

Application of the Fast Gauss Transform to Option Pricing

Mark Broadie and Yusaku Yamamoto *

January 18, 2002

Abstract

In many of the numerical methods for pricing American options based on the dynamic programming approach, the most computationally intensive part can be formulated as summation of Gaussians. Though this operation usually requires $O(MN)$ work when there are M summations to compute and the number of terms appearing in each summation is N , we can reduce the amount of work to $O(M + N)$ by using a technique called fast Gauss transform. In this paper, we apply this technique to the multinomial method and the stochastic mesh method, and show by numerical experiments how it can speed up these methods considerably, both for the Black-Scholes model and Merton's lognormal jump-diffusion model. We also propose some extensions to apply the fast Gauss transform to Kou's double-exponential jump-diffusion model and Heston's stochastic volatility model.

1 Introduction

Many options traded in the market have American features, and it is therefore optimal to exercise before maturity. The rational price of such options can be calculated as a discounted expectation value (under the risk-neutral measure [5]) of the payoff under the optimal (adapted) exercise strategy, that is

$$Q_0(S_0) = \sup_{\tau} e^{-r\tau} E_0[h_{\tau}(S_{\tau})], \quad (1)$$

where S_t is the stock price at time t , $h_t(S_t)$ is the payoff from exercise at time t , and τ is a stopping time. However, unlike European options, there are no explicit formulas for the option price $Q_0(S_0)$ except for some special cases such

*Mark Broadie is at the Columbia Graduate School of Business, 3022 Broadway, New York, NY, 10027-6902, USA. Yusaku Yamamoto is at the Columbia Graduate School of Business and the Center for Applied Probability. The authors are grateful to Professor Steve Kou for many useful suggestions.

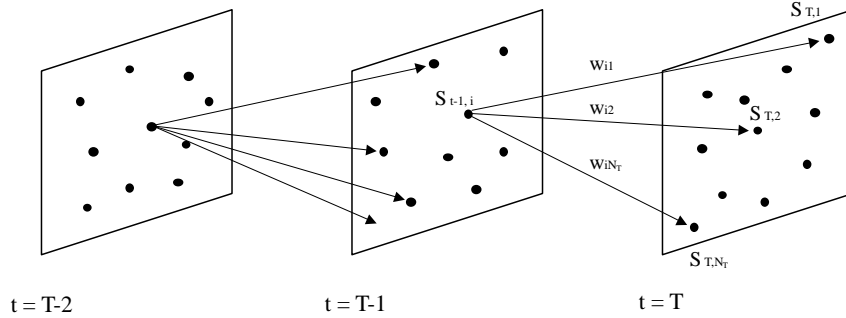


Figure 1: Calculation of the continuation value

as the perpetual American option, and one has to resort to numerical methods for pricing.

Many of the numerical methods for American option pricing use a dynamic programming approach. In this approach, we discretize the time, and starting from the option value at maturity $Q_T(S_T) = h_T(S_T)$, compute the option value at each time step by working backwards. Specifically, the option price at time t and with asset price S_t is computed as the maximum of the immediate exercise value and the continuation value as follows:

$$Q_t(S_t) = \max\{h_t(S_t), e^{-r\Delta t} E_t[Q_{t+1}(S_{t+1})]\}. \quad (2)$$

In the actual algorithms, we also discretize the space as $\{S_{t,1}, S_{t,2}, \dots, S_{t,N_t}\}$, as shown in Fig. 1, and approximate the continuation value at point $(t, S_{t,i})$ as follows:

$$E[Q_{t+1}(S_{t+1})|S_{t,i}] \cong \sum_{j=1}^{N_{t+1}} w_{ij} Q_{t+1,j}, \quad i = 1, 2, \dots, N_t. \quad (3)$$

Here, $Q_{t+1,j}$ denotes the option value at time $t+1$ and asset price $S_{t+1,j}$, and w_{ij} is the weight of $Q_{t+1,j}$ used in the evaluation of the continuation value at $S_{t,i}$. Some of the numerical procedures that can be cast into this framework are binomial and multinomial methods, explicit finite difference methods, and the stochastic mesh method.

Apparently, eq. (3) seems to require $O(N_t N_{t+1})$ computation for each time step, and in fact, in the algorithms listed above, most of the computational efforts are spent to evaluate the expectation value through eq. (3). However, in some cases it can be shown that the matrix w_{ij} has a special structure that enables much faster evaluation of eq. (3). Such a situation arises, for example, when the underlying assets follow the (multi-dimensional) geometric Brownian

motion. In this case, it can be shown that the sum of eq. (3) can be written as sum of Gaussians:

$$G(x_i) = \sum_{j=1}^N q_j \exp \left\{ \frac{(x_i - y_j)^2}{\delta} \right\}, \quad i = 1, 2, \dots, M. \quad (4)$$

Then $G(x_i)$ ($i = 1, \dots, M$) can be evaluated in $O(M + N)$ time using a method called fast Gauss transform, instead of $O(MN)$ time needed for direct evaluation. This technique can also be extended to deal with the lognormal jump-diffusion model. In this paper, we use multinomial type methods and the stochastic mesh method as examples, and show how the fast Gauss transform can improve the efficiency of these methods.

The paper is organized as follows: in section 2 and 3, we describe the algorithms of the multinomial method and the stochastic mesh method, respectively, and show how the computation can be cast into sum of Gaussians. In section 4, we give the basic idea of the fast Gauss transform and describe how the sum in eq. (4) can be calculated in $O(M + N)$ work. Results of numerical experiments can be found in section 5. Section 6 treats two extensions aimed at improving the convergence of the multinomial method for Bermudan options, and applying the fast Gauss transform to the double-exponential jump-diffusion model. Concluding remarks are given in the final section.

2 The multinomial methods

2.1 Multinomial methods for the Black-Scholes model

In this section, we will primarily consider the risk-neutralized Black-Scholes model where the asset price S_t follows the geometric Brownian motion:

$$dS_t = rdt + \sigma dW_t, \quad (5)$$

where W_t is a Wiener process. This can also be written in an integral form:

$$S_t = S_0 \exp \left\{ \left(r - \frac{1}{2} \sigma^2 \right) t + \sigma W_t \right\}. \quad (6)$$

Though we restrict ourselves to a single asset case here, extension of the method to multi-asset cases is quite straightforward.

In a multinomial method, we approximate the Wiener process W_t by a discrete random variable W_{t_i} which is defined at discrete time $t_i = i\Delta t$, and takes discrete value $W_j = j\Delta W$. Following Alford and Webber [1], we require that if W_{t_i} has the value W_j at time t_i , its value at time t_{i+1} takes the values in the set $\{W_{j+k} | k = -b, \dots, b\}$ for some constant b . Then, each node of the tree has $d = 2b + 1$ branches and there are $2bi + 1$ nodes at time t_i , as shown in Fig. 2.

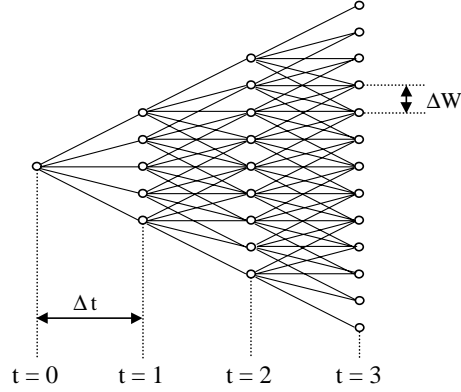


Figure 2: A pentanomial tree ($b=2$)

This is called a multinomial tree of branching order d . We associate each of the branches with branching probability p_i . Let the option value at grid point (i, j) be $Q_{i,j}$. Then, because the approximation W_{t_i} to W_t readily leads to an approximation S_{t_i} to S_t , we can compute conditional expectation of eq. (3) by

$$E[Q_{t_{i+1}} | S_{i,j}] = \sum_{k=-b}^b p_k Q_{i+1, j+k}. \quad (7)$$

To attain high order of convergence in the multinomial methods, one has to choose the probabilities p_i so that W_{t_i} becomes a good approximation of W_t . One of the guidelines for this is given by Heston and Zhou [12], and they show that if the first q moments of W_{t_i} match those of W_t , the multinomial method with M time steps has a convergence rate of $O(M^{\frac{q-1}{2}})$, under the condition that the option payoff is $2q$ times differentiable. One can easily match all the odd moments (which are zero for W_t) by putting $p_{-k} = p_k$ for $k = 1, \dots, b$. To match the first b even moments, and to ensure that the sum of p_i 's is 1, one has to solve the following linear equation:

$$\begin{pmatrix} 1 & 1 & 1 & \dots & 1 \\ 0 & 2 \cdot \Delta z^2 & 2 \cdot (2\Delta z)^2 & \dots & 2 \cdot (b\Delta z)^2 \\ 0 & 2 \cdot \Delta z^4 & 2 \cdot (2\Delta z)^4 & \dots & 2 \cdot (b\Delta z)^4 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 2 \cdot \Delta z^{2b} & 2 \cdot (2\Delta z)^{2b} & \dots & 2 \cdot (b\Delta z)^{2b} \end{pmatrix} = \begin{pmatrix} p_0 \\ p_1 \\ p_2 \\ \vdots \\ p_b \end{pmatrix} \begin{pmatrix} 1 \\ \Delta t \\ 3\Delta t^2 \\ \vdots \\ \frac{(2b)!}{2^b b!} (\Delta t)^b \end{pmatrix} \quad (8)$$

Alford and Webber [1] solve this equation numerically, and confirm that the resulting methods for $b = 3, 7, 11, 15$ and 19 have much higher rate of convergence for European options than the binomial methods.

2.2 Application of the fast Gauss transform

Here, we consider a multinomial method with a very large value of b , and show how it can be cast into a form to take advantage of the fast Gauss transform.

First, let

$$p(x) = \frac{1}{\sqrt{2\pi\Delta t}} \exp\left(-\frac{x^2}{2\Delta t}\right). \quad (9)$$

Then, from the definition of Riemann integral, we have

$$\lim_{\Delta z \rightarrow 0} \sum_{k=-\infty}^{+\infty} (k\Delta z)^{2n} p(k\Delta z) = \int_{-\infty}^{\infty} x^{2n} p(x) dx = \frac{(2n)!}{2^n n!} (\Delta t)^n. \quad (10)$$

Therefore, for sufficiently small Δz ,

$$\sum_{k=-\infty}^{+\infty} (k\Delta z)^{2n} p(k\Delta z) \cong \frac{(2n)!}{2^n n!} (\Delta t)^n. \quad (11)$$

By truncating the infinite sum, we have

$$\sum_{k=-b}^b (k\Delta z)^{2n} p(k\Delta z) \cong \frac{(2n)!}{2^n n!} (\Delta t)^n. \quad (12)$$

This shows that by putting

$$p_k = p(k\Delta z)\Delta z, \quad (13)$$

equation (8) is approximately satisfied. Note that the error introduced by approximation (11) is $O(e^{-c/\Delta z})$, because this is an approximation of the integration of an analytical function over the entire real axis by a trapezoidal rule [18]. Also, the error introduced by (12) decreases exponentially as the number of branches increases. Thus, we can expect the p_k given by eq. (13) is a very accurate approximate solution of eq. (8).

From eqs. (7) and (13), the conditional expectation value can be computed as

$$E[Q_{t_{i+1}} | S_{i,j}] = \frac{\Delta z}{\sqrt{2\pi\Delta t}} \sum_{k=-b}^b Q_{i+1,j+k} \exp\left\{-\frac{1}{2\Delta t}(W_{i,j} - W_{i+1,j+k})^2\right\}. \quad (14)$$

This has the form of sum of Gaussians, and can be computed efficiently by the fast Gauss transform which we will introduce in section 4.

Finally, we point out that the multinomial method based on the fast Gauss transform will be especially useful for Bermudan options, i.e., a variation of American option for which the exercise opportunity is limited to discrete time

points during the life of the option. This is because the method can approximate the underlying asset price process accurately even if the time step is large, thanks to the large number of branches. As a result, we can omit the time steps between the exercise dates. In contrast, in the binomial method, one has to use sufficiently large number of time steps to get an accurate answer, even if the number of exercise dates is small.

2.3 Extension to the lognormal jump-diffusion model

We can extend the method described above to deal with the lognormal jump-diffusion model introduced by Merton [17]. In this model the asset price follows the equation:

$$S_{t+\Delta t} = S_t \exp \left\{ \left(r - \frac{1}{2} \sigma^2 \right) \Delta t + \sigma \sqrt{\Delta t} z_0 + \sum_{i=1}^{N_t^P(\Delta t)} (\delta z_i - \kappa) \right\}, \quad (15)$$

where $N_t^P(\Delta t)$ is the number of jumps between time t and $t + \Delta t$, which follows a Poisson process with intensity λ , and z_i are independent and follow the standard normal distribution $N(0, 1)$. κ and δ are constants that determine the mean and standard deviation of the jumps, respectively. In this model, the market becomes incomplete due to the existence of jumps, and the standard argument for option pricing based on the replicating portfolio is no longer valid. However, under the assumption that jump risk is diversifiable, Merton [17] shows that the price can be written as

$$Q_0(S_0) = e^{-rT} E_0[h_T(S_T)]. \quad (16)$$

Because this has the same form as the formula in the complete market case, the incompleteness of the market causes little difficulty from the computational point of view.

To apply our method to this model, we introduce a change of variables and work with

$$x_t = \log S_t - \left(r - \frac{1}{2} \sigma^2 \right) t, \quad (17)$$

which satisfies the equation

$$x_{t+\Delta t} = x_t + \sigma \sqrt{\Delta t} z_0 + \sum_{i=1}^{N_t^P(\Delta t)} (\delta z_i - \kappa). \quad (18)$$

If we fix the number of jumps between time t and $t + \Delta t$ to n , we have

$$\begin{aligned} x_{t+\Delta t} &= x_t + \sigma \sqrt{\Delta t} z_0 + \sum_{i=1}^n (\delta z_i - \kappa) \\ &\sim N(x_t - n\kappa, \sigma^2 \Delta t + n\delta^2), \end{aligned} \quad (19)$$

because the sum of Gaussian random variables is again a Gaussian random variable. We write the conditional probability density function of $x_{t+\Delta t}$ as

$$\begin{aligned} p^{(n)}(x_{t+\Delta t}|x_t) &\equiv p(x_{t+\Delta t}|x_t, N_t^P(\Delta t) = n) \\ &= \frac{1}{\sqrt{2\pi}\sigma_n} \exp\left\{-\frac{1}{2\sigma_n^2}(x_{t+\Delta t} - x_t - \mu_n)^2\right\}, \end{aligned} \quad (20)$$

where

$$\sigma_n^2 = \sigma\Delta t + n\delta^2, \quad (21)$$

$$\mu_n = -n\kappa. \quad (22)$$

Then, as we have shown in the previous subsection, the expectation value of the option price conditioned on the current asset price and $N_t^P(\Delta t) = n$ can be approximated as

$$E[Q_{t_{i+1}}|S_{i,j}, N_t^P(\Delta t) = n] \cong \Delta z \sum_{k=-b}^b Q_{i+1,j+k} p^{(n)}(x_{t+\Delta t}|x_t). \quad (23)$$

The sum on the right hand side can be calculated using the fast Gauss transform. Finally, the expectation value of the option price is calculated by

$$\begin{aligned} E[Q_{t_{i+1}}|S_{i,j}] &= \sum_{n=0}^{\infty} Pr(N_t^P(\Delta t) = n) E[Q_{t_{i+1}}|S_{i,j}, N_t^P(\Delta t) = n] \\ &= \sum_{n=0}^{\infty} e^{-\lambda\Delta t} \frac{(\lambda\Delta t)^n}{n!} E[Q_{t_{i+1}}|S_{i,j}, N_t^P(\Delta t) = n], \end{aligned} \quad (24)$$

where for numerical computations the infinite sum can be truncated at a sufficiently large value of n . As we shall see in subsection 6.1, this method can be extended to the double-exponential jump-diffusion model proposed by Kou [14].

3 The stochastic mesh method

3.1 The basic algorithm

As another example of DP-based American option pricing methods that can be sped up by the fast Gauss transform, we take up the stochastic mesh method proposed by Broadie and Glasserman [4]. This method is based on Monte Carlo simulation, and computes the expectation value in eq. (3) using randomly distributed sample points at time $t + 1$. Unlike many other pricing algorithms based on Monte Carlo methods, it has the advantage that it can estimate the upper bound on the American option price, and has successfully been applied to the pricing of high dimensional American options. Also, several extensions to improve the convergence and parallel implementations have been studied.

Assume that there are n assets and the dynamics of the price vector S_t is given in the form of the conditional probability density function:

$$Pr(S_{t+\Delta t}|S_t) = f_t^S(S_{t+\Delta t}|S_t). \quad (25)$$

In an implementation of the stochastic mesh method using so-called average mesh density function [4], we first discretize the time and generate b independent sample paths of the asset prices, as in the usual Monte Carlo approach. Let the vector of prices on path i and time step t be $S_{t,i}$. Then the option value at maturity ($t = T$) for each path is given by

$$Q_T(S_{T,i}) = h_T(S_{T,i}), \quad (26)$$

where h_t is the payoff function. Next, we calculate the continuation value on each path at time $t = T - 1$ by estimating the conditional expectation value $E[Q_T(S_T)|S_{T-1,i}]$, using the sample points at $t = T$. However, this cannot be done by taking a simple average, because although one needs a set of sample points that follow the conditional distribution function $f_{T-1}^S(S_T|S_{T-1,i})$ in order to calculate $E[Q_T(S_T)|S_{T-1,i}]$, the actual sample points follow the distribution:

$$g_T^S(S_T) = \frac{1}{b} \sum_{i=1}^b f_{T-1}^S(S_T|S_{T-1,i}) \quad (27)$$

from the construction of the sample paths.

To overcome this difficulty, the stochastic mesh method uses the following relationship:

$$\begin{aligned} E[Q_t(S_t)|S_{t-1,i}] &= \int Q_t(u) f_{t-1}^S(u|S_{t-1,i}) du \\ &= \int Q_t(u) \frac{f_{t-1}^S(u|S_{t-1,i})}{g_t^S(u)} g_t^S(u) du \\ &= E \left[Q_t(S_{t,j}) \frac{f_{t-1}^S(S_{t,j}|S_{t-1,i})}{g_t^S(S_{t,j})} \right] \end{aligned} \quad (28)$$

The last expression is an expectation value under the density function $g_S^{(t-1)}$ and can be calculated using the sample points at $t = T$ as

$$E \left[Q_t(S_{t,j}) \frac{f_{t-1}^S(S_{t,j}|S_{t-1,i})}{g_t^S(S_{t,j})} \right] = \sum_{j=1}^b Q_{t,j} w_{t,ij}, \quad (29)$$

where $w_{t,ij}$ is a weight matrix defined by

$$\begin{aligned} w_{t,ij} &= \frac{f_{t-1}^S(S_{t,j}|S_{t-1,i})}{g_t^S(S_{t,j})} \\ &= \frac{f_{t-1}^S(S_{t,j}|S_{t-1,i})}{\sum_{i'=1}^b f_{t-1}^S(S_{t,j}|S_{t-1,i'})}. \end{aligned} \quad (30)$$

Apparently, the formation of and multiplication by the weight matrix needs $O(b^2)$ computation, and this is the most time-consuming part in the stochastic mesh method.

3.2 Improvement of efficiency by the fast Gauss transform

Prior to applying the fast Gauss transform to the stochastic mesh method, we first investigate the effect of change of variables to the calculation. Suppose we change the variable from S_t to y_t , which is again an n -dimensional vector. Then, the conditional probability density function of y_t is related to that of S_t by

$$f_{t-1}^S(S_t|S_{t-1}) = f_{t-1}^y(y_t|y_{t-1}) \left| \frac{\partial y_t}{\partial S_t} \right|, \quad (31)$$

where $\left| \frac{\partial y_t}{\partial S_t} \right|$ is the Jacobian matrix. By substituting this into the expression of $w_{t,ij}$, we have

$$\begin{aligned} w_{ij,t} &= \frac{f_{t-1}^S(S_{t,j}|S_{t-1,i})}{\sum_{i'=1}^b f_{t-1}^S(S_{t,j}|S_{t-1,i'})} \\ &= \frac{f_{t-1}^y(y_{t,j}|y_{t-1,i}) \left| \frac{\partial y_t}{\partial S_t} \right|_{S_t=S_{t,j}}}{\sum_{i'=1}^b f_{t-1}^y(y_{t,j}|y_{t-1,i'}) \left| \frac{\partial y_t}{\partial S_t} \right|_{S_t=S_{t,j}}} \\ &= \frac{f_{t-1}^y(y_{t,j}|y_{t-1,i})}{\sum_{i'=1}^b f_{t-1}^y(y_{t,j}|y_{t-1,i'})}. \end{aligned} \quad (32)$$

Thus we have established that the weight matrix in the stochastic mesh method (using the average mesh density function) is invariant under change of variables. So we can choose any convenient representation of the state variables to compute the weight matrix and the sum in eq. (29).

We now turn our attention to the multi-asset Black-Scholes model:

$$\begin{aligned} dS_m &= r_m S_m dt + \sigma_m S_m dW_m \\ dW_m dW_l &= \rho_{ml} dt \quad (m \neq l), \end{aligned} \quad (33)$$

where S_m is the price of the m -th asset, and dW_m is a Wiener process. By changing the variables from S_m to

$$x_m = \log S_m - \left(r_m - \frac{1}{2} \sigma_m^2 \right) t, \quad (34)$$

we get a vector variable $x = (x_1, \dots, x_n)^t$ which satisfies

$$\begin{aligned} dx dx^t &= \Sigma R \Sigma dt, \\ \Sigma &= \text{diag}(\sigma_1, \dots, \sigma_n), \\ R &= (\rho_{ij}). \end{aligned} \quad (35)$$

By further introducing a new change of variables using the Cholesky decomposition $R = LL^t$ as

$$y = L^{-1}\Sigma^{-1}x, \quad (36)$$

we finally have the vector y of n independent Wiener process which satisfies

$$dydy^t = Idt. \quad (37)$$

For this new variable, the conditional probability density function can be seen to be

$$f_{t-1}^y(y_1, \dots, y_n) = \prod_{m=1}^n \frac{1}{\sqrt{2\pi\Delta t}} \exp\left\{-\frac{1}{2\Delta t}(y_{t-1,m} - y_m)^2\right\}, \quad (38)$$

and multiplication by the weight matrix can be done using the fast Gauss transform.

4 The fast Gauss transform

4.1 The basic idea

In this section, we describe the basic idea of the fast Gauss transform introduced by Greengard and Strain [9] [19] [10], and show how it can compute the sum of eq. (4) in $O(M + N)$ time. Though we limit ourselves to the one-dimensional case here, the algorithm can be extended to higher dimensional cases. For details of the extension, as well as for more comprehensive description of the algorithm including error analysis, consult [9].

The basis of the fast Gauss transform is the following expansion of the Gaussian in terms of Hermite functions:

$$e^{-(x-y)^2} = \sum_{n=0}^{\infty} \frac{y^n}{n!} h_n(x), \quad (39)$$

where the Hermite function $h_n(x)$ is defined by

$$h_n(x) = (-1)^n \left(\frac{d}{dx}\right)^n e^{-x^2}. \quad (40)$$

It is known that this expansion converges very quickly, and truncation at $n \sim 10$ is sufficient to achieve relative error of 10^{-6} when $|x| < 1/2$.

For the fast Gauss transform, we use a shifted and scaled version of this expansion, namely,

$$e^{-(x_i - y_j)^2/\delta} = \sum_{n=0}^{\infty} \frac{1}{n!} \left(\frac{y_j - y_0}{\sqrt{\delta}}\right)^n h_n\left(\frac{x_i - y_0}{\sqrt{\delta}}\right). \quad (41)$$

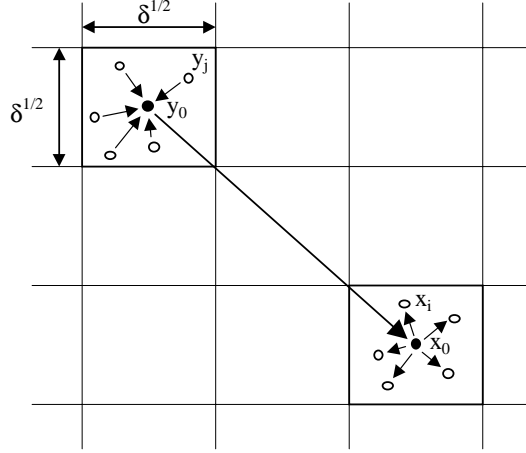


Figure 3: The source and target boxes

The Hermite function appearing in the right hand side of this expression can further be expanded, and we finally obtain

$$e^{-(x_i - y_j)^2 / \delta} = \sum_{m=0}^{\infty} \sum_{n=0}^{\infty} \frac{1}{m!} \frac{1}{n!} \left(\frac{y_j - y_0}{\sqrt{\delta}}\right)^n h_{n+m} \left(\frac{x_0 - y_0}{\sqrt{\delta}}\right) \left(\frac{x_i - x_0}{\sqrt{\delta}}\right)^m \quad (42)$$

Suppose that we want to calculate eq. (4) for $\{x_i\}_{i=1}^M$ and $\{y_j\}_{j=1}^N$. We first consider a special case where all the target points $\{x_i\}$ are in a box with center x_0 and side length $\sqrt{\delta}$, and all the source points $\{y_j\}$ are in another box with center y_0 and side length $\sqrt{\delta}$, as shown in Figure 3. Then the above expansion converges quickly by truncating the sums in eq. (42) at some n_{max} , and the sum of eq. (4) can be written as

$$\begin{aligned} G(x_i) &\sim \sum_{j=1}^N q_j \sum_{m=0}^{n_{max}} \sum_{n=0}^{n_{max}} \frac{1}{m!} \frac{1}{n!} \left(\frac{y_j - y_0}{\sqrt{\delta}}\right)^n h_{n+m} \left(\frac{x_0 - y_0}{\sqrt{\delta}}\right) \left(\frac{x_i - x_0}{\sqrt{\delta}}\right)^m \\ &= \sum_{m=0}^{n_{max}} \frac{1}{m!} \left(\frac{x_i - x_0}{\sqrt{\delta}}\right)^m \\ &\quad \times \left\{ \sum_{n=0}^{n_{max}} h_{n+m} \left(\frac{x_0 - y_0}{\sqrt{\delta}}\right) \left\{ \frac{1}{n!} \sum_{j=1}^N q_j \left(\frac{y_j - y_0}{\sqrt{\delta}}\right)^n \right\} \right\} \end{aligned} \quad (43)$$

This expression shows that the computation of $G(x_i)$ can be divided into three steps:

1. Compute $\frac{1}{n!} \sum_{j=1}^N q_j \left(\frac{y_j - y_0}{\sqrt{\delta}}\right)^n$.
2. Multiply the result by $h_{n+m} \left(\frac{x_0 - y_0}{\sqrt{\delta}}\right)$ and sum over n .
3. Multiply the result by $\frac{1}{m!} \left(\frac{x_i - x_0}{\sqrt{\delta}}\right)^m$ and sum over m .

Step 1 and 3 require $O(N)$ and $O(M)$ computational effort, respectively, while step (2) can be done in constant time that does not depend either on M or N .

In a general case, we divide the space into boxes of side length $\sqrt{\delta}$, and apply the above method to each of the possible pairs of a source box and a target box. Because each x_i and y_j belong to only one box, the total work for step (1) and (3) is still $O(M)$ and $O(N)$, respectively, while step (2) needs work proportional to $O(N_{box}^2)$, where N_{box} is the number of the boxes.

4.2 Implementation details

In the actual algorithm, we can improve the efficiency of the above method by adopting some modifications, as we will state below.

First, because the Gaussian function $e^{-(x-y)^2}$ decreases very rapidly when the distance between x and y becomes large, the interaction between distant boxes in step 2 can be omitted. It is shown in [9] that considering only $2n + 1$ nearest boxes with $n = 8$ is sufficient for double precision accuracy.

Second, if both the source and target boxes contain only a small number of points, it may be faster to evaluate the Gaussian directly than to use the Hermite expansion. Or, if the target box contains only a few points, it may be faster to expand the Gaussian only with respect to y_j , and use eq. (41) to evaluate $G(x_i)$. Greengard and Strain [9] recommend to set some threshold for the number of points in the box, and to use different evaluation methods according to whether the number of points in the source and target boxes is above or below the threshold.

Finally, it is possible to use alternative basis functions to expand the Gaussian. Greengard and Sun [10] propose to use an expansion formula based on the Fourier transform of the Gaussian, instead of the Hermite expansion, and show that it can reduce the work required in step 2 drastically. This choice seems preferable when the number of points is moderate and the computation in step 2 occupies a considerable part of the total work.

5 Numerical experiments

5.1 Multinomial methods

We implemented the multinomial method using the fast Gauss transform for European and Bermudan options under the Black-Scholes model, and compared its efficiency with that of the Cox et al.'s binomial method [15]. In all of the

cases below, we used a closed form solution by the Black-Scholes formula at the penultimate time step, as suggested in [3]. In the case of European option, this corresponds to replacing the actual payoff function at maturity with an analytical payoff function at the penultimate time step, and is effective in removing the oscillatory convergence characteristic of the binomial and multinomial type methods.

All the experiments were done on a Pentium II PC with Red-Hat Linux using gnu C++ compiler. Throughout this section, we denote the initial asset price, the strike price, and the option price at $t = 0$ by S_0 , K , and Q_0 , respectively. We also denote the riskless annual interest rate, dividend rate, and volatility by r , δ , and σ , respectively. Finally, we use T (in years) for option maturity and d for the number of exercise dates.

5.1.1 European option on a single asset

We show the results for a European call option in Fig. fig:binBS. Here, $S_0 = K = 100$, $r = 0.03$, $\delta = 0.07$, $\sigma = 0.20$, and $T = 0.5$. The exact price by the Black-Scholes formula is 4.57776128. We calculated the option value using a binomial method with time steps M from 500 to 10000 (in increments of 500), and a multinomial method with $2b + 1$ branches, where $b = 10$ to 150 (in increments of 10). In the multinomial method, the number of time steps was fixed to 10, because it can guarantee convergence without increasing the number of time steps when the number of branches is increased.

In Fig. fig:binBS, the vertical axis and the horizontal axis represent the error in the calculated option price and the computational time, respectively, both in a log scale. As can be clearly seen from the graph, the steepness of the binomial result is $-1/2$, implying that the error decreases as $O(M^{-1})$. On the other hand, the graph of the multinomial method is nearly vertical, and confirms the very high order of convergence theoretically guaranteed when the option payoff function is analytical. Note that the computational time of the multinomial method and the binomial method cross around error $\sim 10^{-4}$, and the former becomes faster when a higher accuracy is needed.

5.1.2 Bermudan option on a single asset

Next we show the results for a Bermudan option in Fig. fig:binjump. The parameters are the same as in the case of the European option described above, and the number of exercise dates is $d = 10$. We adopted the price $Q_0 = 4.757279$ calculated by a binomial method with 500,000 time steps as a reference price against which to compute the error. The time steps of the binomial method is from 1000 to 20,000 (in increments of 1000), and b for the multinomial method is from 10 to 150 (in increments of 10). The number of time steps in the multinomial method was set equal to the number of exercise dates.

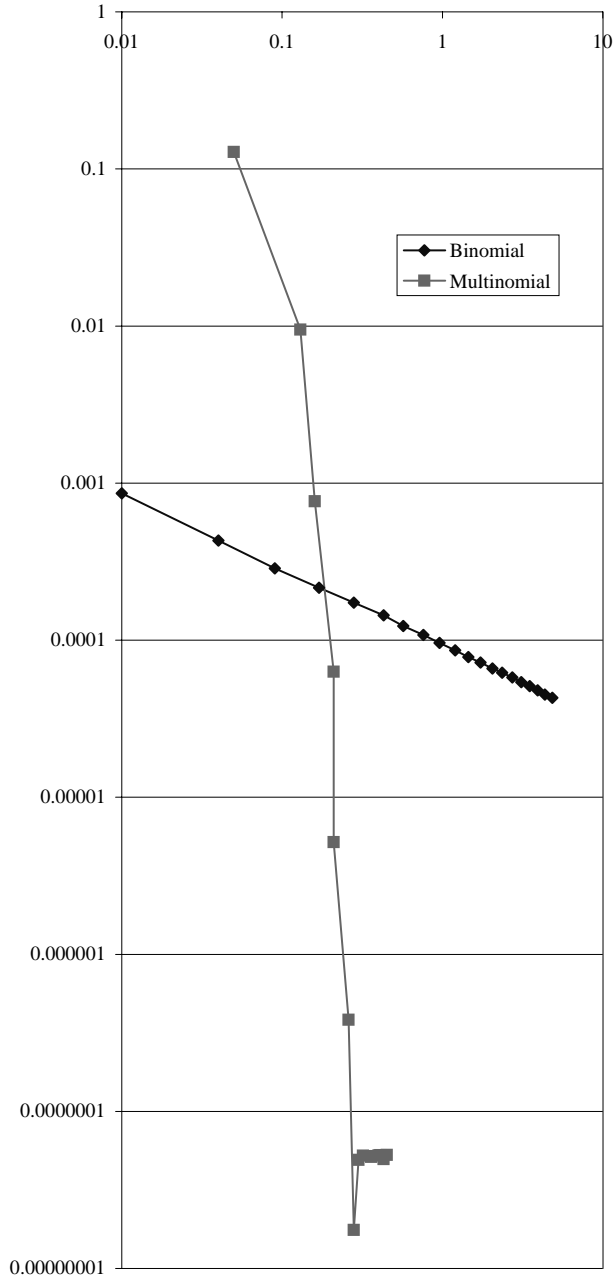


Figure 4: European option price by binomial and multinomial methods

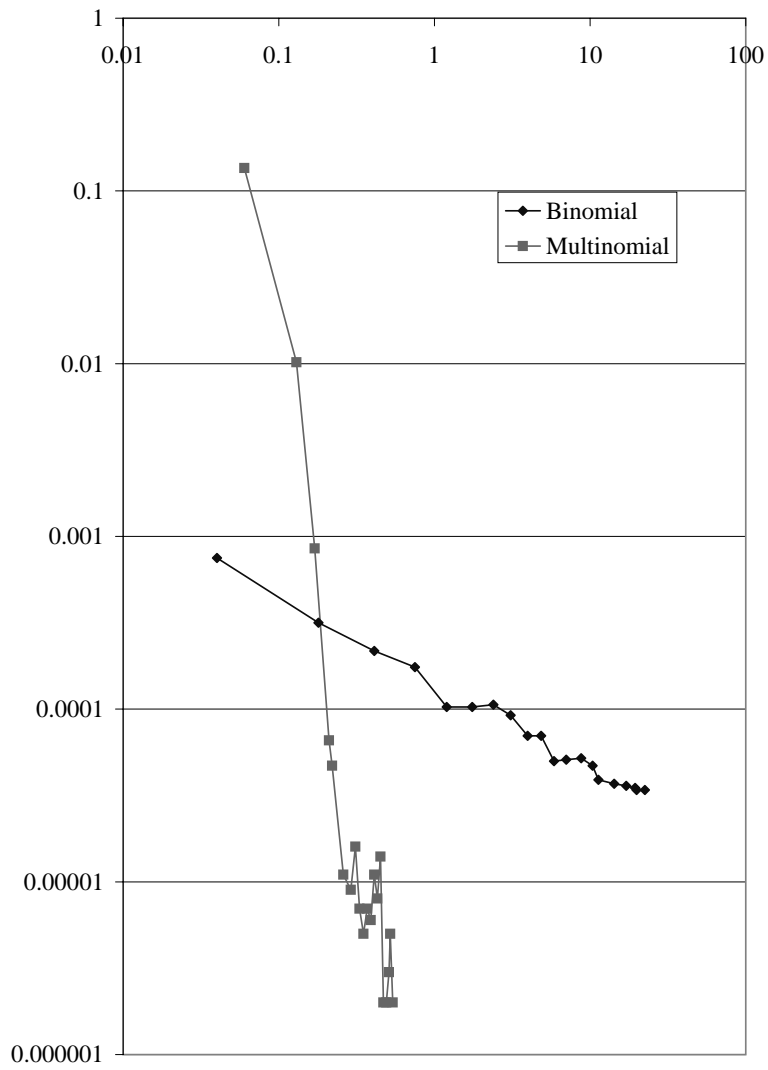


Figure 5: Bermudan option price by binomial and multinomial methods

In this case, again, the multinomial method is more efficient than the binomial method when required accuracy is higher than 10^{-4} . However, the level of asymptotic accuracy attained by the multinomial method is lower than in the European option case. This seems to be because the payoff function of the Bermudan option is not analytical, i.e., it has a discontinuity in the second and higher order derivatives at the exercise boundary [15].

5.2 The stochastic mesh method

We also implemented the stochastic mesh method for pricing Bermudan options with and without the fast Gauss transform, and compared the speed and accuracy of the two implementations. The options we chose are Bermudan call options on one, two, and three underlying assets under the Black-Scholes model, and Bermudan put options on a single asset under the lognormal jump-diffusion model. The environment of the experiments is the same as described in the previous subsection, and we use the same notation.

5.2.1 Bermudan options under the Black-Scholes model

The options we used in our experiment are as follows:

- A Bermudan call option on a single underlying asset, with $S_0 = K = 100$, $r = 0.05$, $\delta = 0.10$, $\sigma = 0.20$, $T = 3.0$, and $d = 2$.
- A Bermudan max call option on two underlying assets, with $S_0 = K = 100$, $r = 0.05$, $\delta = 0.10$, $\sigma = 0.20$, $T = 1.0$, and $d = 3$. The correlation between the two option is $\rho = 0.3$.
- A Bermudan max call option on three underlying assets, with $S_0 = K = 100$, $r = 0.05$, $\delta = 0.10$, $\sigma = 0.20$, and $T = 1.0$. The correlation between the two option is $\rho = 0.3$.

The parameter values used in case 1 are the same as those used in [2], and the option price given there is 7.18. The values in case 2 are the same as in [8], and the price given there is 9.390.

Table 1 shows the results of our computation. For each of the case, we changed the number of sample paths from $b = 1000$ to 100,000. As can be seen from the table, the fast Gauss transform can speed up the stochastic mesh method drastically, making it more than 1300 times faster in the one-dimensional case when $b=10,000$, and about 60 times faster in the two dimensional case when $b=10,000$. The speedup is less striking in the three-dimensional case, but still the new implementation is nearly ten times faster when $b=30,000$ and more than 30 times faster when $b=100,000$. It is also apparent that the computational time is linear in the number of sample paths for the implementation with the fast Gauss transform, while it is quadratic for the conventional

implementation. As for the accuracy, we can say that the fast Gauss transform is sufficiently accurate, for the option prices calculated by the two implementations agree to at least six digits after the decimal point.

Table 1(a) Bermudan call option on a single asset

Number of sample paths	Direct calculation		Fast Gauss transform	
	Price	Time	Price	Time
1,000	7.754283	4.04s	7.754283	0.03s
3,000	7.445551	36.37s	7.445551	0.11s
10,000	7.365219	412.34s	7.365219	0.31s
30,000	7.276099	(Timer overflow)	7.276099	0.96s
100,000	—	—	7.202183	3.38s

Table 1(b) Bermudan max call option on two underlying assets

Number of sample paths	Direct calculation		Fast Gauss transform	
	Price	Time	Price	Time
1,000	10.387124	7.07s	10.387124	2.95s
3,000	9.597424	63.11s	9.597424	5.95s
10,000	9.546258	706.21s	9.546258	11.96s
30,000	—	—	9.438231	21.96s
100,000	—	—	9.337766	49.26s

Table 1(c) Bermudan max call option on three underlying assets

Number of sample paths	Direct calculation		Fast Gauss transform	
	Price	Time	Price	Time
1,000	13.241150	7.21s	13.241150	109.33s
3,000	12.397652	65.01s	12.397652	161.85s
10,000	12.294868	733.77s	12.294868	329.62s
30,000	12.075991	7236.2s	12.075991	778.07s
100,000	—	73377s*	12.015780	2253.76s

* estimated time

5.2.2 Bermudan option price under the lognormal jump-diffusion model

Finally, we calculated the price of a Bermudan put option on a single asset under the lognormal jump-diffusion model. The parameters we used are $S = K = 100$, $r = 0.10$, $\sigma = \sqrt{0.08}$, and $T = 0.5$. The jump parameters introduced in subsection 2.3 are $\lambda = 2.0$, $\delta = 0.2$, and $\kappa = 0.02$. We changed the number of

exercise dates from $d = 1$ (European option) to $d = 2$ and 4. These parameter values are used in [8]. We truncated the infinite sum of eq. (24) at $n = 9$.

The computational results for the two implementations are shown in Table 2. Again, the implementation based on the fast Gauss transform is more than 600 times faster when $b = 3000$, and is also very accurate.

Table 2(a) European put option on a single asset with jumps (d=1)

Number of sample paths	Direct calculation		Fast Gauss transform	
	Price	Time	Price	Time
1,000	8.567110	22.81s	8.567110	0.14s
3,000	8.319821	208.04s	8.319821	0.34s
10,000			8.453335	1.12s
30,000			8.420260	3.67s
100,000			8.375412	12.55s
300,000			8.406095	37.25s

Table 2(b) Bermudan put option on a single asset with jumps (d=2)

Number of sample paths	Direct calculation		Fast Gauss transform	
	Price	Time	Price	Time
1,000	8.717024	45.63s	8.717024	0.24s
3,000	8.755142	416.04s	8.755142	0.62s
10,000			8.712629	2.17s
30,000			8.669887	7.03s
100,000			8.635718	24.09s
300,000			8.612775	72.63s

Table 2(c) Bermudan put option on a single asset with jumps (d=4)

Number of sample paths	Direct calculation		Fast Gauss transform	
	Price	Time	Price	Time
1,000	8.845593	90.37s	8.845593	0.53s
3,000	8.932504	827.06s	8.932504	1.35s
10,000			8.877616	4.33s
30,000			8.740494	13.94s
100,000			8.734955	48.15s
300,000			8.711638	143.78s

6 Extensions to non-Gaussian densities

6.1 Application to the double-exponential jump-diffusion model

Recently, Kou [14] proposed a new jump-diffusion model in which the logarithm of the jump size has a double-exponential distribution instead of the normal distribution. In this model, the asset price at time $t + \Delta t$ can be written as

$$S_{t+\Delta t} = S_t \exp\left\{\left(\mu - \frac{1}{2}\sigma^2\right)\Delta t + \sigma\sqrt{\Delta t}z + \sum_{i=1}^{N_t^P(\Delta t)} X_i\right\}, \quad (44)$$

where $N_t^P(\Delta t)$ is the number of jumps between time t and $t + \Delta t$, which follows a Poisson process with intensity λ , $z \sim N(0, 1)$, and X_i are independent variables that follow a double-exponential distribution:

$$f_X(x) = \frac{1}{2\eta} e^{-|x-\kappa|/\eta}, \quad 0 < \eta < 1. \quad (45)$$

This model has an advantage over the lognormal jump-diffusion model in that it can have both higher peak and heavier tails than normal distribution. So it is worthwhile to consider an extension of our method to this model.

For that purpose, we first change the variable from S_t to

$$u_t = \log S_t - \left(\mu - \frac{1}{2}\sigma^2\right)t, \quad (46)$$

as we did in subsection 2.3. Then u_t satisfies the equation

$$u_{t+\Delta t} = u_t + \sigma\sqrt{\Delta t}z + \sum_{i=1}^{N_t^P(\Delta t)} X_i. \quad (47)$$

To apply our method, we need an explicit form of the conditional probability density function $f_u(u_{t+\Delta t}|u_t)$. As in the case of the lognormal jump-diffusion model, this can be written as a sum over the number of jumps between t and $t + \Delta t$:

$$\begin{aligned} f_u(u_{t+\Delta t}|u_t) &= \sum_{n=0}^{\infty} Pr(N_t^P(\Delta t) = n) f_u(u_{t+\Delta t}|u_t, N_t^P(\Delta t) = n) \\ &= \sum_{n=0}^{\infty} e^{-\lambda\Delta t} \frac{(\lambda\Delta t)^n}{n!} f_u^{(n)}(u_{t+\Delta t} - u_t), \end{aligned} \quad (48)$$

where $f_u^{(n)}(x)$ is a distribution function of sum of $\sigma\sqrt{\Delta t}z$ and n random variables $\{X_i\}_{i=1}^n$ that follow the double-exponential distribution (45).

Kou [14] shows that sum of double-exponential random variables can be decomposed into random sum of exponential random variables as follows:

$$\sum_{i=1}^n X_i - n\kappa = \begin{cases} \sum_{j=1}^{M(n)} \xi_j & \text{with probability } \frac{1}{2} \\ -\sum_{j=1}^{M(n)} \xi_j & \text{with probability } \frac{1}{2}, \end{cases} \quad (49)$$

where $\{\xi_j\}$ are independent random variables which follow an exponential distribution with mean η , and

$$Pr(M(n) = m) = \frac{2^m}{2^{2n-1}} \binom{2n-m-1}{n-1}, \quad 1 \leq m \leq n. \quad (50)$$

He also shows that if $X^{(m)}$ is a random variable such that

$$X^{(m)} = \begin{cases} \sum_{j=1}^m \xi_j & \text{with probability } p \\ -\sum_{j=1}^m \xi_j & \text{with probability } 1-p, \end{cases} \quad (51)$$

and $Y \sim N(0, \sigma^2)$, then the probability density function of $X^{(m)} + Y$ is

$$f_{X^{(m)}+Y}(v) = \frac{\sigma^m e^{\sigma^2/(2\eta^2)}}{\eta^m \sigma \sqrt{2\pi}} \left\{ p e^{-v/\eta} Hh_{m-1}\left(-\frac{v\eta - \sigma^2}{\sigma\eta}\right) + (1-p) e^{v/\eta} Hh_{m-1}\left(\frac{v\eta + \sigma^2}{\sigma\eta}\right) \right\}. \quad (52)$$

Here, $Hh_m(x)$ is a Hh function defined by

$$Hh_m(x) = \frac{1}{m!} \int_x^\infty (t-x)^m e^{-t^2/2} dt. \quad (53)$$

By combining these results, we can write $f_u(u_{t+\Delta t}|u_t)$ as

$$\begin{aligned} & f_u(u_{t+\Delta t}|u_t) \\ &= \sum_{n=0}^{\infty} e^{-\lambda\Delta t} \frac{(\lambda\Delta t)^n}{n!} f_u^{(n)}(u_{t+\Delta t} - u_t) \\ &= \sum_{n=0}^{\infty} \sum_{j=1}^n e^{-\lambda\Delta t} \frac{(\lambda\Delta t)^n}{n!} \frac{2^m}{2^{2n-1}} \binom{2n-m-1}{n-1} f_{X^{(m)}+Y}(u_{t+\Delta t} - u_t - n\kappa). \end{aligned}$$

Because $f_u(u_{t+\Delta t}|u_t)$ is given as a sum of $f_{X^{(m)}+Y}(u_{t+\Delta t} - u_t - n\kappa)$, we only need to develop a fast algorithm for the latter. Moreover, because this consists of two terms as shown in eq. (52), we have only to consider each of the terms, that is,

$$f^{(-)}(v) = e^{-v/\eta} Hh_{m-1}\left(-\frac{v\eta - \sigma^2}{\sigma\eta}\right) \quad (54)$$

and

$$f^{(+)}(v) = e^{v/\eta} Hh_{m-1}\left(\frac{v\eta + \sigma^2}{\sigma\eta}\right). \quad (55)$$

Let's consider an expansion of $f^{(+)}(x_i - y_j)$, as we did in eq. (42). First from the recurrence of the Hh function

$$\frac{d}{dx} Hh_m(x) = -Hh_{m-1}(x), \quad m = 0, 1, 2, \dots \quad (56)$$

we can find a Taylor expansion of the Hh function as we did in (41), where the Taylor coefficients are expressed again by Hh functions (Though Hh_m functions are defined only for $m \geq -1$, we can naturally extend the definition for $m < -1$ by using the Hermite functions defined by (40)). Then, we can again expand the Hh function in the sum using (56), obtaining a two-sided expansion of the Hh function as we had in (42). Moreover, for the other factor in eq. (55), $e^{v/\eta}$, we have an expansion

$$e^{x_i - y_j} = e^{x_i - x_0} e^{x_0 - y_0} e^{y_0 - y_j}. \quad (57)$$

In this way, $f^{(+)}(x_i - y_j)$ has the same form of expansion as eq. (42). Based on this, we can construct a fast algorithm like the fast Gauss transform.

6.2 Application to a stochastic volatility model

As another example of the models to which our method can be applied, we take a stochastic volatility model analyzed by Heston [11]. In this model, the asset price S and its variance v evolve in time following the two-dimensional diffusion process:

$$dS_t = \mu_t S_t dt + \sqrt{v_t} S_t dZ_1 \quad (58)$$

$$dv_t = -\lambda(v_t - \bar{v})dt + \eta\sqrt{v_t}dZ'_2 \quad (59)$$

where \bar{v} is the mean of the variance toward which v_t reverts, η is the volatility of the variance, and Z_1 and Z_2 are standard Brownian motions with correlation ρ . By adopting the change of variables

$$x_t = \ln S_t - \int_0^t \mu_s ds \quad (60)$$

$$dZ'_2 = \rho dZ_1 + \sqrt{1 - \rho^2} dZ_2, \quad (61)$$

we can simplify the equations as follows:

$$dx_t = -\frac{1}{2}v_t dt + \sqrt{v_t} dZ_1 \quad (62)$$

$$dv_t = -\lambda(v_t - \bar{v})dt + \eta\sqrt{v_t}\rho dZ_1 + \eta\sqrt{1 - \rho^2} dZ_2, \quad (63)$$

where Z_1 and Z_2 are independent standard Brownian motions.

As in the cases of other models, to apply our method, we first have to calculate the conditional probability density function $f(x_T, v_T|x, v; t, T)$, where (x, v) is the initial position at time t and (x_T, v_T) is the final position at time T . From eqs. (62) and (63), this function can be shown to be the solution of the backward Kolmogorov equation [13]:

$$\frac{\partial f}{\partial t} = \frac{1}{2}v \frac{\partial^2 f}{\partial x^2} + \eta\rho v \frac{\partial^2 f}{\partial x \partial v} + \frac{1}{2}\eta^2 v \frac{\partial^2 f}{\partial v^2} - \frac{1}{2}v \frac{\partial f}{\partial x} - \lambda(v - \bar{v}) \frac{\partial f}{\partial v} \quad (64)$$

with the terminal condition

$$f(x_T, v_T|x, v; T, T) = \delta(x - x_T)\delta(v - v_T). \quad (65)$$

To solve the partial differential equation eqs. (64) and (65), we follow Heston's procedure to find a closed-form price formula for European options under this model, and take the Fourier transform of (64) and (65) with respect to x . Then we again take the Fourier transform of the results with respect to v_T , and obtain

$$\frac{\partial \tilde{f}}{\partial t} = -\frac{1}{2}k^2 v \tilde{f} + ik\eta\rho v \frac{\partial \tilde{f}}{\partial v} + \frac{1}{2}\eta^2 v \frac{\partial^2 \tilde{f}}{\partial v^2} - \frac{1}{2}ikv \tilde{f} - \lambda(v - \bar{v}) \frac{\partial \tilde{f}}{\partial v} \quad (66)$$

$$\tilde{f}(x_T, m|k, v; T, T) = e^{-ikx_T + imv}, \quad (67)$$

where k and m are new variables arising from the Fourier transform w.r.t. x and v_T , respectively. Now we assume the solution to have the form

$$\tilde{f}(x_T, m|k, v; T, T) = \exp\{C(x_T, m, k, t, T)\bar{v} + D(x_T, m, k, t, T)v\}. \quad (68)$$

Substituting this into eqs. (66) and (67) and equating the coefficients of both v and \bar{v} to zero gives the ordinary differential equations in C and D :

$$\begin{aligned} \frac{\partial C}{\partial t} &= \lambda D \\ \frac{\partial D}{\partial t} &= \alpha - \beta D + \gamma D^2 \equiv \gamma(D - r^+)(D - r^-), \end{aligned} \quad (69)$$

where

$$\begin{aligned} \alpha &= -\frac{1}{2}k^2 - \frac{1}{2}ik, & \beta &= -ik\eta\rho + \lambda, & \gamma &= \frac{1}{2}\eta^2, \\ r^\pm &= \frac{\beta \pm \sqrt{\beta^2 - 4\alpha\gamma}}{2\gamma} \equiv \frac{\beta \pm d}{\eta^2}. \end{aligned}$$

The terminal conditions are

$$\begin{aligned} C(x_T, m, k, T, T) &= -\frac{ikx_T}{\bar{v}} \\ D(x_T, m, k, T, T) &= im. \end{aligned} \quad (70)$$

The solutions to eqs. (69) and (70) are given as

$$C(x_T, m, k, t, T) = \lambda \left\{ r^-(t-T) - \frac{2}{\eta} \ln \frac{(img - r^-) - g(im - r^-)e^{-d(t-T)}}{r^-(g-1)} \right\} - \frac{ikv_T}{\bar{v}} \quad (71)$$

$$\begin{aligned} &\equiv C'(m, k, t, T) - \frac{ikv_T}{\bar{v}} \\ D(x_T, m, k, t, T) &= r^- \cdot \frac{(img - r^-) - e^{-d(t-T)}(im - r^-)}{(img - r^-) - ge^{-d(t-T)}(im - r^-)} \\ &\equiv D(m, k, t, T), \end{aligned} \quad (72)$$

where $g = \frac{r^-}{r^+}$. Finally, the solution to the original PDE is given by substituting these into the expression of \tilde{f} , and taking the inverse Fourier transform with respect to m and k :

$$f(x_T, v_T | x, v; t, T) = \frac{1}{(2\pi)^2} \int_{-\infty}^{\infty} dk \int_{-\infty}^{\infty} dm e^{ik(x-x_T)} e^{-imv_T} e^{D(m,k,t,T)v} e^{C'(m,k,t,T)\bar{v}}. \quad (73)$$

In the actual computation, we calculate this integral numerically by using discrete points (k_l, m_n) for (k, m) and introducing the weights $w_{k_l m_n}$:

$$f(x_T, v_T | x, v; t, T) \cong \sum_{k_l} \sum_{m_n} w_{k_l m_n} e^{ik_l(x-x_T)} e^{-im_n v_T} e^{D(m_n, k_l, t, T)v} e^{C'(m_n, k_l, t, T)\bar{v}}. \quad (74)$$

Now we are in a position to state the outline of a fast algorithm to compute the sum

$$G(x_i, v_i) = \sum_{j=1}^N q_j f(y_j, u_j | x_i, v_i; t, T), \quad i = 1, 2, \dots, M. \quad (75)$$

First, set two points (x_0, v_0) and (y_0, v_0) around which f should be expanded with respect to (x_i, v_i) and (y_j, u_j) , respectively. Noting that, from eq. (74), f depends on x_i, v_i, y_j and u_j only through exponential functions, we can expand $f(y_j, u_j | x_i, v_i; t, T)$ as follows:

$$\begin{aligned} f(y_j, u_j | x_i, v_i; t, T) &= \sum_{k_l} \sum_{m_n} w_{k_l m_n} e^{ik_l(x_i - x_0)} e^{D(m_n, k_l, t, T)(v_i - v_0)} \\ &\quad \times e^{ik_l(x_0 - y_0)} e^{D(m_n, k_l, t, T)v_0} e^{-im_n u_0} e^{C'(m_n, k_l, t, T)\bar{v}} \\ &\quad \times e^{ik_l(y_0 - y_j)} e^{im_n(u_0 - u_j)}. \end{aligned} \quad (76)$$

This readily provides a method to compute $G(x_i, v_i)$ in $O(M+N)$ time, because we can divide the computation into the following three steps:

1. Compute $\sum_{j=1}^N q_j e^{ik_l(y_0-y_j)} e^{im_n(u_0-u_j)}$.
2. Multiply the result by $e^{ik_l(x_0-y_0)} e^{D(m_n, k_l, t, T)v_0} e^{-im_n u_0} e^{C'(m_n, k_l, t, T)\bar{v}}$.
3. For each i , multiply the result of 2 by $e^{ik_l(x_i-x_0)} e^{D(m_n, k_l, t, T)(v_i-v_0)}$ and sum over k_l and m_n .

When we fix the number of points (k_l, m_n) used for numerical integration, steps 1, 2 and 3 can be done in $O(N)$, $O(1)$, and $O(M)$ work, respectively. Hence, the total work is $O(M + N)$, instead of $O(MN)$ which would be required by direct computation. Using this algorithm, we can construct a method similar to the stochastic mesh method and the multinomial method in two-dimensional (x, v) space.

The method can further be extended to include jumps in the asset price or even jumps in volatility, using the affine jump-diffusion framework of Duffie, Pan and Singleton [6]. Their framework provides ordinary partial differential equations for $C(x_T, m, k, t, T)$ and $D(x_T, m, k, t, T)$ similar to eq. (69), and in particular, when the distribution of the jump size depends neither on x nor v , the equation for $D(x_T, m, k, t, T)$ is exactly the same as we have derived here. So one only needs to change the multiplication factor $e^{C'(m_n, k_l, t, T)\bar{v}}$ in step 2 above in order to include the jumps.

7 Conclusion

In this paper, we have shown that in many of the numerical methods for pricing American options, the most computationally intensive part can be formulated as summation of Gaussians. In particular, we demonstrated this for the multinomial method and the stochastic mesh method for the Black-Scholes model and the lognormal jump-diffusion model. We then introduced the idea of the fast Gauss transform, and showed how it can reduce the order of computational work from $O(MN)$ to $O(M + N)$, where M is the number of summations and N is the number of terms appearing in each summation.

The results of numerical experiments show that the multinomial method based on the fast Gauss transform has a convergence rate much higher than that of the binomial method, and is more efficient than the latter when higher accuracy is needed. It was also shown that the stochastic mesh method can be accelerated by the fast Gauss transformation considerably, at least in one, two, and three dimensions.

Finally, we proposed an extension of the fast Gauss transform to handle non-Gaussian densities. We applied this method to Kou's double-exponential jump-diffusion model and Heston's stochastic volatility model. Implementation details and computational effectiveness remain to be investigated.

References

- [1] J. Alford and N. Webber: Very High Order Lattice Methods for One Factor Models, Jan. 2001.
- [2] L. Andersen and M. Broadie: Practical Primal-Dual Simulation Algorithms for Pricing Multidimensional American Options, Working paper, Columbia University, March 2001.
- [3] M. Broadie and J. Detemple: American Option Valuation: New Bounds, Approximations, and a Comparison of Existing Methods, *Review of Financial Studies*, Vol. 9, No. 4, pp. 1211-1250 (1996).
- [4] M. Broadie and P. Glasserman: A Stochastic Mesh Method for Pricing High-Dimensional American Options, Working paper, Columbia University, 1997.
- [5] D. Duffie: *Dynamic Asset Pricing Theory*, 3rd ed., Princeton University Press, 1996.
- [6] D. Duffie, J. Pan and K. Singleton: Transform Analysis and Asset Pricing for Affine Jump Diffusions, *Econometrica*, Vol. 68, pp. 1343-1376 (2000).
- [7] A. Florence: *Computational Multilinear Algebra*, Ph.D. dissertation, Cornell University, 2001.
- [8] M. Fu, S. Laprise, D. Madan, Y. Su and R. Wu: Pricing American Options: A Comparison of Monte Carlo Approaches, *Journal of Computational Finance*, Vol. 4, No. 3, pp. 39-88 (2001).
- [9] L. Greengard and J. Strain: The Fast Gauss Transform, *SIAM Journal on Scientific and Statistical Computing*, Vol. 12, No. 1, pp. 79-94 (1991).
- [10] L. Greengard and X. Sun: A new Version of the Fast Gauss Transform, *Documenta Mathematica*, Extra Volume ICM, III, pp. 575-584 (1998).
- [11] S. Heston: A Closed-form Solution for Options with Stochastic Volatility with Applications to Bond and Currency Options, *Review of Financial Studies*, Vol. 6, No. 2, pp. 327-343 (1993).
- [12] S. Heston and G. Zhou: On the Rate of Convergence of Discrete-Time contingent Claims, *Mathematical Finance*, Vol. 10, No. 1 pp. 53-75 (2000).
- [13] I. Karatzas and S. Shreve: *Brownian Motion and Stochastic Calculus*, 2nd ed., Springer-Verlag, New York, 1992.
- [14] S. G. Kou: A Jump Diffusion Model for Option Pricing, Working paper, Columbia University, Oct. 2001.

- [15] Y. K. Kwok: *Mathematical Models of Financial Derivatives*, Springer, 1998.
- [16] D. Lamberton and B. Lapeyre: *Introduction to Stochastic Calculus Applied to Finance*, Chapman & Hall / CRC, 1996.
- [17] R. Merton: *Continuous-Time Finance*, Blackwell, 1992.
- [18] I. Sloan and S. Joe: *Lattice Methods for Multiple Integration*, Clarendon Press, Oxford, 1994.
- [19] J. Strain: The Fast Gauss Transform with Variable Scales, *SIAM Journal on Scientific and Statistical Computing*, Vol. 12, No. 5, pp. 1131-1139 (1991).