Available online at www.sciencedirect.com

SCIENCE @ DIRECT®

Computers and Mathematics with Applications 49 (2005) 885–894

An International Journal
computers & mathematics with applications

www.elsevier.com/locate/camwa

A Distributed Algorithm for European Options with Nonlinear Volatility

C.-H. LAI, A. K. PARROTT AND S. ROUT
School of Computing and Mathematical Sciences, University of Greenwich
Old Royal Naval College, Greenwich, London SE10 9LS, U.K.
<C.H.Lai><A.K.Parrott><S.Rout>@gre.ac.uk

M. E. HONNOR
Department of Mathematics, University of Hertfordshire
Hatfield Campus, Herts AL10 9AB, U.K.
M.Honnor@herts.ac.uk

(Received and accepted March 2004)

Abstract—A distributed algorithm is developed to solve nonlinear Black-Scholes equations in the hedging of portfolios. The algorithm is based on an approximate inverse Laplace transform and is particularly suitable for problems that do not require detailed knowledge of each intermediate time steps. © 2005 Elsevier Ltd. All rights reserved.

Keywords—Option pricing, Black-Scholes equation, Finite-difference schemes, Nonlinear volatility.

1. INTRODUCTION

Financial modelling in the area of option pricing involves the understanding of hedge assets and portfolios in order to control the risk due to movements in share prices. Such activities depend on financial analysis tools being available to the trader with which he can make rapid and systematic evaluation of buy/sell contracts. In turn, analysis tools rely on fast numerical algorithms for the solution of financial mathematical models. There are many different financial activities apart from shares buy/sell activities. However, it is not the intention of this paper to discuss various financial activities. The main aim of this paper is to propose and discuss a distributed algorithm for the numerical solution of a European option. Both linear and nonlinear cases are considered.

The algorithm is based on the concept of the Laplace transform and its numerical inverse. First, a Laplace transform is applied to the linear Black-Scholes equation, which leads to a set of mutually independent linear ordinary differential equations. The set of differential equations may then be solved concurrently in a distributed environment. The scalability of the algorithm has been studied theoretically in [1,2]. This paper provides numerical tests to demonstrate the effectiveness of the algorithm for financial analysis. Time dependent functions for volatility and interest rates

The authors wish to thank Alan Davies, University of Hertfordshire, for his invaluable discussions and suggestions of the algorithm.

are also discussed. Second, an extension is given of the algorithm for nonlinear Black-Scholes equations where the volatility is a function of the option value. The Laplace transform is applied to a linearization of the nonlinear Black-Scholes equation. A set of mutually independent linear ordinary differential equations is obtained, and these equations may be solved in a distributed computing environment. The numerical inverse Laplace transform then is obtained within an outer iteration loop. The convergence behaviour of the algorithm is discussed.

The algorithm relies on fast computation of the numerical inverse Laplace transform. The main goal of this paper is to demonstrate the feasibility and effectiveness of using an inverse Laplace transform in applications to linear and nonlinear Black-Scholes equations. This paper will also examine the various computational issues of such a numerical inverse in terms of distributed computing.

2. THE BLACK-SCHOLES MODEL

Let $v(S,t)$ denote the value of an option, where S is the current value of the underlying asset and t is the time. The value of the option relates to the current value of the underlying asset via two stochastic parameters, namely, the volatility σ and the interest rate r, of the Black-Scholes equation,

$$\frac{\partial v}{\partial t} + \frac{1}{2}\sigma^2 S^2 \frac{\partial^2 v}{\partial S^2} + rS\frac{\partial v}{\partial S} - rv = 0 \in \Omega^+ \times [T, 0), \tag{1}$$

where $\Omega^+ = \{S : S \geq 0\}$. The stochastic background of the equation is not discussed in this paper, and readers who are interested should consult reference [3].

In this paper, attention is paid to European options, which mean that the holder of the option may execute at expiration a prescribed asset, known as the underlying asset, for a prescribed amount, known as the strike price. There are two different types of option, namely, the call option and the put option. At expiration, the holder of the call option has the right to buy the underlying asset and the holder of the put option has the right to sell the underlying asset. For a European put option with strike price k and expiration date T, it is sensible to impose the boundary conditions,

$$v(0,t) = ke^{-r(T-t)}, \qquad v(L,t) = 0,$$

where L is usually a large value. At expiration, if $S < k$, then one should exercise the call option, i.e., handing over an amount k to obtain an asset with S. However, if $S > k$ at expiration, then one should not exercise the option because of the loss $k - S$. Therefore, the final condition,

$$v(S,T) = \max\{k - S, 0\},$$

needs to be imposed. The solution v for $t < T$ is required.

The financial interpretation of the above model is as follows. The difference between the return on an option portfolio, which involves the first two terms, and the return on a bank deposit, which involves the last two terms, should be zero for a European option. Note that within a given short period, it is possible to assume the interest rate to be a constant rather than a stochastic parameter.

Since equation (1) is a backward equation, it needs to be transformed to a forward equation by using $\tau = T - t$, which leads to

$$\frac{\partial V}{\partial \tau} = \frac{1}{2}\sigma^2 S^2 \frac{\partial^2 V}{\partial S^2} + rS\frac{\partial V}{\partial S} - rV \in \Omega^+ \times (0, T], \tag{2}$$

subject to initial condition,

$$V(S,0) = \max\{k - S, 0\},$$

and boundary conditions,

$$V(0,\tau) = ke^{-r\tau}, \qquad V(L,\tau) = 0.$$

An analytic solution may be derived if a change of variable is made where the Black-Scholes model is converted to a time-dependent heat conduction equation with constant coefficients [3]. However, a field method, such as the finite-volume method, is of more interest for two reasons. First, there are many examples in multifactor model such that a reduction of the time dependent coefficient to a constant coefficient heat is impossible. Hence, analytic form of solutions cannot be found. Second, financial modelling typically requires large number of simulations and solutions at intermediate time steps are usually not of interest. Efficiency of the numerical algorithm is very important in order to make evaluation and decision before the agreement of a contract is reached. Ideally, one would like to use an algorithm which can be completely distributed onto a number of processors with only minimal communications between processors.

3. THE DISTRIBUTED ALGORITHM

It should be noted that in a field method for time dependent partial differential equations, the time step length is often restricted by a stability criterion and by the truncation errors in the discretised approximation of the time derivatives. On the other hand, concurrent computation of all time steps is almost impossible, i.e., it is not possible to apply a distributed algorithm.

Since the solutions at intermediate steps of (2) are usually not of interest, it is possible to apply the Laplace transform [4] to (2) and to reduce it to a number of mutually independent boundary value problems. Let

$$l(V) \equiv \int_0^\infty e^{-\lambda\tau} V(S,\tau)\, d\tau = U(\lambda; S)$$

be the Laplace transform of the function $V(S,\tau)$, then the Laplace transform of (2) leads to

$$\frac{1}{2}\sigma^2 S^2 \frac{d^2U}{dS^2} + rS\frac{dU}{dS} - (r+\lambda)U = -V(S,0) \in \Omega^+, \tag{3}$$

subject to the boundary condition,

$$U(\lambda;0) = \frac{k}{\lambda + r}, \qquad U(\lambda;L) = 0.$$

Here, $\lambda \in \{\lambda_j\}$ is a finite set of transformation parameters defined by

$$\lambda_j = j\frac{\ln 2}{T}, \qquad j = 1, 2, \dots, m, \tag{4}$$

where m is required to be chosen as an even number [5]. Therefore, the original problem (2) is converted to m independent parametric boundary value problems as described by (3), and these problems may be distributed and solved independently in a distributed environment which consists of a number of processors linked by a network. From experience, the value of m is usually a small even number not larger than ten [2]. Numerical experiments in Sections 4 and 5 also confirm such experience.

In order to retrieve $V(S,T)$, the approximate inverse Laplace transform due to Stehfest [5] given by

$$V(S,T) \approx \frac{\ln 2}{T}\sum_{j=1}^{m} w_j U(\lambda_j; S), \tag{5}$$

where

$$w_j = (-1)^{m/2+j} \sum_{k=(1+j)/2}^{\min(j,m/2)} \frac{k^{m/2}(2k)!}{(m/2-k)!k!(k-1)!(j-k)!(2k-j)!}$$

is known as the weighting factor, is used. This approximate inverse Laplace transform is by no means the most accurate one. The authors select Stehfest method because of previous experience with the method used for linear problems [2,5] and wish to investigate the application of the inverse method to option pricing problems.

For time varying $\sigma(t)$ and $r(t)$, it is possible to make a suitable coordinate transformation to the Black-Scholes equation in order to obtain a time independent-like heat equation [1]. Hence, the above Laplace transform method may still be applied.

4. EXAMPLES WITH CONSTANT VOLATILITY

An example of European put option is given in this section. The spatial domain is chosen to be $\Omega^+ = \{S : 0 \leq S \leq 320\}$, the strike price being $k = 100$ and the expiration date being $T = 0.25$ (three months). The two parameters σ and r are chosen to be 0.4 and 0.5 respectively, throughout the simulation period. A second-order finite-volume method is applied to each parametric equation as given by (3). The mesh size is chosen to be $h = 320/2^9$.

As a comparison the forward Black-Scholes equation as given by (2) is solved by means of an Euler marching scheme along the temporal axis with time step-length being 1/365, i.e., one day, in conjunction with the above finite-volume scheme applied along the spatial axis S. The discretisation leads to a set of tridiagonal system of equations at every time-step, which then may be solved by a direct method. The numerical solution for $V(S,T)$ obtained in this case is denoted by V_{TI}. The reason for choosing this method is that the computational complexity is reasonably low for the accuracy that it can achieve compared with higher-order schemes. On the other hand, the Laplace transformed set of equations is solved, sequentially in the same computational environment, with different values of m. An approximation to $V(S,T)$ corresponding to each value of m is found by using the inverse Laplace transform as described in Section 3 and is denoted as V_{IL}. Discrepancies between solutions, $\|V_{TI} - V_{IL}\|_2$, are recorded and shown in Table 1 for comparisons. Timings were obtained on a Sun Ultra-5 workstation using an F90 program which implements the above two methods.

Table 1. Timing and discrepancy comparisons.

m	2	4	6	8	10
$\|V_{TI} - V_{IL}\|_2$	1.0767	0.0812	0.0111	0.0037	0.0032
Time (V_{IL})	0.006	0.009	0.014	0.017	0.018

m	12	14	16	18
$\|V_{TI} - V_{IL}\|_2$	0.0032	0.0032	0.0032	0.0032
Time (V_{IL})	0.021	0.028	0.028	0.035

For comparison time (V_{TI}) 0.133

The timing of each run observed in this example consists of two parts. First, the second-order finite-volume solver time and, second, overheads due to the computation of inverse Laplace transforms. Dividing the timings for the inverse Laplace algorithm gives a crude estimation of the distributed processing time. Suppose there are as many processors available as the value of m, the scalability of the algorithm can still be easily observed from the sequential timings recorded in Table 1 using the above crude estimation of the corresponding distributed timings. The discrepancy $\|V_{TI} - V_{IL}\|_2$ approaches an asymptotic value of 0.0032 when $m \geq 8$. Therefore, it is not necessary to take m very much larger than ten. This result confirms the previous tests on a linear heat conduction problem [2].

As the total timings shown in Table 1 are very small, one may argue that the distributed algorithm is not necessary. However, as discussed in Section 5, the situation becomes very different in nonlinear problems due to the linearization steps, and hence, the total computation increases.

5. SOLVING NONLINEAR MODELS

Very often, over a short period of time the interest rate, r, is fixed while the volatility, σ, is varying. The volatility may be a function of the transaction costs [6], the second derivative of the option value [7], or, in some cases, the solution of a nonlinear initial value problem [6]. In order to develop the nonlinear solver in this section, the volatility proposed in [8], i.e.,

$$\sigma = \sigma_0\sqrt{1+a}$$

is used, where a is the proportional transaction cost scaled by σ_0 and the transaction time. Here, the authors adopted a heuristic approach in which the transaction cost is related to the option value and follows a Gaussian distribution. In order to demonstrate the inverse Laplace transform technique for nonlinear problems, a sine function is used to produce the effect of a pulse-like distribution instead of a Gaussian distribution. Therefore, the proportional transaction cost, a, may be replaced by a function of the option value such as

$$a = \sin\left(\frac{V\pi}{k}\right),$$

where k is the strike price. This volatility is used in the subsequent numerical tests.

The forward Black-Scholes model as given in (2) may be rewritten as

$$\frac{\partial V}{\partial \tau} = A(V)\frac{\partial^2 V}{\partial S^2} + rS\frac{\partial V}{\partial S} - rV, \tag{6}$$

where $A(V) = (1/2)\sigma(V)^2S^2$. Two linearization techniques can be applied to (6).

First, the coefficient A is computed by using an approximation $\bar{V}$, which is updated in every step of an iterative update process. Each step of the iterative update process involves a numerical solution to the equation,

$$\frac{\partial V}{\partial \tau} = A(\bar{V})\frac{\partial^2 V}{\partial S^2} + rS\frac{\partial V}{\partial S} - rV, \tag{7}$$

defined in the time interval $\tau \in (t_i, t_{i+1}]$. Let $V^{(n)}(S, t_{i+1})$ and $V^{(n)}(S, t_i)$be the numerical solutions of equation (6) at $\tau = t_{i+1}$ and $\tau = t_i$, respectively. The iterative update process to obtain the numerical solution $V^{(n)}(S, t_{i+1})$, using $V^{(n)}(S, t_i)$ as the initial approximation to $\bar{V}$, is described in the algorithm below.

ALGORITHM C1. Iterative coefficient—temporal integration.

```
Initial approximation:- $V^{(0)}(S,t_{i+1}) := V^{(n)}(S,t_i)$;
$k := 0$;
Iterate
    $k := k+1$;
    $\bar{V} := V^{(k-1)}(S,t_{i+1})$; {Store for comparison}
    Compute $A(\bar{V})$;
    $V^{(k)}(S,t_{i+1})$ := Apply a temporal marching step to equation (7);
Until $\left\|V^{(k)}(S,t_{i+1}) - V^{(k-1)}(S,t_{i+1})\right\| < \varepsilon$
$n := k$
```

Numerical solutions for equation (6) may be obtained by using a temporal integration method, which involves the nonlinear iteration loop, described in Algorithm C1, using an Euler marching step applied to equation (7), and an outer iteration loop, with $t_i = i\delta\tau$, $i = 0, 1, \dots$, where $\delta\tau$ is the step length of the temporal integration, applied to cover all of the temporal steps. Algorithm C1 is used to produce a reference solution for comparison.

Alternatively, a Laplace transform can be applied to equation (7), now being defined in the time interval $\tau \in (T_i, T_{i+1}]$, in its differential form which leads to

$$A(\bar{V})\frac{d^2U}{dS^2} + rS\frac{dU}{dS} - (r+\lambda)U = -V(S,T_i), \tag{8}$$

where

$$U = l(V).$$

An iterative update process is required to update $\bar{V}$ and to obtain the numerical solution $V^{(n)}(S,T_{i+1})$, using $V^{(n)}(S,T_i)$ as the initial approximation to $\bar{V}$, and is described in the algorithm below.

ALGORITHM C2. Iterative coefficient—inverse Laplace transform.

```
Initial approximation:- $V^{(0)}(S,T_{i+1}) := V^{(n)}(S,T_i)$;
$k := 0$;
Iterate
  $k := k+1$;
  $\bar{V} := V^{(k-1)}(S,T_{i+1})$; {Store for comparison}
  Compute $A(\bar{V})$;
  Parallel for $j := 1$ to $m(i)$
    Solve (8) for $U(\lambda_j;S)$;
  End Parallel for
  Compute $V^{(k)}(S,T_{i+1})$ using inverse Laplace in (5);
Until $\left\|V^{(k)}(S,T_{i+1}) - V^{(k-1)}(S,T_{i+1})\right\| < \varepsilon$
$n := k$
```

Here, $m(i)$ is the number of transformation parameters and $T_i = i\Delta\tau$. In order to solve equation (8) for $U(\lambda_j;S)$, one can employ the same finite-volume technique described in Section 4. In order to solve equation (6) for $V(S,T)$, Algorithm C2 needs to be iterated through $T_i := T_1, T_2, \ldots, T$ by using suitable values of $m(i)$ in the form of an outer iteration. In essence, the actual implementation does not require different values of $m(i)$ for many problems, and the results shown in this paper use the same number of transformation parameters, denoted as $\bar{m}$, for different values of i during the outer iteration loop. Note that in this case, $\Delta\tau$ can be chosen to be much greater than $\delta\tau$ because the fine details of $V(S,\tau)$ at each time step of a temporal integration is not required in the inverse Laplace transformation calculation. This paper does not intend to provide a full analysis of different possible choices of $\Delta\tau$. In the numerical tests shown in Section 6, $\Delta\tau = T/10$, $T/20$, $T/40$, $T/80$, all normalised with respect to one year, and $t_0 = 0$.

Second, a small perturbation may be applied to equation (6), defined in the time interval $\tau \in (t_i, t_{i+1}]$, and leads to

$$\begin{aligned}\left\{\frac{\partial}{\partial\tau} - \left(A'(V)\frac{\partial^2 V}{\partial S^2} + A(V)\frac{\partial^2}{\partial S^2} + rS\frac{\partial}{\partial S} - r\right)\right\}\delta V \\ = -\left\{\frac{\partial V}{\partial\tau} - \left(A(V)\frac{\partial^2 V}{\partial S^2} + rS\frac{\partial V}{\partial S} - rV\right)\right\},\end{aligned} \tag{9}$$

where δV is a small incremental change of V. Let

$$\delta V^{(n)}(S,t_{i+1})$$

be the numerical solutions of equation (9) at $\tau = t_{i+1}$. Newton's iterative method of obtaining the numerical solution $\delta V^{(n)}(S,t_{i+1})$, using $V^{(n)}(S,t_i)$ as the initial approximation to $V^{(0)}(S,t_{i+1})$, is described in the algorithm below.

ALGORITHM NM1. Newton's method—temporal integration.

```
Initial approximation:- $V^{(0)}(S,t_{i+1}) := V^{(n)}(S,t_i)$;
$k := 0$;
Iterate
  $k := k+1$;
```

```
    $\bar{V} := V^{(k-1)}(S, t_{i+1})$ ;{Store for comparison}
    Compute $A(\bar{V})$; Compute $A'(\bar{V})$;
    Compute $A'(\bar{V})\frac{\partial^2 \bar{V}}{\partial S^2}$;
    Compute $-\left\{\frac{\partial \bar{V}}{\partial \tau} - \left(A(\bar{V})\frac{\partial^2 \bar{V}}{\partial S^2} + rS\frac{\partial \bar{V}}{\partial S} - r\bar{V}\right)\right\}$;
    $\delta V^{(k)}(S, t_{i+1})$ :=Apply a temporal marching step to equation (9);
    $V^{(k)}(S, t_{i+1}) := \bar{V} + \delta V^{(k)}(S, t_{i+1})$;
Until $\|\delta V^{(k)}(S, t_{i+1})\| < \varepsilon$
$n := k$
```

Similar to the iterative coefficient method numerical solutions for equation (6) may be obtained by using a temporal integration method, which involves the nonlinear iteration loop, described in Algorithm NM1, using an Euler marching step applied to equation (7), and an outer iteration loop, with $t_i = i\delta\tau$, $i = 0, 1, \ldots$, where $\delta\tau$ is the step length of the temporal integration, applied to cover all of the temporal steps.

Alternatively, a Laplace transform then may be applied to (9), now being defined in the time interval $\tau \in (T_i, T_{i+1}]$, in its differential form results to

$$\begin{aligned} l(\delta V) - \delta V(S, T_i) - \left(A'(V)\frac{\partial^2 V}{\partial S^2} + A(V)\frac{\partial^2}{\partial S^2} + rS\frac{\partial}{\partial S} - r\right) l(\delta V) \\ = -l(V) - V(S, T_i) - \left(A(V)\frac{\partial^2 V}{\partial S^2} + rS\frac{\partial V}{\partial S} - rV\right) \end{aligned} \tag{10}$$

Newton's iterative method of obtaining the numerical solution $l(\delta V^{(n)}(S, T_{i+1}))$, using $V^{(n)}(S, T_i)$ as the initial approximation to $V^{(0)}(S, T_{i+1})$, is described in the algorithm below.

ALGORITHM NM2. Newton's method—inverse Laplace transform.

```
Initial approximation:- $V^{(0)}(S, T_{i+1}) := V^{(n)}(S, T_i)$;
$k := 0$;
Iterate
    $k := k + 1$;
    $\bar{V} := V^{(k-1)}(S, T_{i+1})$ ;{Store for comparison}
    Compute $A(\bar{V})$; Compute $A'(\bar{V})$;
    Compute $A'(\bar{V})\frac{\partial^2 \bar{V}}{\partial S^2}$;
    Compute $-l(\bar{V}) - V(S, T_i) - \left(A(\bar{V})\frac{\partial^2 \bar{V}}{\partial S^2} + rS\frac{\partial \bar{V}}{\partial S} - r\bar{V}\right)$;
    Parallel for $j := 1$ to $m(i)$
        Solve (10) for $l(\delta V^{(k)}(S, T_{i+1}))$;
        End Parallel for
        Compute $\delta V^{(k)}(S, T_{i+1})$ using inverse Laplace in (5);
        $V^{(k)}(S, T_{i+1}) := \bar{V} + \delta V^{(k)}(S, T_{i+1})$;
Until $\|\delta V^{(k)}(S, T_{i+1})\| < \varepsilon$
$n := k$
```

Here, $m(i)$ is the number of transformation parameters. In order to solve equation (10) for $l(\delta V^{(n)}(S, T_{i+1}))$, one can employ the same finite-volume technique described in Section 4. Similar to the method of iterative coefficient Algorithm NM2 can now be iterated through $T_i := T_1, T_2, \ldots, T$ by choosing suitable values of $m(i)$ in the form of an outer iteration. The other discussion for the choice of $m(i)$ and $\Delta\tau$ also applies.

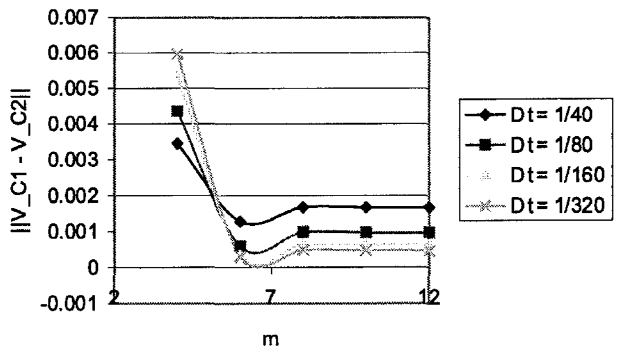
6. EXAMPLES WITH NONLINEAR VOLATILITY

As an example of the distributed algorithm and the two linearization methods, the same problem of European put option described in Section 4 is used. The volatility σ is chosen as a function as described above and the parameters σ_0 and r are chosen to be 0.4 and 0.5, respectively, throughout the simulation period. A second-order finite-volume method is applied to each parametric equation as given by (8) or (10). The mesh size is chosen to be $h = 320/2^9$.

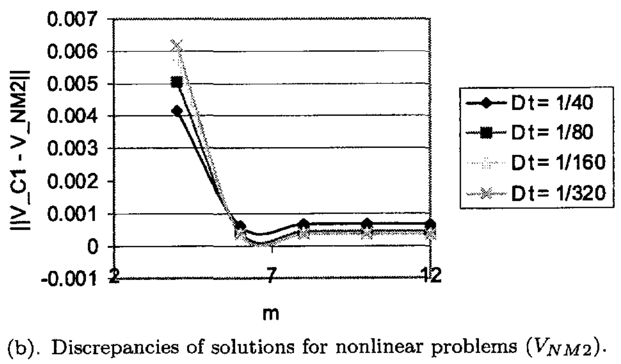
The Laplace transformed set of equations for each of the linearization methods is solved, sequentially in the same computational environment, with different values of $\bar{m}$ and an approximation to $V(S,T)$ corresponds to each value of $\bar{m}$ is found by using the inverse Laplace transform as described in Section 3. The approximations obtained by means of the iterative coefficient and Newton's methods are denoted as V_{C2} and V_{NM2}, respectively. By using the choices of $\Delta\tau = T/10$, $T/20$, $T/40$, $T/80$, the number of outer iterations required for Algorithm C2 and Algorithm NM2 is 10, 20, 40, and 80, respectively.

The above two distributed algorithms are compared with the linearised forward Black-Scholes equation given in (8) solved by means of Euler marching scheme, Algorithm C1, along the temporal axis with $\delta\tau = 1/365$, i.e., one day, in conjunction with the second-order finite-volume scheme applied along the spatial axis S. The discretisation leads to a number of tridiagonal systems of equations due to the linearisation step at every time step, which may be solved by a direct method. The numerical solution $V(S,T)$ obtained by this temporal integration is denoted as V_{C1}. An F77 program was written, which implements the above two linearisation methods, and run on a COMPAQ laptop. The stopping criterion used in the linearization step is chosen as $\varepsilon = 10^{-5}$.

Defining one work unit as the computational work required for solving a tridiagonal system of equations results from a chosen mesh size. The total sequential work unit is obtained by multiplying the total number of work unit to $\bar{m}$, and the total parallel work unit is simply the



(a). Discrepancies of solutions for nonlinear problems (V_{C2}).



(b). Discrepancies of solutions for nonlinear problems (V_{NM2}).

Figure 1.

Table 2. Computational work unit (V_{C1} requires 246 work unit).

$\bar{m}$	4	6	8	10	12
$\Delta\tau$	Using Algorithm C2				
1/40	58	58	58	58	180
1/80	103	103	103	103	123
1/160	177	177	177	177	186
1/320	326	326	326	326	327
$\Delta\tau$	Using Algorithm NM2				
1/40	43	43	43	43	43
1/80	83	70	71	71	71
1/160	134	126	126	126	126
1/320	249	245	220	214	213

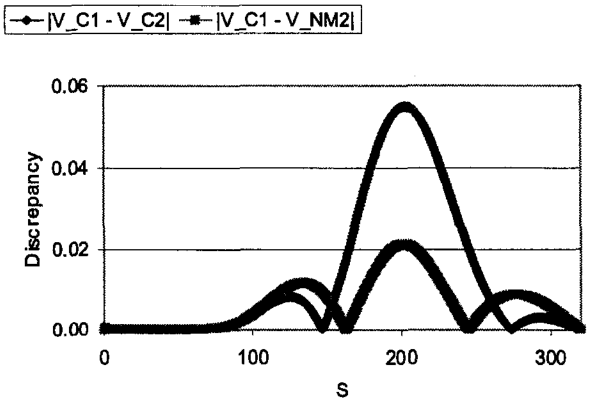


Figure 2. Point-wise discrepancies of solutions for nonlinear problems.

total work unit plus inverse Laplace transform and communication overheads. Discrepancies in solutions, i.e., $\|V_{C1} - V_{C2}\|_2$ and $\|V_{C1} - V_{NM2}\|_2$ using various $\Delta\tau$, are recorded in Figures 1a and 1b. In general, the discrepancy levels off when $m \geq 8$, which suggests that the use of more terms in the inverse Laplace transform at a fixed value of $\Delta\tau$ has no effect on the accuracy. On the other hand, smaller $\Delta\tau$ produces small discrepancy at the expense of requiring more work unit as recorded in Table 2 for comparisons. Furthermore, the work unit required by using Algorithm NM2 is less than that of Algorithm C2, and there is no sudden increase of work when $m = 12$.

The other feature of the two linearization methods is that the use of iterative coefficients has no advantage over Newton's method. Finally, the pointwise discrepancies of the solutions, $|V_{C1}(S,T) - V_{C2}(S,T)|$ and $|V_{C1}(S,T) - V_{NM2}(S,T)|$, as compared with a time-stepping method applied to the nonlinear problem with $\bar{m} = 10$ and $\Delta\tau = 1/80$ for the case $h = 320/2^9$ is recorded in Figure 2. Such comparison is not the best way of comparing results, but it gives an idea of the deviation from the numerical solution obtained by a temporal integration using Euler's method, which is well documented with error analyses in the literature. It shows that the largest discrepancies occur nearby the strike price. Therefore, it is the resolution problem of the finite-volume scheme near the strike price. On the other hand, the order of discrepancies is of the order of 10^{-3} which is in consistent with the results for linear problems. Finally, the numerical approximations to $V(S,T)$ using various methods and the initial condition $V(S,0)$ are plotted in Figure 3.

Note that the linearization techniques applied above may be extended to handle American options and Asian options with little difficulties. A complimentary problem can always be formed for American options, in which the free boundary is embedded into the formulation. A Laplace

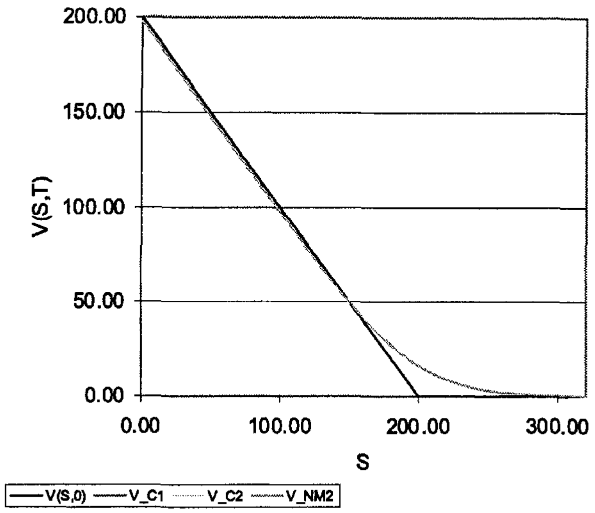


Figure 3. Comparison of numerical solutions at time T.

transform can be applied then to the complimentary problem [3] followed with an inner iteration loop to drive the inequality to convergence. Work in this area is currently being conducted by the authors.

7. CONCLUSIONS

A distributed algorithm for solving European options model is discussed. Numerical examples are provided for a European put option with one spatial variable S. Timings of the linear problems obtained on a Sun Ultra-5 workstation show the advantages of the present Laplace transform approach for option pricing. A projection of the timing to a distributed computing environment shows the scalability of the algorithm. Two linearisation methods were used in conjunction with the inverse Laplace transform method for nonlinear Black-Scholes models are discussed. Work unit counts were also presented. The computational work unit suggests that inverse Laplace techniques have advantages in solving nonlinear option pricing problems. Further, investigation into other methods of the inverse Laplace transform is currently undertaken by the authors.

REFERENCES

1. D. Crann, The Laplace transform: Numerical inversion for computational methods, In Technical Report No. 21, University of Hertfordshire, (July 1996).
2. D. Crann, A.J. Davies, C.-H. Lai and S.W. Leong, Time domain decomposition for European options in financial modelling, In *Proceedings of the 10th International Conference on Domain Decomposition Methods,* August 1997, Colorado, *Domain Decomposition Methods 10,* (Edited by J. Mandel, C. Farhat and X.-C. Cai), American Mathematical Society, (1998).
3. P. Wilmott, S. Howison and J. Dewynne, *The Mathematics of Financial Derivatives,* Press Syndicate of the University of Cambridge, New York, (1973).
4. D.V. Widder, *The Laplace Transform,* Princeton University Press, Princeton, (1946).
5. H. Stehfest, Numerical inversion of Laplace transforms, *Comm ACM* **13**, 47–49, (1970).
6. G. Barles and H.M. Soner, Option pricing with transaction costs and a nonlinear Black-Scholes equation, *Finance Stochast* **2**, 369–397, (1998).
7. A. Parás and M. Avellaneda, Dynamic hedging portfolios for derivative securities in the presence of large transaction costs, *Appl. Math. Finance* **1**, 165–193, (1994).
8. P. Boyle and T. Vorst, Option replication in discrete time with transaction costs, *Journal of Finance* **47**, 271–293, (1973).