Application of the improved fast Gauss transform to option pricing under jump-diffusion processes

doi: 10.21314/JCF.2014.276

Journal of Computational Finance **18**(2), 31–55

Application of the improved fast Gauss transform to option pricing under jump-diffusion processes

Takayuki Sakuma
Graduate School of Business Sciences, University of Tsukuba,
3-29-1 Otsuka, Bunkyo-ku, Tokyo 112-0012, Japan;
email: sakuma@gssm.otsuka.tsukuba.ac.jp

Yuji Yamada
Graduate School of Business Sciences, University of Tsukuba,
3-29-1 Otsuka, Bunkyo-ku, Tokyo 112-0012, Japan;
email: yuji@gssm.otsuka.tsukuba.ac.jp

(Received January 18, 2012; revised August 22, 2012; accepted December 16, 2012)

Efficient kernel summation is an active research topic in machine learning and computational physics. Fast multipole methods (FMMs) in particular are known as efficient computational methods in these fields, but they have not gained much attention in computational finance. In this paper, we apply the improved fast Gauss transform (IFGT), a version of an FMM, to the computation of European-type option prices under Merton's jump-diffusion model. IFGT is applied to computing the nonlocal integral terms in partial integrodifferential equations, and our results indicate that IFGT is useful for the fast computation of option pricing under this model.

1 INTRODUCTION

Jump-diffusion models, which incorporate jumps in asset dynamics, have attracted attention in financial industries because the standard Black–Scholes model cannot capture the skew and smile effects observed in option markets. Jump-diffusion models are convenient in practice in the sense that the analytical formula is appropriate for some types of option prices. For example, under Merton's jump-diffusion model (Merton 1976), an analytical expression of path-independent European call option prices exists. Analytical formulas for some path-dependent options also exist under Kou's jump-diffusion model (Kou and Wang 2003, 2004). Otherwise, Monte Carlo simulation and the finite-difference method are the standard tools used to compute prices. Monte Carlo simulation in particular is generally time consuming. In the case

of the finite-difference method, it is necessary to compute nonlocal integral operators for the corresponding partial integrodifferential equations (PIDEs) numerically, which is computationally expensive. A popular efficient approach to computing the operators is to use the fast Fourier transform (FFT) (see, for example, Andersen and Andreasen 2000; Tavella and Randall 2000; d'Halluin *et al* 2004, 2005); however, this method requires two FFT operations at each time step and does not allow the direct use of nonuniform grids.

In the fields of machine learning and computational physics, efficient kernel summation is actively studied. The fast multipole methods (FMMs) originally proposed by Greengard and Rokhlin (1987) have, in particular, been developed for efficient computation. In principle, the computational cost of these methods is $O(M)$ given M points, which is less than that of the FFT (for which the cost is $O(M \log M)$). FMMs have not yet attracted much attention in the field of computational finance, and they have been applied to only a few cases. For example, the FMM method called the fast Gauss transform (FGT) (Greengard and Strain 1991) was applied by d'Halluin *et al* (2005) to compute integral terms in PIDEs, but they suggest the FFT approach is superior to that of FGT. The FGT approach requires an impractically large number of grid points in order to achieve the same accuracy as the FFT method. Broadie and Yamamoto (2003) applied FGT to the multinomial method and stochastic mesh method, and the results of their numerical experiments indicate that the use of FGT enhances the efficiency of these methods for pricing European and Bermudan options. They also applied FGT with the double-exponential integration formula for pricing path-dependent options (Broadie and Yamamoto 2005). For the present paper, we applied the improved fast Gauss transform (IFGT) (Yang *et al* 2005; Raykar *et al* 2005) to computing nonlocal integral operators in PIDEs under Merton's jump-diffusion model. Numerical results indicate that IFGT evaluation is more efficient than FFT evaluation and can achieve the same accuracy with a practical number of grid points.

Outline

This paper is organized as follows. Section 2 introduces the IFGT. In Section 3, we introduce Merton's jump-diffusion model and discuss the numerical results of pricing call options under this model using the FFT and IFGT methods. In Section 4, we apply IFGT to a two-dimensional version of Merton's jump-diffusion model. Section 5 is the conclusion.

2 IMPROVED FAST GAUSS TRANSFORM

In this section, we briefly introduce IFGT (Yang *et al* 2005; Raykar *et al* 2005). Given source points $x_1, x_2, \ldots, x_{M_1}$, target points $y_1, y_2, \ldots, y_{M_2}$ and weight coefficients

$C_1, C_2, \ldots, C_{M_1}$, the kernel summation is the computation of the following sum:

$$F(y_j) = \sum_{i=1}^{M_1} C_i K(x_i, y_j).$$

FMM has been widely used for efficient computation; this efficiency is due to the basic idea of the method, which is to analytically approximate the potentially full-rank matrix as a sum of low-rank approximations. FGT (Greengard and Strain 1991) is an FMM for computing with respect to the Gaussian kernel

$$K(x_i, y_j) = \exp\left(-\frac{1}{h^2}\|y_j - x_i\|^2\right).$$

FGT applies the Hermite expansion of $\exp(-(1/h^2)\|y_j - x_i\|^2)$, but terms in the expansion increase exponentially in the multidimensional case. On the other hand, IFGT (Yang *et al* 2005; Raykar *et al* 2005) uses the multivariate Taylor expansion and succeeds in computing with a cost that grows only polynomially in the multidimensional case. The basic schemes of IFGT given in Raykar *et al* (2005) are as follows. Given a desired error bound of $\varepsilon > 0$, IFGT computes the approximated $F(y_j)$ (denoted by $\hat{F}(y_j)$), such that

$$\max_j \frac{|F(y_j) - \hat{F}(y_j)|}{\sum_{i=1}^{M_1} |C_i|} < \varepsilon.$$

To achieve the desired error bound, the points are divided into clusters S_l ($l = 1, 2, \ldots, L$) by farthest point clustering (Gonzalez 1985). Given a cluster center $x*_l$, the term $\exp(-(2/h^2)\Delta y_j \Delta x_i)$ in the expansion

$$\exp\left(-\frac{1}{h^2}\|y_j - x_i\|^2\right) = \exp\left(-\frac{1}{h^2}\|\Delta y_j\|^2\right)\exp\left(-\frac{1}{h^2}\|\Delta x_i\|^2\right)\exp\left(-\frac{2}{h^2}\Delta y_j \Delta x_i\right)$$

is expressed as a multivariate Taylor expansion at $x*_l$ as the following:

$$\exp\left(-\frac{2}{h^2}\Delta y_j \Delta x_i\right) \approx \sum_{\beta=0}^{p_i} \frac{2^\beta}{\beta!}\left(\frac{y_j - x*_l}{h}\right)^\beta \left(\frac{x_i - x*_l}{h}\right)^\beta.$$

The truncation number p_i can be chosen separately for each source point x_i. Then

$$\hat{F}(y_j) = \sum_{l=1}^{L}\sum_{\beta=0}^{p_i} C'_\beta \exp\left(-\frac{1}{h^2}\|y_j - x*_l\|^2\right)\left(\frac{y_j - x*_l}{h}\right)^\beta, \qquad (2.1)$$

where

$$C'_\beta = \frac{2^\beta}{\beta!} \sum_{x_i \in S_l} C_i \exp\left(-\frac{1}{h^2}\|x_i - x*_l\|^2\right)\left(\frac{x_i - x*_l}{h}\right)^\beta.$$

Defining $p_{\max} := \max_i p_i$ and using an indicator function, (2.1) is equivalent to

$$\hat{F}(y_j) = \sum_{l=1}^{L} \sum_{|\beta| \le p_{\max}-1} C'_\beta \exp\left(-\frac{1}{h^2}\|y_j - x*_l\|^2\right)\left(\frac{y_j - x*_l}{h}\right)^\beta,$$

where

$$C'_\beta = \frac{2^\beta}{\beta!} \sum_{x_i \in S_l} C_i \exp\left(-\frac{1}{h^2}\|x_i - x*_l\|^2\right)\left(\frac{x_i - x*_l}{h}\right)^\beta 1_{|\beta| \le p_i - 1}.$$

The computational cost of computing $\hat{F}(y_j)$ at M_2 points is $O(M_2)$; that of computing C'_β is $O(M_1)$. Because these can be computed separately, the total computational cost is $O(M_1 + M_2)$. Furthermore, the Gaussian kernel decays rapidly, so Raykar *et al* (2005) achieve faster computation by discarding all the points in a cluster that are farther away from the target point y_j than a certain distance (called the cutoff radius and denoted c_y^l):

$$\hat{F}(y_j) = \sum_{|y_j - x*_l| < c_y^l} \sum_{|\beta| \le p_{\max}-1} C'_\beta \exp\left(-\frac{1}{h^2}\|y_j - x*_l\|^2\right)\left(\frac{y_j - x*_l}{h}\right)^\beta,$$

where

$$C'_\beta = \frac{2^\beta}{\beta!} \sum_{x_i \in S_l} C_i \exp\left(-\frac{1}{h^2}\|x_i - x*_l\|^2\right)\left(\frac{x_i - x*_l}{h}\right)^\beta 1_{|\beta| \le p_i - 1}.$$

The details of the algorithm to choose the number of clusters, L, truncation numbers, p_i, and cutoff radii, c_y^l, are given in Raykar *et al* (2005), which also discusses in detail the differences between FGT and IFGT. In the present paper, we stress the following two advantages of IFGT over the original FGT.

First, Raykar *et al* (2005) note that FGT uses two types of expansion (far-field and local) and needs the cumbersome translation between the two expansions. On the other hand, the multivariate Taylor expansion in IFGT has nice properties as both a far-field and a local expansion; we can thus avoid representing two kinds of expansion, as well as the translation operation.

Second, Raykar *et al* (2005) suggest that, in order to obtain the same error bound, IFGT requires fewer terms of the multivariate Taylor expansion than the FGT requires of either the Hermite or the Taylor expansion. This seems consistent with the result of d'Halluin *et al* (2005) that FGT requires a larger number of grid points to achieve an accuracy similar to that of FFT. This is because the slow convergence of the Hermite expansion makes finer spatial discretization necessary in order to achieve the desired accuracy.

3 ONE-DIMENSIONAL JUMP-DIFFUSION MODELS

3.1 Merton's jump-diffusion model

In this paper, we consider Merton's jump-diffusion model (Merton 1976). Merton's model dynamics of an asset value S_t under the risk-neutral measure gives

$$S(t) = S(0)\exp\left[(r - \tfrac{1}{2}\sigma^2 - \lambda\kappa)t + \sigma W(t) + \sum_{i=1}^{N(t)} Y_i\right],$$

where r is the risk-free rate, σ is the volatility of the Brownian term, $N(t)$ is a Poisson process with parameter λ and $\kappa = E[\mathrm{e}^Y] - 1$. The logarithm of the jump size Y_i follows the normal distribution $f(x)$, with mean μ and variance γ^2:

$$f(x) = \frac{1}{2\pi\gamma^2}\exp\left(-\frac{(x-\mu)^2}{2\gamma^2}\right). \tag{3.1}$$

Then the value of option $V(S,t)$ with maturity T satisfies the following PIDE (Merton 1976):

$$\frac{\partial V}{\partial \tau} = \frac{\sigma^2}{2}\frac{\partial^2 V}{\partial S^2} + (r - \lambda\kappa)S\frac{\partial V}{\partial S} - (r+\lambda)V + \lambda\int_{\eta\geq 0} V(S\eta, t)f(\eta)\,\mathrm{d}\eta, \tag{3.2}$$

where $\tau = T - t$ is the time to maturity. The change of variables $x = \log(S/K)$ and $v(x,\tau) = V(S,\tau)$ gives the PIDE

$$\frac{\partial v}{\partial \tau} = \frac{\sigma^2}{2}\frac{\partial^2 v}{\partial x^2} + \left(r - \frac{\sigma^2}{2} - \lambda\kappa\right)\frac{\partial v}{\partial x} - (r+\lambda)v + \lambda\int_{-\infty}^{+\infty} v(x+z,\tau)f(z)\,\mathrm{d}z. \tag{3.3}$$

In the case of a European call option, the initial condition is given by

$$v(x,0) = \max(K\mathrm{e}^x - K, 0),$$

where K is the strike price. The boundary conditions are given by

$$v(x,\tau)\begin{cases} \to 0, & x \to -\infty, \\ \simeq K\mathrm{e}^x - K\mathrm{e}^{-r\tau}, & x \to +\infty. \end{cases}$$

In the case of a digital call option, the initial condition is given by $v(x,0) = 1$ if $x > 0$, and the boundary conditions are given by

$$v(x,\tau)\begin{cases}\to 0, & x \to -\infty,\\ \simeq \mathrm{e}^{-r\tau}, & x \to +\infty.\end{cases}$$

For simplicity, we numerically solve this equation via the Crank–Nicolson scheme for $1 \leqslant i \leqslant M$ and $1 \leqslant j \leqslant N$ (M represents the number of discretization points, and N represents the number of time steps),

$$\frac{v_i^{j+1} - v_i^j}{\Delta\tau} = \tfrac{1}{2}D(v_i^{j+1} + v_i^j) + \lambda\,\mathrm{Int}(x_i, t_j). \tag{3.4}$$

$D(v_i^j)$ are discretizations of the differential terms, and $\mathrm{Int}(x_i, t_j)$ is the integral term, defined as follows:

$$\begin{aligned}\mathrm{Int}(x,\tau) &:= \frac{1}{2\pi\gamma^2}\int_{-\infty}^{+\infty} v(s,\tau)\exp\left(-\frac{(s-(x-\mu))^2}{2\gamma^2}\right)\mathrm{d}s\\ &\approx \frac{1}{2\pi\gamma^2}\sum_{i=1}^{M} v(s_i,\tau)\exp\left(-\frac{(s_i-(x-\mu))^2}{2\gamma^2}\right)w_i\,\Delta s,\end{aligned}$$

where the w_i terms are weights that depend on the numerical evaluation of the integral. At each time step, we evaluate $\mathrm{Int}(x_i, t_j)$ explicitly, but use of iteration schemes (see, for example, d'Halluin *et al* 2004, 2005) is also possible as a means of maintaining second-order accuracy with respect to time. Direct computation of the integral term $I(x_i, t_j)$ is computationally expensive, so in this paper we apply IFGT for the computation. IFGT can deal with nonuniform grids directly, and it is also possible to construct nonuniform grids in such a way that more accuracy is achieved. However, for simplicity, we use a uniform grid in this paper. Another widely used approach is the FFT method (see, for example, Andersen and Andreasen 2000; Tavella and Randall 2000; d'Halluin *et al* 2004, 2005). At each time step, the procedure of the FFT method is to first compute the Fourier transform of v. Next, we multiply the Fourier transform of v by the Fourier transform of f and, finally, we compute the inverse Fourier transform. The FFT method is more efficient than direct computation, but two FFT operations are necessary at each time step. Also, the FFT method does not allow the direct use of nonuniform grids and requires extended regions to avoid wraparound effects.

3.2 Numerical experiments

IFGT in this paper is executed on freely available C++ software.[1] The integral operator $\mathrm{Int}(x,\tau)$ is approximated by the trapezoid rule. The desired error bound ε is set as

[1] Codes are available at www.umiacs.umd.edu/labs/cvl/pirl/vikas/Software/IFGT/IFGT_code.htm.

10^{-6}, and the upper limit on the number of clusters is set as 10. In addition, all the truncation numbers are set as a constant number, p_c, and we vary the value of $p_{max}(= p_c)$ to see how p_{max} influences the accuracy of option prices. The grid points are fixed under the Crank–Nicolson scheme, so for the IFGT evaluation we apply farthest point clustering at the initial time step only. The number of grid points to compute $\text{Int}(x, \tau)$ is set as 2^α, where α is the smallest integer satisfying $M < 2^\alpha$.

Table 1 on the next page and Table 2 on page 41 list the results of a comparison between the IFGT method and the FFT method for European calls and digital calls, respectively. Truncation of the multivariate Taylor expansion in the IFGT affects the accuracy of the numerical evaluation of $\text{Int}(x, \tau)$ in PIDEs. For example, a small number for p_{max} (eg, $p_{max} = 15$) is not sufficient for approximating $\text{Int}(x, \tau)$ successfully, and the maximum error value cannot necessarily be reduced just by increasing M and N. On the other hand, the IFGT method achieves an accuracy similar to that of FFT if p_{max} is sufficiently large ($p_{max} = 20$ for European calls and $p_{max} = 35$ for digital calls). In addition, the values of $\text{Ratio}(T)$ are greater than 2 in most cases, which indicates that IFGT is more efficient even if it is compared with just one operation of FFT. Figure 1 on page 44 shows the price, Delta and Gamma values of a digital call; as shown, these values are smooth functions of the stock price.

As presented in Table 3 on page 45 and Table 4 on page 45, the value of γ appearing in the integral operator $\text{Int}(x, \tau)$ influences the IFGT evaluation. Decreasing γ increases the effect of exponential decay on the integrand, leading to rapid convergence of the multivariate Taylor expansion. Therefore, for example, $p_{max} = 1$ is sufficient for pricing European and digital calls with $\gamma = 0.05$.

It is possible to apply IFGT to another jump-diffusion model, called the stochastic volatility model with jumps in return and volatility (SVCJ model) (Duffie *et al* 2000), as is shown in Appendix A. It is interesting to consider the possibility of applying IFGT to Kou's jump-diffusion model (Kou 2002), but, even without the help of IFGT, we can easily achieve a linear computational cost. One method is proposed in Carr and Mayo (2007), and we detail this in Appendix B.

4 APPLICATION TO THE TWO-DIMENSIONAL MERTON JUMP-DIFFUSION MODEL

In principle, IFGT can be applied to multidimensional cases. Therefore, as one example, we consider the following two-dimensional Merton model, proposed by Huang and Kou (2006). However, we assume here that the logarithm of the jump size follows a normal distribution rather than an asymmetric Laplace distribution. With this model, the asset values S_1 and S_2 follow under the risk-neutral measure

TABLE 1 European call price using the IFGT and FFT methods ($K = 100$, $T = 0.25$). [Table continues on next two pages.]

		IFGT ($p_{\max} = 15$)				FFT		
M	N	Max error	Time (s)	Ratio	Ratio(T)	Max error	Time (s)	Ratio
254	50	0.003335	0.015	—	4.133	0.003313	0.062	—
506	100	0.000878	0.062	4.133	3.258	0.000905	0.202	3.258
1010	200	0.000204	0.203	3.274	4.379	0.000172	0.889	4.401
2018	400	0.000097	0.839	4.133	4.396	0.000063	3.688	4.148
4034	800	0.000040	3.309	3.944	4.788	0.000021	15.844	4.296
8066	1600	0.000047	13.394	4.048	5.100	0.000012	68.314	4.312

		IFGT ($p_{\max} = 20$)				FFT		
M	N	Max error	Time (s)	Ratio	Ratio(T)	Max error	Time (s)	Ratio
254	50	0.003304	0.019	—	3.263	0.003313	0.062	—
506	100	0.000908	0.062	3.263	3.258	0.000905	0.202	3.258
1010	200	0.000166	0.265	4.274	3.355	0.000172	0.889	4.401
2018	400	0.000057	1.182	4.460	3.120	0.000063	3.688	4.148
4034	800	0.000023	4.275	3.617	3.706	0.000021	15.844	4.296
8066	1600	0.000011	16.330	3.820	4.183	0.000012	68.314	4.312

TABLE 1 Continued.

		IFGT ($p_{\max} = 25$)				FFT		
M	N	Max error	Time (s)	Ratio	Ratio(T)	Max error	Time (s)	Ratio
254	50	0.003304	0.031	—	2.000	0.003313	0.062	—
506	100	0.000908	0.074	2.387	2.730	0.000905	0.202	3.258
1010	200	0.000166	0.312	4.216	2.849	0.000172	0.889	4.401
2018	400	0.000057	1.212	3.885	3.043	0.000063	3.688	4.148
4034	800	0.000023	4.955	4.088	3.198	0.000021	15.844	4.296
8066	1600	0.000011	19.916	4.019	3.430	0.000012	68.314	4.312

		IFGT ($p_{\max} = 35$)				FFT		
M	N	Max error	Time (s)	Ratio	Ratio(T)	Max error	Time (s)	Ratio
254	50	0.003304	0.029	—	2.138	0.003313	0.062	—
506	100	0.000908	0.109	3.759	1.853	0.000905	0.202	3.258
1010	200	0.000166	0.418	3.835	2.127	0.000172	0.889	4.401
2018	400	0.000057	1.587	3.797	2.324	0.000063	3.688	4.148
4034	800	0.000023	6.446	4.062	2.458	0.000021	15.844	4.296
8066	1600	0.000011	24.972	3.874	2.736	0.000012	68.314	4.312

TABLE 1 Continued.

		IFGT ($p_{\max} = 45$)				FFT		
M	N	Max error	Time (s)	Ratio	Ratio(T)	Max error	Time (s)	Ratio
254	50	0.003304	0.031	—	2.000	0.003313	0.062	—
506	100	0.000908	0.124	4.000	1.629	0.000905	0.202	3.258
1010	200	0.000166	0.497	4.008	1.789	0.000172	0.889	4.401
2018	400	0.000057	1.975	3.974	1.867	0.000063	3.688	4.148
4034	800	0.000023	8.011	4.056	1.978	0.000021	15.844	4.296
8066	1600	0.000011	30.841	3.850	2.215	0.000012	68.314	4.312

Parameters used are $r = 0.05$, $\sigma = 0.25$, $\lambda = 0.1$, $\mu = -0.9$ and $\gamma = 0.35$. Max error represents the maximum of the difference between the price computed by the Crank–Nicolson scheme and the analytical prices given in Merton (1976) on $[\log(90/K), \log(110/K)]$ with respect to x. Ratio represents the ratio of the CPU time for the given grid points to the time with half as many points in each direction. Ratio(T) represents the ratio of the CPU time taken in the case of the FFT method to the CPU time taken in the case of the IFGT method.

TABLE 2 Digital call price using the IFGT and FFT methods ($K = 100$, $T = 0.25$). [Table continues on next two pages.]

		IFGT ($p_{max} = 15$)				FFT		
M	N	Max error	Time (s)	Ratio	Ratio(T)	Max error	Time (s)	Ratio
254	50	0.001953	0.015	—	3.067	0.002129	0.046	—
506	100	0.000551	0.062	4.133	3.258	0.000558	0.202	4.391
1010	200	0.000205	0.234	3.774	3.675	0.000139	0.860	4.257
2018	400	0.000303	0.794	3.393	4.613	0.000035	3.663	4.259
4034	800	0.000327	3.386	4.264	4.566	0.000009	15.459	4.220
8066	1600	0.000335	12.881	3.804	5.190	0.000002	66.850	4.324
		IFGT ($p_{max} = 20$)				FFT		
M	N	Max error	Time (s)	Ratio	Ratio(T)	Max error	Time (s)	Ratio
254	50	0.002127	0.015	—	3.067	0.002129	0.046	—
506	100	0.000618	0.062	4.133	3.258	0.000558	0.202	4.391
1010	200	0.000197	0.264	4.258	3.258	0.000139	0.860	4.257
2018	400	0.000096	0.970	3.674	3.776	0.000035	3.663	4.259
4034	800	0.000070	4.156	4.285	3.720	0.000009	15.459	4.220
8066	1600	0.000063	16.409	3.948	4.074	0.000002	66.850	4.324

TABLE 2 Continued.

		IFGT ($p_{\max} = 25$)				FFT		
M	N	Max error	Time (s)	Ratio	Ratio(T)	Max error	Time (s)	Ratio
254	50	0.002121	0.015	—	3.067	0.002129	0.046	—
506	100	0.000548	0.078	5.200	2.590	0.000558	0.202	4.391
1010	200	0.000130	0.302	3.872	2.848	0.000139	0.860	4.257
2018	400	0.000034	1.148	3.801	3.191	0.000035	3.663	4.259
4034	800	0.000012	4.543	4.957	3.403	0.000009	15.459	4.220
8066	1600	0.000008	18.438	4.059	3.626	0.000002	66.850	4.324

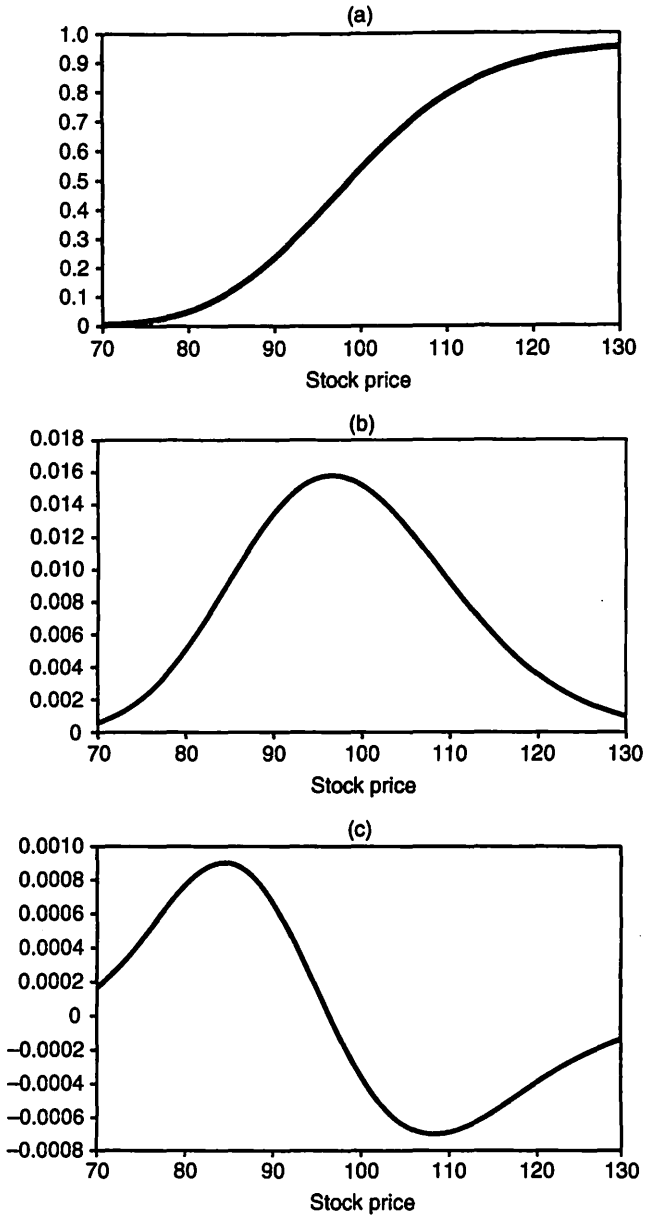
		IFGT ($p_{\max} = 35$)				FFT		
M	N	Max error	Time (s)	Ratio	Ratio(T)	Max error	Time (s)	Ratio
254	50	0.002127	0.031	—	1.484	0.002129	0.046	—
506	100	0.000556	0.093	3.000	2.172	0.000558	0.202	4.391
1010	200	0.000137	0.405	4.355	2.123	0.000139	0.860	4.257
2018	400	0.000033	1.551	3.830	2.362	0.000035	3.663	4.259
4034	800	0.000009	5.948	3.835	2.599	0.000009	15.459	4.220
8066	1600	0.000003	25.253	4.246	2.647	0.000002	66.850	4.324

TABLE 2 Continued.

		IFGT ($p_{max} = 45$)				FFT		
M	N	Max error	Time (s)	Ratio	Ratio(T)	Max error	Time (s)	Ratio
254	50	0.002128	0.031	—	1.484	0.002129	0.046	—
506	100	0.000558	0.124	4.000	1.629	0.000558	0.202	4.391
1010	200	0.000139	0.486	3.919	1.770	0.000139	0.860	4.257
2018	400	0.000035	1.949	4.010	1.879	0.000035	3.663	4.259
4034	800	0.000008	7.842	4.024	1.971	0.000009	15.459	4.220
8066	1600	0.000002	29.802	3.800	2.243	0.000002	66.850	4.324

Parameters used are $r = 0.05$, $\sigma = 0.25$, $\lambda = 0.1$, $\mu = -0.9$ and $\gamma = 0.35$. Max error represents the maximum difference between the price computed by the Crank–Nicolson scheme and the analytical prices given in Merton (1976) on $[\log(90/K), \log(110/K)]$ with respect to x. Ratio represents the ratio of the CPU time for the given grid points to the time with half as many points in each direction. Ratio(T) represents the ratio of the CPU time in the case of the FFT method to the CPU time in the case of the IFGT method.

FIGURE 1 Digital call option ($K = 100$, $T = 0.25$).



Parameters used are $r = 0.05$, $\sigma = 0.25$, $\lambda = 0.1$, $\mu = -0.9$ and $\gamma = 0.35$. (a) Option value. (b) Delta value. (c) Gamma value.

TABLE 3 Max error for different $p_{\max}$ and γ values in the case of a European call ($M = 8066$, $N = 1600$).

$p_{\max}$	$\gamma = 0.40$	$\gamma = 0.35$	$\gamma = 0.15$	$\gamma = 0.05$	$\gamma = 0.0001$
1	0.011004	0.002559	0.000119	0.000010	0.000010
5	0.021614	0.020654	0.004607	0.000010	0.000010
10	0.000765	0.002238	0.006046	0.000010	0.000010
20	0.000050	0.000011	0.006046	0.000010	0.000010
30	0.000050	0.000011	0.000264	0.000010	0.000010
40	0.000050	0.000011	0.000010	0.000010	0.000010
50	0.000050	0.000011	0.000010	0.000010	0.000010
60	0.000050	0.000011	0.000010	0.000010	0.000010
70	0.000050	0.000011	0.000010	0.000010	0.000010

TABLE 4 Max error for different $p_{\max}$ and γ values in the case of a digital call ($M = 8066$, $N = 1600$).

$p_{\max}$	$\gamma = 0.40$	$\gamma = 0.35$	$\gamma = 0.15$	$\gamma = 0.05$	$\gamma = 0.0001$
1	0.000593	0.000639	0.000014	0.000002	0.000002
5	0.000336	0.001453	0.000268	0.000002	0.000002
10	0.000849	0.000981	0.000682	0.000002	0.000002
20	0.000085	0.000063	0.000484	0.000002	0.000002
30	0.000002	0.000006	0.000123	0.000002	0.000002
40	0.000002	0.000003	0.000007	0.000002	0.000002
50	0.000002	0.000002	0.000002	0.000002	0.000002
60	0.000002	0.000002	0.000002	0.000002	0.000002
70	0.000002	0.000002	0.000002	0.000002	0.000002

(Huang and Kou 2006)

$$S_1(t) = S_1(0)\exp\left[(r - \tfrac{1}{2}\sigma_1^2 - \lambda\kappa_1)t + \sigma_1 W_1(t) + \sum_{i=1}^{N(t)} Y_i^{(1)}\right],$$

$$S_2(t) = S_2(0)\exp\Bigg[(r - \tfrac{1}{2}\sigma_2^2 - \lambda\kappa_2)t + \sigma_2[\rho W_1(t) + \sqrt{1-\rho^2}W_2(t)] + \sum_{i=1}^{N(t)} Y_i^{(2)}\Bigg],$$

where $\kappa_k = E[e^{Y^k}] - 1$, W_1 and W_2 are independent Brownian motions, ρ is the correlation and $N(t)$ is a Poisson process with parameter $\lambda = \lambda_c + \lambda_1 + \lambda_2$. In this model, each asset exhibits a common jump with λ_c and independent jumps with λ_1 and λ_2, respectively. The logarithm of each jump size Y_i follows a normal distribution. The infinitesimal generator of process $(S_1(t), S_2(t))$ is given in Huang and Kou (2006) as

$$\begin{aligned}\operatorname{Im} v = {} & (r - \tfrac{1}{2}\sigma_1^2 - \lambda\kappa_1)\frac{\partial v}{\partial x_1} + (r - \tfrac{1}{2}\sigma_2^2 - \lambda\kappa_2)\frac{\partial v}{\partial x_2} + \frac{\sigma_1^2}{2}\frac{\partial^2 v}{\partial {x_1}^2} \\ & + \frac{\sigma_2^2}{2}\frac{\partial^2 v}{\partial {x_2}^2} + \rho\sigma_1\sigma_2\frac{\partial^2 v}{\partial x_1 x_2} \\ & + \lambda_c \int_{-\infty}^{+\infty}\int_{-\infty}^{+\infty} v(x_1 + y_1, x_2 + y_2, \tau) f^C_{Y^{(1)},Y^{(2)}}(y_1, y_2)\,dy_1\,dy_2 \\ & - \lambda_c v(x_1, x_2, \tau) \\ & + \lambda_1 \int_{-\infty}^{+\infty} v(x_1 + y_1, x_2, \tau) f_{Y^{(1)}}(y_1)\,dy_1 - \lambda_1 v(x_1, x_2, \tau) \\ & + \lambda_2 \int_{-\infty}^{+\infty} v(x_1, x_2 + y_2, \tau) f_{Y^{(2)}}(y_2)\,dy_2 - \lambda_2 v(x_1, x_2, \tau),\end{aligned}$$

where $f^C_{Y^{(1)},Y^{(2)}}$ is the bivariate normal density with mean m_c and variance J_c,

$$m_c = \begin{pmatrix} m_{1,c} \\ m_{2,c} \end{pmatrix}, \qquad J_c = \begin{pmatrix} \nu_{1,c}^2 & \rho_c \nu_{1,c}\nu_{2,c} \\ \rho_c \nu_{1,c}\nu_{2,c} & \nu_{1,c}^2 \end{pmatrix},$$

and $f_{Y^{(1)}}$, $f_{Y^{(2)}}$ are independent normal densities with means m_1 and m_2 and variances v_1^2 and v_2^2, respectively. Then, letting $x_1 = \log(S_1/C_f)$ and $x_2 = \log(S_2/C_f)$ (C_f is a scaling factor), the value of option v satisfies the following PIDE (see, for example, Cont and Tankov 2003):

$$v_\tau - \operatorname{Im} v + rv = 0.$$

To solve the two-dimensional PIDE numerically, we use the simplest alternating direction implicit (ADI) scheme of the Douglas–Rachford method (Douglas and Rachford 1956):

$$\begin{aligned}(1 - \theta A_1)Y &= [1 + A_0 + (1 - \theta)A_1 + A_2]V^n, \\ (1 - \theta A_2)V^{n+1} &= Y - \theta A_2 V^n.\end{aligned}$$

A_0 denotes the mixed derivatives and the double integral operators, A_1 denotes the spatial derivatives in the x_1 direction and A_2 denotes the spatial derivatives in the

x_2 direction. This method is of first-order accuracy in time, and at each time step the Thomas algorithm is used to solve for Y and V^{n+1}. In the following numerical experiment, we set $\theta = 0.5$ and use uniform grid points for simplicity. If we also assume $\lambda_1 = \lambda_2 = 0$ for simplicity, we need to compute the following integral at each time step:

$$\begin{aligned}\mathrm{Int}(x_1, x_2, \tau) &:= \int_{-\infty}^{+\infty}\int_{-\infty}^{+\infty} v(x_1 + y_1, x_2 + y_2, \tau) f^C_{Y^{(1)},Y^{(2)}}(y_1, y_2)\,\mathrm{d}y_1\,\mathrm{d}y_2 \\ &= \int_{-\infty}^{+\infty}\int_{-\infty}^{+\infty} v(s_1, s_2, \tau) f^C_{Y^{(1)},Y^{(2)}}(s_1 - x_1, s_2 - x_2)\,\mathrm{d}s_1\,\mathrm{d}s_2,\end{aligned}$$

where $f^C_{Y^{(1)},Y^{(2)}}(s_1 - x_1, s_2 - x_2)$ is a bivariate normal density

$$\begin{aligned}&\frac{1}{2\pi\nu_{1,c}\nu_{2,c}\sqrt{1-\rho_c{}^2}} \\ &\quad\times\exp\left(-\frac{1}{2(1-\rho_c{}^2)}\left[\frac{(s_1-(x_1+m_{1,c}))^2}{\nu_{1,c}^2}\frac{(s_2-(x_2+m_{2,c}))^2}{\nu_{2,c}^2}\right.\right. \\ &\qquad\left.\left.-2\rho_c\frac{(s_1-(x_1+m_{1,c}))(s_2-(x_2+m_{2,c}))}{\nu_{1,c}\nu_{2,c}}\right]\right).\end{aligned}$$

We can apply IFGT to this integral by expressing it as follows (see Appendix C for the derivation):

$$v(s_1, s_2, \tau)\exp\left(-\frac{(s_1-(x_1+m_{1,c}))^2 + (s_2'-(x_2'+m_{2,c}'))^2 + (s_3'(s_1,s_2)-(x_3'(x_1,x_2)+m_{3,c}'))^2}{2\nu_{1,c}^2(1+|\rho_c|)}\right),$$

where

$$x_2' := \frac{\nu_{1,c}}{\nu_{2,c}}x_2, \qquad s_2' := \frac{\nu_{1,c}}{\nu_{2,c}}s_2, \qquad m_{2,c}' := \frac{\nu_{1,c}}{\nu_{2,c}}m_{2,c},$$

$$x_3'(x_1, x_2) := \sqrt{\frac{|\rho_c|}{1-|\rho_c|}}x_1 - \mathrm{sgn}(\rho_c)\sqrt{\frac{|\rho_c|}{1-|\rho_c|}}x_2',$$

$$s_3'(s_1, s_2) := \sqrt{\frac{|\rho_c|}{1-|\rho_c|}}s_1 - \mathrm{sgn}(\rho_c)\sqrt{\frac{|\rho_c|}{1-|\rho_c|}}s_2',$$

$$m_{3,c}' := \sqrt{\frac{|\rho_c|}{1-|\rho_c|}}m_{1,c} - \mathrm{sgn}(\rho_c)\sqrt{\frac{|\rho_c|}{1-|\rho_c|}}m_{2,c}'.$$

TABLE 5 Basket call price using the IFGT and FFT methods ($K = 100$, $T = 0.15$).

		IFGT ($p_{\max} = 1$)				FFT		
M	N	Price	Time (s)	Ratio	Ratio(T)	Price	Time (s)	Ratio
62	32	2.040948	0.390	—	2.392	2.041042	0.933	—
124	64	2.067723	3.024	7.754	2.447	2.067734	7.399	7.930
248	128	2.072201	22.720	7.513	2.774	2.072202	63.020	8.517
496	256	2.073188	185.689	8.173	3.021	2.073188	561.027	8.902

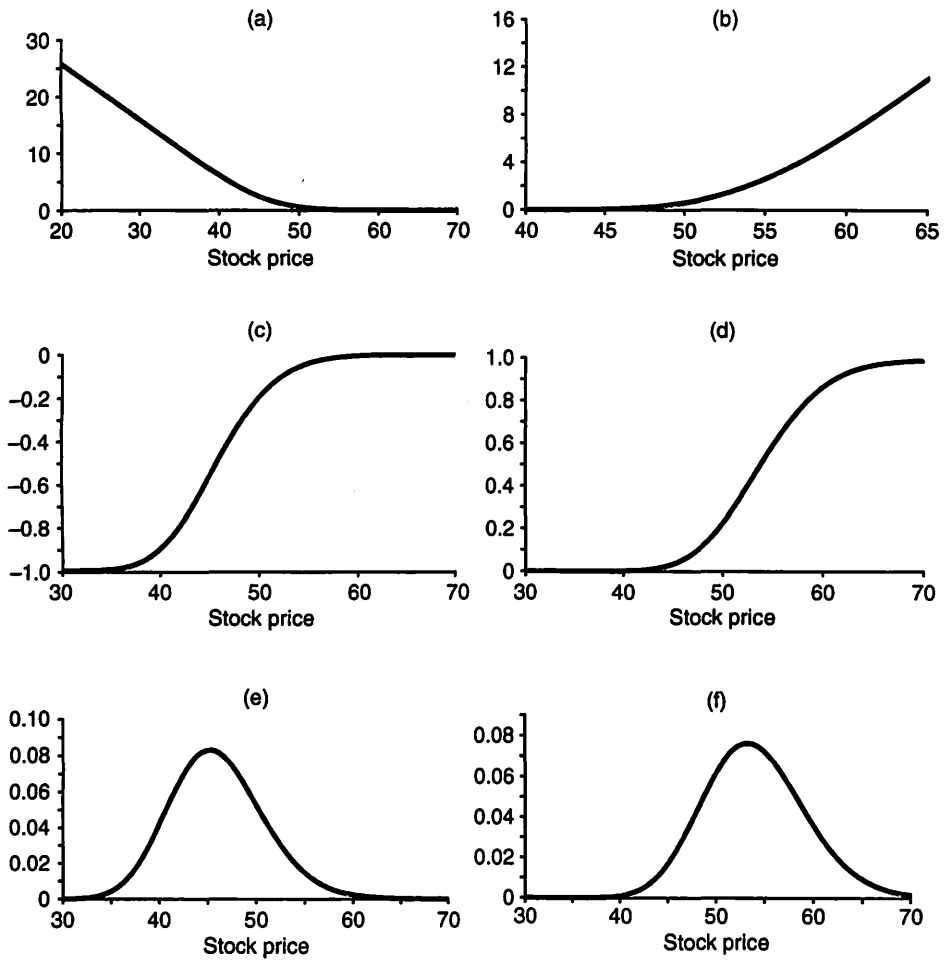
Parameters are $r = 0.05$, $\sigma_1 = 0.15$, $\sigma_2 = 0.15$, $\rho = -0.5$, $\lambda_c = 0.1$, $m_{1,c} = -1.00$, $v_{1,c} = 0.45$, $m_{2,c} = -0.75$, $v_{2,c} = 0.40$, $\rho_c = -0.5$, $C_f = 50$ and $S_1(0) = S_2(0) = 50$. The reference price is 2.073939 computed by 1 000 000 Monte Carlo simulation paths (Glasserman 2004) with Sobol sequences.

Table 5 lists the pricing results for a basket call with payoff $\max(C_f e^{x_1} + C_f e^{x_2} - K, 0)$ for the case of an integral operator approximated by the double trapezoid rule. As in the one-dimensional case, all the truncation numbers are equal. The desired error bound ε is set at 10^{-6}, and the upper limit on the number of clusters is set at 10. The Douglas–Rachford method with explicit evaluation of the integral operator seems to give linear convergence, as expected. Again, we achieve the same accuracy in less time than using the FFT method. Values of ρ_c and $v_{1,c}$ appearing in the integral operator $\mathrm{Int}(x_1, x_2, \tau)$ influence the IFGT evaluation. Moreover, the exponential effect in the integrand is larger than in the one-dimensional case, and the multivariate Taylor expansion converges rapidly. As a result, a truncation number of $p_{\max} = 1$ is sufficient to achieve the same accuracy as the FFT method, and, according to our numerical experiments, this is true for $v_{1,c}$ across the range from 0.1 to 0.7 with $\rho_c = 0.5$. Figure 2 on the facing page shows the price, Delta, and Gamma values of a spread call with payoff $\max(C_f e^{x_2} - C_f e^{x_1} - K, 0)$; as shown, these values are smooth functions of the stock price.

5 CONCLUSION

In this paper, we use IFGT (Yang *et al* 2005; Raykar *et al* 2005) to solve PIDEs efficiently under Merton's jump-diffusion model. The implementation is straightforward, and our numerical examples demonstrate that the IFGT method is more efficient than the FFT method and can achieve the same accuracy with a practical number of grid points. For evaluation of PIDEs, we apply the simple finite difference method (the Crank–Nicolson scheme in one-dimension and the Douglas–Rachford scheme in two-dimensions), but it is easy to expect that a combination of IFGT with a higher-order ADI scheme (see, for example, Craig and Sneyd 1988; Hundsdorfer 2002), along with

FIGURE 2 Spread call option ($K = 4.0$, $T = 0.15$): price, Delta and Gamma.



Parameters are $r = 0.05$, $\sigma_1 = 0.15$, $\sigma_2 = 0.15$, $\rho = -0.5$, $\lambda_c = 0.1$, $m_{1,c} = -1.00$, $v_{1,c} = 0.45$, $m_{2,c} = -0.75$, $v_{2,c} = 0.40$, $\rho_c = -0.5$ and $C_f = 50$. (a) Option value, $S_2(0) = 50$. (b) Option value, $S_1(0) = 50$. (c) Delta value, $S_2(0) = 50$. (d) Delta value, $S_1(0) = 50$. (e) Gamma value, $S_2(0) = 50$. (f) Gamma value, $S_1(0) = 50$.

the use of nonuniform grid points, can achieve greater efficiency. IFGT is applied only for a Merton-type jump model in the present paper, but another FMM, known as the kernel-independent FMM (see, for example, Ying *et al* 2004; Fong and Darve 2009), can be used for other jump models as well. We expect that the use of such an FMM will contribute to efficient option pricing under general Lévy processes, and this is left for future research.

APPENDIX A. APPLICATION OF IMPROVED FAST GAUSS TRANSFORM TO THE STOCHASTIC VOLATILITY MODEL WITH JUMPS IN RETURN AND VOLATILITY

Letting $X_t = \log(S_t)$ and V_t be the instantaneous variance, the asset under the SVCJ model (Duffie *et al* 2000; Feng and Linetsky 2008) follows

$$\begin{aligned} X_t &= (u - \tfrac{1}{2}V_{t-})\,\mathrm{d}t + \sqrt{V_{t-}}\,\mathrm{d}W_{1t} + \mathrm{d}J_t, \\ V_t &= \kappa(\theta - V_{t-})\,\mathrm{d}t + \xi\sqrt{V_{t-}}\,\mathrm{d}W_{2t} + \mathrm{d}J_t{}^v, \end{aligned}$$

where $(J_t, J_t{}^v)$ is a two-dimensional jump process, W_{1t} and W_{2t} are Brownian motions with $\mathrm{d}W_{1t}\,\mathrm{d}W_{2t} = \rho\,\mathrm{d}t$, κ is a mean-reversion parameter and ξ is a volatility parameter of V_t. The infinitesimal generator for (X_t, V_t) is given in Feng and Linetsky (2008) as

$$\begin{aligned} \mathrm{Im}\, f := \left(u - \frac{v}{2}\right)\frac{\partial f}{\partial x} + \kappa(\theta - v)\frac{\partial f}{\partial v} + \frac{v}{2}\frac{\partial^2 f}{\partial x^2} + \frac{\xi^2 v}{2}\frac{\partial^2 f}{\partial v^2} + \frac{\rho\xi v}{2}\frac{\partial^2 f}{\partial x\,\partial v} \\ + \lambda\int_0^{+\infty}\int_{-\infty}^{+\infty} f(x + x_1, v + v_1)p(x_1, v_1)\,\mathrm{d}x_1\,\mathrm{d}v_1 - \lambda f(x, v), \end{aligned}$$

where $p(x_1, v_1)$ is the joint bivariate normal density

$$\frac{1}{v\sqrt{2\pi s^2}}\exp\left(-\frac{v_1}{v} - \frac{(x_1 - m - \rho_J v_1)^2}{2s^2}\right).$$

This implies that, given jump size v_1 of V_t, the jump size of X_t is normally distributed with mean $m + \rho_J v_1$ and variance s^2. Then the value of option F satisfies the following PIDE (see, for example, Cont and Tankov 2003):

$$F_\tau - \mathrm{Im}\, F + rF = 0.$$

In this case, we need to calculate a two-dimensional integral term, so we can apply IFGT by expressing the integral as follows:

$$\begin{aligned} \frac{\lambda}{v\sqrt{2\pi s^2}}\int_0^{+\infty}\int_{-\infty}^{+\infty} \exp\left(-\frac{v_1}{v}\right) f(x_1, v_1) \\ \times \exp\left(-\frac{(x'(x, v) - (x'(x_1, v_1) + m))^2}{2s^2}\right)\mathrm{d}x_1\,\mathrm{d}v_1, \end{aligned}$$

where $x'(x, v) := x - \rho_J v$. But the value of s is expected to be small (for example, calibration gives $s = 0.0001$ in Duffie *et al* (2000)). In this case, the simple summation of some integrands around x and v seems to be sufficient, because the exponential function $\exp(-(x'(x, v) - x'(x_1, v_1) + m)^2/2s^2)$ decays rapidly for small s.

APPENDIX B. KOU'S JUMP-DIFFUSION MODEL

We leave the details of Kou's jump-diffusion model to Kou (2002), but to compute option prices under PIDE, it is necessary to compute the following form of integral operator:

$$\begin{aligned}\int_{-\infty}^{+\infty} & v(x+z,\tau)f(z)\,dz \\ &= A_1\int_{-\infty}^{0} v(x+z,\tau)e^{G_1 z}\,dz + A_2\int_{0}^{+\infty} v(x+z,\tau)e^{-G_2 z}\,dz \\ &= A_1 e^{-G_1 x}\int_{-\infty}^{x} v(s,\tau)e^{G_1 s}\,ds + A_2 e^{G_2 x}\int_{x}^{+\infty} v(s,\tau)e^{-G_2 s}\,ds,\end{aligned}$$

where A_1, A_2, G_1 and G_2 are scalar constants. It is interesting to consider the possibility of applying IFGT to this type of kernel, but, even without the help of IFGT, we can easily achieve a linear computational cost. One method is proposed in Carr and Mayo (2007) and, according to their paper, the second term is expressed as follows:

$$\begin{aligned}A_2 e^{G_2 x}\int_{x}^{+\infty} & v(s,\tau)e^{-G_2 s}ds \\ &= A_2 e^{G_2 x}\int_{0}^{+\infty} v(s,\tau)e^{-G_2 s}\,ds - A_2 e^{G_2 x}\int_{0}^{x} v(s,\tau)e^{-G_2 s}\,ds.\end{aligned}$$

The second integral can be computed recursively as follows, to achieve a linear computational cost:

$$\int_{0}^{x_{i+1}} v(s,\tau)e^{-G_2 s}\,ds = \int_{0}^{x_i} v(s,\tau)e^{-G_2 s}\,ds + v(x_{i+1},\tau)e^{-G_2 x_{i+1}} w_{i+1}\Delta s,$$

where the w_i terms are weights that depend on the numerical evaluation of the integral.

APPENDIX C. APPLICATION OF IMPROVED FAST GAUSS TRANSFORM TO THE TWO-DIMENSIONAL MERTON MODEL

First, we assume $\rho_c \geq 0$. Then

$$\begin{aligned}&\frac{(s_1-(x_1+m_{1,c}))^2}{\nu_{1,c}^2} + \frac{(s_2-(x_2+m_{2,c}))^2}{\nu_{2,c}^2} \\ &\quad - 2\rho_c\frac{(s_1-(x_1+m_{1,c}))(s_2-(x_2+m_{2,c}))}{\nu_{1,c}\nu_{2,c}}\end{aligned}$$

$$
\begin{aligned}
&= \frac{1-\rho_c}{\nu_{1,c}^2}[s_1 - (x_1 + m_{1,c})]^2 + \frac{1-\rho_c}{\nu_{2,c}^2}[s_2 - (x_2 + m_{2,c})]^2 \\
&\quad + \rho_c\left[\frac{(s_1 - (x_1 + m_{1,c}))}{\nu_{1,c}} - \frac{(s_2 - (x_2 + m_{2,c}))}{\nu_{2,c}}\right]^2 \\
&= \frac{1-\rho_c}{\nu_{1,c}^2}[s_1 - (x_1 + m_{1,c})]^2 \\
&\quad + \frac{1-\rho_c}{\nu_{1,c}^2}\left[\frac{\nu_{1,c}}{\nu_{2,c}}s_2 - \left(\frac{\nu_{1,c}}{\nu_{2,c}}x_2 + \frac{\nu_{1,c}}{\nu_{2,c}}m_{2,c}\right)\right]^2 \\
&\quad + \frac{1-\rho_c}{\nu_{1,c}^2}\left[\nu_{1,c}\sqrt{\frac{\rho_c}{1-\rho_c}}\frac{(s_1 - (x_1 + m_{1,c}))}{\nu_{1,c}}\right. \\
&\qquad\qquad \left. - \nu_{1,c}\sqrt{\frac{\rho_c}{1-\rho_c}}\frac{(s_2 - (x_2 + m_{2,c}))}{\nu_{2,c}}\right]^2 \\
&:= \frac{1-\rho_c}{\nu_{1,c}^2}[s_1 - (x_1 + m_{1,c})]^2 + \frac{1-\rho_c}{\nu_{1,c}^2}[s_2' - (x_2' + m_{2,c}')]^2 \\
&\quad + \frac{1-\rho_c}{\nu_{1,c}^2}[s_3'(s_1, s_2) - (x_3'(x_1, x_2) + m_{3,c}')]^2,
\end{aligned}
$$

and the integrand is expressed as

$$
v(s_1, s_2, \tau)\exp\left(-\frac{\begin{array}{r}(s_1 - (x_1 + m_{1,c}))^2 + (s_2' - (x_2' + m_{2,c}'))^2 \\ +(s_3'(s_1, s_2) - (x_3'(x_1, x_2) + m_{3,c}'))^2\end{array}}{2\nu_{1,c}^2(1+\rho_c)}\right),
$$

where

$$
x_2' := \frac{\nu_{1,c}}{\nu_{2,c}}x_2, \qquad s_2' := \frac{\nu_{1,c}}{\nu_{2,c}}s_2, \qquad m_{2,c}' := \frac{\nu_{1,c}}{\nu_{2,c}}m_{2,c},
$$

$$
x_3'(x_1, x_2) := \sqrt{\frac{\rho_c}{1-\rho_c}}x_1 - \sqrt{\frac{\rho_c}{1-\rho_c}}x_2',
$$

$$
s_3'(s_1, s_2) := \sqrt{\frac{\rho_c}{1-\rho_c}}s_1 - \sqrt{\frac{\rho_c}{1-\rho_c}}s_2',
$$

$$
m_{3,c}' := \sqrt{\frac{\rho_c}{1-\rho_c}}m_{1,c} - \sqrt{\frac{\rho_c}{1-\rho_c}}m_{2,c}'.
$$

Similarly for $\rho_c < 0$, the integrand is expressed as

$$v(s_1, s_2, \tau) \exp\left(- \frac{(s_1 - (x_1 + m_{1,c}))^2 + (s_2' - (x_2' + m_{2,c}'))^2 + (s_3'(s_1, s_2) - (x_3'(x_1, x_2) + m_{3,c}'))^2}{2v_{1,c}^2(1 - \rho_c)} \right),$$

where

$$x_3'(x_1, x_2) := \sqrt{\frac{|\rho_c|}{1 - |\rho_c|}} x_1 + \sqrt{\frac{|\rho_c|}{1 - |\rho_c|}} x_2',$$

$$s_3'(s_1, s_2) := \sqrt{\frac{|\rho_c|}{1 - |\rho_c|}} s_1 + \sqrt{\frac{|\rho_c|}{1 - |\rho_c|}} s_2',$$

$$m_{3,c}' := \sqrt{\frac{|\rho_c|}{1 - |\rho_c|}} m_{1,c} + \sqrt{\frac{|\rho_c|}{1 - |\rho_c|}} m_{2,c}'.$$

Combining the two cases, the integral is expressed as

$$v(s_1, s_2, \tau) \exp\left(- \frac{(s_1 - (x_1 + m_{1,c}))^2 + (s_2' - (x_2' + m_{2,c}'))^2 + (s_3'(s_1, s_2) - (x_3'(x_1, x_2) + m_{3,c}'))^2}{2v_{1,c}^2(1 + |\rho_c|)} \right),$$

where

$$x_2' := \frac{v_{1,c}}{v_{2,c}} x_2, \qquad s_2{}' := \frac{v_{1,c}}{v_{2,c}} s_2, \qquad m_{2,c}' := \frac{v_{1,c}}{v_{2,c}} m_{2,c},$$

$$x_3'(x_1, x_2) := \sqrt{\frac{|\rho_c|}{1 - |\rho_c|}} x_1 - \operatorname{sgn}(\rho_c) \sqrt{\frac{|\rho_c|}{1 - |\rho_c|}} x_2',$$

$$s_3'(s_1, s_2) := \sqrt{\frac{|\rho_c|}{1 - |\rho_c|}} s_1 - \operatorname{sgn}(\rho_c) \sqrt{\frac{|\rho_c|}{1 - |\rho_c|}} s_2',$$

$$m_{3,c}' := \sqrt{\frac{|\rho_c|}{1 - |\rho_c|}} m_{1,c} - \operatorname{sgn}(\rho_c) \sqrt{\frac{|\rho_c|}{1 - |\rho_c|}} m_{2,c}'.$$

REFERENCES

Andersen, L., and Andreasen, J. (2000). Jump-diffusion processes: volatility smile fitting and numerical methods for option pricing. *Review of Derivatives Research* **4**, 231–262.

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

Broadie, M., and Yamamoto, Y. (2005). A double-exponential fast Gauss transform algorithm for pricing discrete path-dependent options. *Operations Research*, **53**, 764–779.

Carr, P., and Mayo, A. (2007). On the numerical evaluation of option prices in jump diffusion processes. *European Journal of Finance* **13**, 353–372.

Cont, R., and Tankov, P. (2003). *Financial Modelling with Jump Processes*. Chapman & Hall/CRC, Boca Raton, FL.

Craig, I. J. D., and Sneyd, A. D. (1988). An alternating-direction implicit scheme for parabolic equations with mixed derivatives. *Computers and Mathematics with Applications* **16**, 341–350.

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

d'Halluin, Y., Forsyth, P. A., and Vertzal, K. R. (2005). Robust numerical methods for contingent claims under jump diffusion processes. *IMA Journal of Numerical Analysis* **25**, 87–112.

Douglas, J., and Rachford, H. H. (1956). On the numerical solution of heat conduct problems in two and three space variables. *Transactions of the American Mathematical Society* **82**, 421–439.

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

Feng, L., and Linetsky, V. (2008). Pricing options in jump-diffusion models: an extrapolation approach. *Operations Research* **56**, 304–325.

Fong, W., and Darve, E. (2009). The black-box fast multipole method. *Journal of Computational Physics* **228**, 8712–8725.

Glasserman, P. (2004). *Monte Carlo Methods in Financial Engineering*. Springer.

Gonzalez, T. (1985). Clustering to minimize the maximum intercluster distance. *Theoretical Computer Science* **38**, 293–306.

Greengard, L., and Rokhlin, V. (1987). A fast algorithm for particle simulations. *Journal of Computational Physics* **73**, 325–348.

Greengard, L., and Strain, J. (1991). The fast Gauss transform. *SIAM Journal on Scientific and Statistical Computing* **12**, 79–94.

Huang, Z., and Kou, S. G. (2006). First passage times and analytical solutions for options on two assets with jump risk. Preprint, Columbia University.

Hundsdorfer, W. (2002). Accuracy and stability of splitting with stabilizing corrections. *Applied Numerical Mathematics* **42**, 213–233.

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

Kou, S. G., and Wang, H. (2003). First passage times of a jump diffusion process. *Advances in Applied Probability* **35**, 504–531.

Kou, S. G., and Wang, H. (2004). Option pricing under a double exponential jump diffusion model. *Management Science* **50**, 1178–1192.

Merton, R. C. (1976). Option pricing when underlying stock returns are discontinuous. *Journal of Financial Economics* **3**, 125–144.

Raykar, V. C., Yang, C., Duraiswami, R., and Gumerov, N. (2005). Fast computation of sums of Gaussians in high dimensions. Technical Report CS-TR-4767, Department of Computer Science, University of Maryland.

Tavella, D., and Randall, C. (2000). *Pricing Financial Instruments: The Finite Difference Method.* Wiley.

Yang, C., Duraiswami, R., and Davis, L. (2005). Efficient kernel machines using the improved fast Gauss transform. In *Advances in Neural Information Processing Systems*, Saul, L. K., Weiss, Y., and, Bottoupp, L. (eds), Vol. 17, pp. 1561–1568. MIT Press, Cambridge, MA.

Ying, L., Biros, G., and Zorin, D. (2004). A kernel-independent adaptive fast multipole method in two and three dimensions. *Journal of Computational Physics* **196**, 591–626.