<td>2600</td>
<td>6.353858E-06</td>
<td>2.001</td>
</tr>
<tr>
<td>3100</td>
<td>4.469202E-06</td>
<td>2.000</td>
</tr>
<tr>
<td>3600</td>
<td>3.313767E-06</td>
<td>2.000</td>
</tr>
</tbody>
</table>

Table 1: $E_2$ of the Cubic Spline approximation for pricing of a European call under the Merton Jump-diffusion model are presented. $N$ is the number of the interpolation points. $\hat{x}_i = \log S_i$ is any evaluation points of a range of $S$ from 0.3 to 1.7 and the total numbers are 201. $T$ is the Time-to-maturity. The parameters are: $r = 0.05$, $q = 0$, $\sigma = 0.15$, $\sigma_J = 0.45$, $\mu_J = -0.9$, $\lambda = 0.1$, $K = 1$ and $T = 0.25$. The parameters are taken from \cite{AndersenAndreasen2000}. The order of convergence is 2 in space.

<table>
<thead>
<tr>
<th>N</th>
<th>$E_2(\hat{x}_i, T)$</th>
<th>$R_2$</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>5.568295E-03</td>
<td>N/A</td>
</tr>
<tr>
<td>600</td>
<td>1.539291E-04</td>
<td>2.003</td>
</tr>
<tr>
<td>1100</td>
<td>4.596552E-05</td>
<td>1.994</td>
</tr>
<tr>
<td>1600</td>
<td>2.174172E-05</td>
<td>1.998</td>
</tr>
<tr>
<td>2100</td>
<td>1.263001E-05</td>
<td>1.997</td>
</tr>
<tr>
<td>2600</td>
<td>8.235945E-06</td>
<td>2.002</td>
</tr>
<tr>
<td>3100</td>
<td>5.793457E-06</td>
<td>2.000</td>
</tr>
<tr>
<td>3600</td>
<td>4.295872E-06</td>
<td>2.000</td>
</tr>
</tbody>
</table>

Table 2: $E_2$ of the Cubic Spline approximation for pricing of a European put under the Merton Jump-diffusion model are presented. $N$ is the number of the interpolation points. $\hat{x}_i = \log S_i$ is any evaluation points of a range of $S$ from 0.3 to 1.7 and the total numbers are 141. $T$ is the Time-to-maturity. The parameters are: $r = 0.05$, $q = 0.02$, $\sigma = 0.15$, $\sigma_J = 0.4$, $\mu_J = -1.08$, $\lambda = 0.1$, $K = 1$ and $T = 0.1$. The parameters are taken from \cite{AndersenAndreasen2000}. The order of convergence is 2 in space.
<table>
<thead>
<tr>
<th>$N$</th>
<th>$E_2(\hat{x}_i, T)$</th>
<th>$R_2$</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>6.883556E-04</td>
<td>N/A</td>
</tr>
<tr>
<td>600</td>
<td>1.889097E-05</td>
<td>2.007</td>
</tr>
<tr>
<td>1100</td>
<td>5.612280E-06</td>
<td>2.002</td>
</tr>
<tr>
<td>1600</td>
<td>2.651002E-06</td>
<td>2.002</td>
</tr>
<tr>
<td>2100</td>
<td>1.538267E-06</td>
<td>2.002</td>
</tr>
<tr>
<td>2600</td>
<td>1.003022E-06</td>
<td>2.002</td>
</tr>
<tr>
<td>3100</td>
<td>7.054350E-07</td>
<td>2.001</td>
</tr>
<tr>
<td>3600</td>
<td>5.227885E-07</td>
<td>2.004</td>
</tr>
</tbody>
</table>

Table 3: $E_2$ of the Cubic Spline approximation for pricing of a European call under the Merton Jump-diffusion model are presented. $N$ is the number of the interpolation points. $\hat{x}_i = \log S_i$ is any evaluation points of a range of $S$ from 0.3 to 1.7 and the total numbers are 141. $T$ is the Time-to-maturity. The parameters are: $r = 0.05$, $q = 0.01$, $\sigma = 1$, $\sigma_J = 0.6$, $\mu_J = -1.08$, $\lambda = 0.1$, $K = 1$ and $T = 1$, whereas the parameter $\sigma = 1$ is selected to stress our numerical algorithm. The order of convergence is 2 in space.

<table>
<thead>
<tr>
<th>$N$</th>
<th>$E_2(\hat{x}_i, T)$</th>
<th>$R_2$</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>4.034493E-03</td>
<td>N/A</td>
</tr>
<tr>
<td>600</td>
<td>1.113014E-04</td>
<td>2.004</td>
</tr>
<tr>
<td>1100</td>
<td>3.311668E-05</td>
<td>2.000</td>
</tr>
<tr>
<td>1600</td>
<td>1.565011E-05</td>
<td>2.000</td>
</tr>
<tr>
<td>2100</td>
<td>9.084581E-06</td>
<td>2.000</td>
</tr>
<tr>
<td>2600</td>
<td>5.925440E-06</td>
<td>2.001</td>
</tr>
<tr>
<td>3100</td>
<td>4.168045E-06</td>
<td>2.000</td>
</tr>
<tr>
<td>3600</td>
<td>3.090654E-06</td>
<td>2.000</td>
</tr>
</tbody>
</table>

Table 4: $E_2$ of the Cubic Spline approximation for pricing of a European put under the Kou Jump-diffusion model are presented. $N$ is the number of the interpolation points. $\hat{x}_i = \log S_i$ is any evaluation points of a range of $S$ from 0.3 to 1.7 and the total numbers are 141. $T$ is the Time-to-maturity. The parameters are: $r = 0$, $q = 0$, $\sigma = 0.2$, $\alpha_1 = 3$, $\alpha_2 = 2$, $\lambda = 0.2$, $p = 0.5$, $K = 1$ and $T = 0.2$. The parameters are taken from [Almendral and Oosterlee 2005b](http://jmlr.org/papers/v6/almendral05a.html). The order of convergence is 2 in space.
Table 5: $E_2$ of the Cubic Spline approximation for pricing of a European call under the Kou Jump-diffusion model are presented. $N$ is the number of the interpolation points. \( \hat{x}_i = \log S_i \) is any evaluation points of a range of $S$ from 0.3 to 1.7 and the total numbers are 141. $T$ is the Time-to-maturity. The parameters are: $r = 0.05$, $q = 0$, $\sigma = 0.15$, $\alpha_1 = 3.0465$, $\alpha_2 = 3.0465$, $\lambda = 0.1$, $p = 0.3445$, $K = 1$ and $T = 0.25$. The parameters are taken from (Carr and Mayo, 2007). The order of convergence is 2 in space.

<table>
<thead>
<tr>
<th>$N$</th>
<th>$E_2(\hat{x}_i, T)$</th>
<th>$R_2$</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>4.440921E-03</td>
<td>N/A</td>
</tr>
<tr>
<td>600</td>
<td>1.205043E-04</td>
<td>2.013</td>
</tr>
<tr>
<td>1100</td>
<td>3.587776E-05</td>
<td>1.999</td>
</tr>
<tr>
<td>1600</td>
<td>1.695646E-05</td>
<td>2.000</td>
</tr>
<tr>
<td>2100</td>
<td>9.842519E-06</td>
<td>2.000</td>
</tr>
<tr>
<td>2600</td>
<td>6.418917E-06</td>
<td>2.001</td>
</tr>
<tr>
<td>3100</td>
<td>4.514415E-06</td>
<td>2.001</td>
</tr>
<tr>
<td>3600</td>
<td>3.346805E-06</td>
<td>2.001</td>
</tr>
</tbody>
</table>

Table 6: $E_2$ of the Cubic Spline approximation for pricing of a European put under the Kou Jump-diffusion model are presented. $N$ is the number of the interpolation points. \( \hat{x}_i = \log S_i \) is any evaluation points of a range of $S$ from 0.3 to 1.7 and the total numbers are 141. $T$ is the Time-to-maturity. The parameters are: $r = 0.04$, $q = 0.03$, $\sigma = 1$, $\alpha_1 = 4$, $\alpha_2 = 4$, $\lambda = 0.3$, $p = 0.6$ $K = 1$ and $T = 1$, whereas the parameter $\sigma = 1$ is selected to stress our numerical algorithm. The order of convergence is 2 in space.

<table>
<thead>
<tr>
<th>$N$</th>
<th>$E_2(\hat{x}_i, T)$</th>
<th>$R_2$</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>6.710733E-04</td>
<td>N/A</td>
</tr>
<tr>
<td>600</td>
<td>1.841203E-04</td>
<td>2.007</td>
</tr>
<tr>
<td>1100</td>
<td>5.470175E-06</td>
<td>2.002</td>
</tr>
<tr>
<td>1600</td>
<td>2.584042E-06</td>
<td>2.002</td>
</tr>
<tr>
<td>2100</td>
<td>1.499549E-06</td>
<td>2.001</td>
</tr>
<tr>
<td>2600</td>
<td>9.779429E-07</td>
<td>2.001</td>
</tr>
<tr>
<td>3100</td>
<td>6.878041E-07</td>
<td>2.001</td>
</tr>
<tr>
<td>3600</td>
<td>5.098150E-07</td>
<td>2.003</td>
</tr>
</tbody>
</table>
Table 7: Comparison between Explicit scheme (Briani et al., 2007), ARS-233 Scheme (Briani et al., 2007) and Cubic Spline interpolation scheme in evaluating of European call/put under the Merton Jump-diffusion Model. The input parameters are: $r = 0.05$, $q = 0$, $\sigma = 0.2$, $\sigma_J = 0.8$, $\mu_J = 0$, $\lambda = 0.1$, $K = 100$, $T = 1$, and $x = \log 100$. Reference prices of 13.218501 (call) and 8.341444 (put) and parameters from Briani et al. (2007).
**Table 8: Comparison of FD with BDF2 (Almendral and Oosterlee, 2005b), FE with BDF2 (Almendral and Oosterlee, 2005b) and Cubic Spline interpolation scheme in evaluating of a European call (put) under the Merton Jump-diffusion Model. The input parameters are: \( r = 0 \), \( q = 0 \), \( \sigma = 0.2 \), \( \sigma_J = 0.5 \), \( \mu_J = 0 \), \( \lambda = 0.1 \), \( K = 1 \), \( T = 1 \), and \( S = 1 \). Reference prices of 0.094135525 for both call and put and parameters from (Almendral and Oosterlee, 2005b).**

<table>
<thead>
<tr>
<th>Method</th>
<th>N</th>
<th>Value</th>
<th>( E_{rel}(\log S, T) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>FD with BDF2</td>
<td>1025</td>
<td>9.411968E-02</td>
<td>1.682457e-04</td>
</tr>
<tr>
<td>FE with BDF2</td>
<td>1025</td>
<td>9.412972E-02</td>
<td>6.165536E-05</td>
</tr>
<tr>
<td>Cubic Spline</td>
<td>1025</td>
<td>9.413023E-02</td>
<td>5.621522E-005</td>
</tr>
</tbody>
</table>
FD with BDF2 (Almendral and Oosterlee, 2005b)

<table>
<thead>
<tr>
<th>N</th>
<th>Value</th>
<th>$E_{rel}(\log S, T)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>513</td>
<td>4.240E-02</td>
<td>6.346096E-03</td>
</tr>
</tbody>
</table>

FE with BDF2 (Almendral and Oosterlee, 2005b)

<table>
<thead>
<tr>
<th>N</th>
<th>Value</th>
<th>$E_{rel}(\log S, T)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>513</td>
<td>4.24579E-02</td>
<td>5.1285862E-03</td>
</tr>
</tbody>
</table>

Cubic Spline

<table>
<thead>
<tr>
<th>N</th>
<th>Value</th>
<th>$E_{rel}(\log S, T)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>513</td>
<td>4.254583E-02</td>
<td>3.061686E-03</td>
</tr>
</tbody>
</table>

Table 9: Comparison of FD with BDF2 (Almendral and Oosterlee, 2005b), FE with BDF2 (Almendral and Oosterlee, 2005b) and Cubic Spline interpolation scheme in evaluating of a European call (put) under the Kou Jump-diffusion Model. The input parameters are: $r = 0$, $q = 0$, $\sigma = 0.2$, $\alpha_1 = 3$, $\alpha_2 = 2$, $\lambda = 0.2$, $p = 0.5$, $K = 1$, $T = 0.2$, and $S = 1$. Reference prices of 0.0426761 for both call and put and parameters from (Almendral and Oosterlee, 2005b).
4.2. American Vanilla Put Options

In this section we adapt an RBF-algorithm to compute American put-option prices. We then compare the option prices obtained from our RBF-algorithm with the Jackson et al. FST methods of (Jackson et al., 2008). As mentioned in Section 2, an American put option problem is a free boundary problem because of the possibility of early exercise at any point during its life, leading to the free boundary condition:

\[ u(x, \tau) = \max \left( K - e^x, u(x, \tau) \right) \]

Together with the smooth pasting condition mentioned in section 2, this uniquely determines the exercise boundary.

The Jackson et al. FST methods suggest that their solutions can achieve second order in space when they implement their methods to price American put options. They implement their methods in the context of the LCP. As we have seen in Section 2, the value of an American option \( u(\tau, x) \) is always greater than or equal to the payoff function \( G(e^x) \). To numerically keep the condition \( u(\tau, x) - G(e^x) \geq 0 \) to be continuously held (see Section 2), this can be achieved when boundary conditions are applied. The numerical algorithm for this idea can be defined as follows:

\[
V(S, (m + 1)\Delta t, K, r, \sigma, q) = \max \{ \text{FFT}^{-1} \left[ \text{FFT} \left[ V(S, m\Delta t, K, r, \sigma, q) e^{\psi \Delta t} \right] \right], G(e^x) \} \tag{38}
\]

where time interval \( \Delta t \) is obtained by dividing time-to-maturity \( T \) by the total number \( M \), \( m\Delta \) is the time-step, where \( m \in \{0, 1, 2, \ldots, M - 1\} \), \( \psi(z) \) is the characteristic function of the Merton/Kou models, \( V(S, (m + 1)\Delta t, K, r, \sigma, q) \) is the American put price at time \((m + 1)\Delta t\) and the payoff condition \( G(e^x) \) is equal to \( \max(K - e^x, 0) \). These methods also are required to swap between real and Fourier spaces at each time-step when the American option prices are calculated at each time interval. This is due to no convenient representation of the \( \max(.,.) \) operator in Fourier space. For the full schematic and numerical description of this method, we refer readers to (Jackson et al., 2008).

As before, we use ESM to approximate \( u(x, 0) = \max(K - e^x, 0) \) and then continue to work with the interpolation points found at \( \tau = 0 \). The algorithm now reads as follows:
1. Divide time-to-maturity $T$ by total numbers of time-steps $M$ to obtain time interval $\Delta t$ and create a list of equally spaced time-points $m\Delta t$, $m \in \{0, 1, 2, \ldots, M - 1\}$.

2. Find the RBF-approximation to the initial value $u(x, 0)$ using ESM. This will provide us with a set of interpolation points $x_1, \ldots, x_n$, together with an initial vector $\rho(0) = (\rho_1(0), \ldots, \rho_N(0))$.

3. Assume we have already determined $\rho(m\Delta t)$ (if $m = 0$, we have $\rho(0)$) in equation (31). Solve the system of (stiff) ODEs to find $\rho((m+1)\Delta t)$ at the next successive time-step, $(m+1)\Delta t$.

4. Then at time $(m+1)\Delta t$, for each interpolation point $x_i$, define

$$u(x_i, (m+1)\Delta t) = \max ((K - e^{x_i}), \sum_{j=1}^{N} \rho_j((m+1)\Delta t) \phi(|x_i - x_j|)).$$

5. Find a new vector $\rho((m+1)\Delta t)$ such that $u(x_i, (m+1)\Delta t) = \sum_{j=1}^{N} \rho_j((m+1)\Delta t) \phi(|x_i - x_j|)$ for all $i$.

6. Repeat Step 3.) to 5.) until $m = M - 1$.

7. Finally, substitute $\rho(T)$ back into $\sum_{j=1}^{N} \rho_j(T) \phi(|x - x_j|)$ to get an approximate value of $u(x, T)$.

The settings of our numerical experiment are the same as those in section 4.1. The results from Table 10 and 15 suggest that our Cubic Spline approximation method for pricing of American put options is second order in spatial variables and first order in time variables when the number of interpolation numbers $N$ and the number of time-steps $M_0$ are twofold and fourfold respectively.

5. Conclusion

We have implemented an RBF method to solve the PIDE boundary value problem for pricing American put and European call/put options on a dividend-paying stock in a Merton (Merton, 1976) and Kou (Kou, 2002) jump-diffusion market. By using Briani et al.’s numerical scheme, we find out a finite computational range of our global integral. Our results also suggest that the Cubic Spline approximation scheme can achieve second-order convergence in both spatial and time variables when it is used to compute European call/put options as well as second order convergence in spatial variables and first-order convergence in time variables when it is used to compute American put options. Beside this, we compare our RBF-approximation
Table 10: $E_2$ of the Cubic Spline approximation for pricing of an American put under the Merton model are presented. $N$ is the number of the interpolation points. $M_0$ is the number of the time steps. $\hat{x}_i = \log S_i$ is any evaluation points of a range of $S$ from 0.3 to 1.7 and the total numbers are 141. $T$ is the Time-to-maturity. The parameters are: $r = 0.05$, $q = 0$, $\sigma = 0.15$, $\sigma_f = 0.45$, $\mu_f = -0.9$, $\lambda = 0.1$, $K = 1$ and $T = 0.25$. The parameters are taken from (Andersen and Andreasen, 2000). The order of convergence is 2 in space and 1 in time.

<table>
<thead>
<tr>
<th>$N$</th>
<th>$M_0$</th>
<th>$E_2(\hat{x}_i, T)$</th>
<th>$R_2$</th>
</tr>
</thead>
<tbody>
<tr>
<td>225</td>
<td>10</td>
<td>1.065795E-03</td>
<td>N/A</td>
</tr>
<tr>
<td>450</td>
<td>40</td>
<td>2.946799E-04</td>
<td>1.8547</td>
</tr>
<tr>
<td>900</td>
<td>160</td>
<td>7.496031E-05</td>
<td>1.9749</td>
</tr>
<tr>
<td>1800</td>
<td>640</td>
<td>2.030907E-05</td>
<td>1.8840</td>
</tr>
<tr>
<td>3600</td>
<td>2560</td>
<td>5.402608E-06</td>
<td>1.9104</td>
</tr>
</tbody>
</table>

Table 11: $E_2$ of the Cubic Spline approximation for pricing of an American put under the Merton model are presented. $N$ is the number of the interpolation points. $M_0$ is the number of the time steps. $\hat{x}_i = \log S_i$ is any evaluation points of a range of $S$ from 0.3 to 1.7 and the total numbers are 141. $T$ is the Time-to-maturity. The parameters are: $r = 0.05$, $q = 0.02$, $\sigma = 0.0$, $\sigma_f = 0.4$, $\mu_f = -1.08$, $\lambda = 0.1$, $K = 1$ and $T = 0.1$. The parameters are taken from (Andersen and Andreasen, 2000). The order of convergence is 2 in space and 1 in time.

<table>
<thead>
<tr>
<th>$N$</th>
<th>$M_0$</th>
<th>$E_2(\hat{x}_i, T)$</th>
<th>$R_2$</th>
</tr>
</thead>
<tbody>
<tr>
<td>225</td>
<td>10</td>
<td>2.21531E-04</td>
<td>N/A</td>
</tr>
<tr>
<td>450</td>
<td>40</td>
<td>2.846899E-04</td>
<td>1.6956</td>
</tr>
<tr>
<td>900</td>
<td>160</td>
<td>7.999130E-05</td>
<td>1.8315</td>
</tr>
<tr>
<td>1800</td>
<td>640</td>
<td>2.240200E-05</td>
<td>1.8362</td>
</tr>
<tr>
<td>3600</td>
<td>2560</td>
<td>5.980951E-06</td>
<td>1.9052</td>
</tr>
</tbody>
</table>

23
Table 12: $E_2$ of the Cubic Spline approximation for pricing of an American put under the Merton model are presented. $N$ is the number of the interpolation points. $M_0$ is the number of the time steps. $\hat{x}_i = \log S_i$ is any evaluation points of a range of $S$ from 0.3 to 1.7 and the total numbers are 141. $T$ is the Time-to-maturity. The parameters are: $r = 0.05$, $q = 0.01$, $\sigma = 1$, $\sigma_J = 0.6$, $\mu_J = -1.08$, $\lambda = 0.1$, $K = 1$ and $T = 1$, whereas the parameter $\sigma = 1$ is selected to stress our numerical algorithm. The order of convergence is 2 in space and 1 in time.

<table>
<thead>
<tr>
<th>N</th>
<th>$M_0$</th>
<th>$E_2(\hat{x}_i, T)$</th>
<th>$R_2$</th>
</tr>
</thead>
<tbody>
<tr>
<td>225</td>
<td>10</td>
<td>9.891272E-04</td>
<td>N/A</td>
</tr>
<tr>
<td>450</td>
<td>40</td>
<td>1.839887E-04</td>
<td>2.4265</td>
</tr>
<tr>
<td>900</td>
<td>160</td>
<td>4.490722E-05</td>
<td>2.0346</td>
</tr>
<tr>
<td>1800</td>
<td>640</td>
<td>1.087954E-05</td>
<td>2.0453</td>
</tr>
<tr>
<td>3600</td>
<td>2560</td>
<td>2.671901E-06</td>
<td>2.0257</td>
</tr>
</tbody>
</table>

Table 13: $E_2$ of the Cubic Spline approximation for pricing of an American put under the Kou Jump-diffusion model are presented. $N$ is the number of the interpolation points. $M_0$ is the number of the time steps. $\hat{x}_i = \log S_i$ is any evaluation points of a range of $S$ from 0.3 to 1.7 and the total numbers are 141. $T$ is the Time-to-maturity. The parameters are: $r = 0$, $q = 0$, $\sigma = 0.2$, $\alpha_1 = 3$, $\alpha_2 = 2$, $\lambda = 0.2$, $p = 0.5$, $K = 1$ and $T = 0.2$. The parameters are taken from (Almendral and Oosterlee, 2005b). The order of convergence is 2 in space and 1 in time.

<table>
<thead>
<tr>
<th>N</th>
<th>$M_0$</th>
<th>$E_2(\hat{x}_i, T)$</th>
<th>$R_2$</th>
</tr>
</thead>
<tbody>
<tr>
<td>225</td>
<td>10</td>
<td>6.589592E-04</td>
<td>N/A</td>
</tr>
<tr>
<td>450</td>
<td>40</td>
<td>2.074520E-04</td>
<td>1.6674</td>
</tr>
<tr>
<td>900</td>
<td>160</td>
<td>5.580623E-05</td>
<td>1.8943</td>
</tr>
<tr>
<td>1800</td>
<td>640</td>
<td>1.498170E-05</td>
<td>1.8972</td>
</tr>
<tr>
<td>3600</td>
<td>2560</td>
<td>4.013413E-06</td>
<td>1.9003</td>
</tr>
</tbody>
</table>
Table 14: $E_2$ of the Cubic Spline approximation for pricing of an American put under the Kou Jump-diffusion model are presented. $N$ is the number of the interpolation points. $M_0$ is the number of the time steps. $\hat{x}_i = \log S_i$ is any evaluation points of a range of $S$ from 0.3 to 1.7 and the total numbers are 141. $T$ is the Time-to-maturity. The parameters are: $r = 0.05$, $q = 0$, $\sigma = 0.15$, $\alpha_1 = 3.0465$, $\alpha_2 = 3.0465$, $\lambda = 0.1$, $p = 0.3445$, $K = 1$ and $T = 0.25$. The parameters are taken from [Carr and Mayo, 2007]. The order of convergence is 2 in space and 1 in time.

<table>
<thead>
<tr>
<th>$N$</th>
<th>$M_0$</th>
<th>$E_2(\hat{x}_i, T)$</th>
<th>$R_2$</th>
</tr>
</thead>
<tbody>
<tr>
<td>225</td>
<td>10</td>
<td>9.200626E-04</td>
<td>N/A</td>
</tr>
<tr>
<td>450</td>
<td>40</td>
<td>3.007262E-04</td>
<td>1.6133</td>
</tr>
<tr>
<td>900</td>
<td>160</td>
<td>7.902329E-05</td>
<td>1.9281</td>
</tr>
<tr>
<td>1800</td>
<td>640</td>
<td>2.126498E-05</td>
<td>1.8938</td>
</tr>
<tr>
<td>3600</td>
<td>2560</td>
<td>5.647481E-06</td>
<td>1.9128</td>
</tr>
</tbody>
</table>

Table 15: $E_2$ of the Cubic Spline approximation for pricing of an American put under the Kou Jump-diffusion model are presented. $N$ is the number of the interpolation points. $M_0$ is the number of the time steps. $\hat{x}_i = \log S_i$ is any evaluation points of a range of $S$ from 0.3 to 1.7 and the total numbers are 141. $T$ is the Time-to-maturity. The parameters are: $r = 0.04$, $q = 0.03$, $\sigma = 1$, $\alpha_1 = 4$, $\alpha_2 = 4$, $\lambda = 0.3$, $p = 0.6$, $K = 1$ and $T = 1$, whereas the parameter $\sigma = 1$ is selected to stress our numerical algorithm. The order of convergence is 2 in space and 1 in time.

<table>
<thead>
<tr>
<th>$N$</th>
<th>$M_0$</th>
<th>$E_2(\hat{x}_i, T)$</th>
<th>$R_2$</th>
</tr>
</thead>
<tbody>
<tr>
<td>225</td>
<td>10</td>
<td>6.553901E-04</td>
<td>N/A</td>
</tr>
<tr>
<td>450</td>
<td>40</td>
<td>9.645335E-05</td>
<td>2.7645</td>
</tr>
<tr>
<td>900</td>
<td>160</td>
<td>2.225501E-05</td>
<td>2.1157</td>
</tr>
<tr>
<td>1800</td>
<td>640</td>
<td>4.893031E-06</td>
<td>2.1853</td>
</tr>
<tr>
<td>3600</td>
<td>2560</td>
<td>1.144807E-06</td>
<td>2.0956</td>
</tr>
</tbody>
</table>
method against FDM and FEM. Our results suggest that one can achieve a high accuracy by implementing our meshless scheme. Moreover, in terms of meshless interpolation methods, we use cubic spline as a basis function rather than MQ and IMQ. This basis function can avoid the open question of choosing an optimal shape parameter $c$ of both MQ and IMQ. Finally, several drawbacks associated with grid-based methods like the FDM have been avoided: we do not have to make assumptions on the behavior of the solutions outside of the solution domain, we seem to avoid the stability problems associated with explicit or implicit finite difference schemes. Moreover, we dramatically improve the accuracy of pricing options in particular for small times to maturity.

At this stage of development, the Cubic Spline approximation scheme is first order in time for American put options although a second order time-stepping scheme, BDFs of order 2 is implemented. A further investigation into various approaches to improve the Cubic Spline approximation time order will be treated in a future paper. Our Method extends in principle to pure jump Lévy type models for the underlying stocks, like the Variance Gamma (VG) model or the CGMY model.

**Appendix A. A Finite Computational Range in the Jump-diffusion Model**

In the Merton Model suppose in a domain $\Omega \in \mathbb{R}$ European option price $u(x, \tau)$ satisfies Lipchitz inequality such that

$$|u(x_1, \tau) - u(x_2, \tau)| \leq L|x_1 - x_2|, \forall x_1, x_2 \in \Omega.$$

Then we choose a parameter $\epsilon > 0$ and select the bounded intervals $[y_-, y_\epsilon]$ as the set of all points $y$ that verify

$$k(y) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(y-\mu)^2}{2\sigma^2}} \geq \epsilon.$$

Because of the symmetry of $k(y)$ we set $y_- = -y_\epsilon$. Then the truncation of the integral domain giving an error to approximation of the problem can be
estimated by
\[
\left| \int_{-\infty}^{\infty} (u(x + y) - u(x))k(y) \, dy - \int_{-y_c}^{y_c} (u(x + y) - u(x))k(y) \, dy \right|
\]
\[
\leq L \left| \int_{-\infty}^{-y_c} (x + y - x)k(y) \, dy - \int_{-y_c}^{y_c} (x + y - x)k(y) \, dy \right| \quad \text{(A.1a)}
\]
\[
\leq L \left( \int_{-\infty}^{-y_c} |y|k(y) \, dy + \int_{y_c}^{\infty} |y|k(y) \, dy \right) \quad \text{(A.1b)}
\]
\[
= 2 \int_{y_c}^{\infty} y \frac{1}{\sqrt{2\pi}\sigma_J} \exp\left(-\frac{(y - \mu_J)^2}{2\sigma_J^2}\right) \, dy \quad \text{(A.1c)}
\]
\[
= 2 \int_{y_c - \mu_J}^{\infty} (y + \mu_J) \frac{1}{\sqrt{2\pi}\sigma_J} \exp\left(-\frac{y^2}{2\sigma_J^2}\right) \, dy \quad \text{(A.1d)}
\]
\[
= 2 \int_{y_c - \mu_J}^{\infty} (y + \mu_J) \frac{1}{\sqrt{2\pi}\sigma_J} \exp\left(-\frac{y^2}{2\sigma_J^2}\right) \, dy \quad \text{(A.1e)}
\]
\[
\leq 2 \int_{y_c - \mu_J}^{\infty} (y + \mu_J) \frac{1}{\sqrt{2\pi}\sigma_J} \exp\left(-\frac{y^2}{2\sigma_J^2}\right) \, dy \quad \text{(A.1f)}
\]
\[
= \frac{4\sigma_J}{\sqrt{2\pi}} \exp\left(-\frac{(y_c - \mu_J)^2}{2\sigma_J^2}\right) \quad \text{(A.1g)}
\]
\[
= 2\sigma_J^2 \epsilon \quad \text{(A.1h)}
\]

Hence by using (A.1g) and (A.1h),
\[
y_c = \sqrt{-2\sigma_J^2 \log(\epsilon\sigma_J\sqrt{2\pi}/2)} + \mu_J \quad \text{(A.2)}
\]

We use the aforementioned arguments to find the finite computational range 
\([y_\epsilon, y] \] in the Kou model. We carry out the reasoning for the positive semi-axis (the reasoning goes similarly for the negative semi-axis) and set \(k(y) = p\alpha_1 e^{-\alpha_1 y} \) for \(y \geq 0 \) \(((1 - p)\alpha_2 e^{\alpha_2 x} \) for \(y < 0 \)). Then, \(y_c \) can be found out by
the following equations:
\[
\left| \int_0^\infty (u(x + y) - u(x))\lambda f(y)\,dy - \int_0^{y_c} (u(x + y) - u(x))\lambda f(y)\,dy \right| \\
\leq L \left| \int_0^\infty (x + y - x)\lambda f(y)\,dy - \int_0^{y_c} (x + y - x)\lambda f(y)\,dy \right| \quad (A.3a)
\]
\[
\leq L \int_{y_c}^{\infty} |y|f(y)\,dy \\
= \int_{y_c}^{\infty} |y|p\alpha_1 e^{-\alpha_1y}\,dy \quad (A.3b)
\]
\[
= p\alpha_1 e^{-y_\alpha_1} \left( \frac{1}{\alpha_1^2} + \frac{y_c}{\alpha_1} \right) \quad (A.3c)
\]
\[
(\text{Gradsteyn and Ryzhik, 1994, equation 3.351})
\]
\[
= \frac{p}{\alpha_1} e^{-y_\alpha_1} (1 + y_c\alpha_1) \quad (A.3d)
\]
\[
\leq \frac{p}{\alpha_1} e^{-y_\alpha_1} \alpha_1 e^{y_c} \quad (A.3e)
\]
\[
= pe^{y_c(1-\alpha_1)} \quad (A.3f)
\]
\[
= \epsilon, \quad (A.3g)
\]
\[
= \epsilon, \quad (A.3h)
\]
as a result,
\[
y_\epsilon = \log(\epsilon/p)/(1 - \alpha_1). \quad (A.4)
\]
Similar arguments can be applied to \( y < 0 \), so
\[
y_{-\epsilon} = -\log \left( \epsilon/(1 - p) \right)/(1 - \alpha_2). \quad (A.5)
\]

References


