International Journal of Pure and Applied Mathematics
Volume 5 No. 4 2003, 447-463

A PARALLEL MATHEMATICAL SOFTWARE FOR ASIAN OPTIONS PRICING

F. Perla
Faculty of Economics
University of Naples "Parthenope"
Via Medina 40, 80133 Naples, ITALY
e-mail: francesca.perla@uninav.it

Abstract: Advances in high performance computing provide new opportunities for implementing computationally intensive *financial applications.* However, the main critical aspects remain related to the general problem to realize efficient mathematical software in high performance computing environment.

In this paper we present a parallel mathematical software to evaluate *Asian options* - an important class of financial derivatives - based on Monte Carlo and Quasi Monte Carlo methods. The reason for choosing those methods is that a mathematical model describing the considered Asian options involve multidimensional integrals with a very high dimension.

First results, in terms of accuracy and efficiency obtained testing the developed routine on a *cluster of PC connected by a fast network*, are also showed.

AMS Subject Classification: 90A09, 68N, 65D32
Key Words: high performance computing, finance problems, Asian options, multidimensional quadrature

1. Introduction

Financial modeling represents a promising industry application of *High Performance Computing* (HPC). In a previous paper [16] we reported the experiences related to the development of mathematical software to manage a partic-

Received: February 24, 2003

ular class of financial derivatives, namely *Collateralized Mortgage Obligations* (CMO), in HPC environment. The mathematical model arising in the simulation of CMO involve *multidimensional integrals*. It is well known that the most promising methods to compute high dimensional integrals are the Monte Carlo (MC) methods and even more, the Quasi Monte Carlo (QMC) methods [9]. Then, in [16] we developed parallel algorithms and software, based on those methods, to calculate the multidimensional integrals arising from the evaluation of CMO. The obtained results, in terms of accuracy and efficiency, were very encouraging. Now, we intend to apply that approach to develop a parallel mathematical software to evaluate another class of financial instruments, named *Asian options*.

Asian options involve a pay-off that depends on the average of the asset price over some prespecified period of time. Asian options are an important class of path-dependent options as, by their design, they reduce the significance of the closing price at the maturity of the option and, then, remove extreme sensitivity of the option's expiration value to the underlying cash price on a particular day. These features make Asian options very attractive conservative financial instruments and have become more and more commonplace. They are especially useful in cases, where the underlying asset is a foreign currency or a commodity such as metals or fuels.

Much effort has recently been spent on the pricing of various Asian options. Valuing such options is nontrivial when the form of averaging is *arithmetic*, that is the terminal value of the contract is determined by the arithmetic average of the past prices. In this case, it is known that no closed-form solution exists. For the arithmetic Asian options a lot of methods have been proposed. Most of them involve Laplace transforms [11, 12], analytic approximation based on moment matching [18, 25], convolution methods using the Fast Fourier transform [6], a number of PDE methods [1, 10, 22, 26, 27] and binomial methods [7, 14]. Further, many researchers have employed MC and QMC methods [2, 4, 11, 13, 15, 18]. Owing to their dependence on the entire path of the underlying asset, Asian options appear to fit Monte Carlo simulation.

Since the aim of this work is to develop an efficient and portable high performance mathematical software to evaluate Asian options, the choice of the methods to implement depends, not only on their accuracy, but also on the efficiency that they deliver in high performance computing environments. Looking at the results of our previous experiments [16], we prefer to concentrate on the MC and QMC methods since they are efficient and flexible methods to be used in high performance computing environment.

In the Section 2 we describe the considered Asian option in order to show that a mathematical model arising in the simulation of this option involve a multidimensional integral. In Section 3 we shortly describe the numerical methods used, referring the reader to [16, 19] for a deeper description of MC and QMC methods. Further, we show the algorithm based on those methods to evaluate the Asian option. Finally, in Section 4 we present the developed parallel algorithm, the related mathematical software and the results of computational experiments carried out on a parallel environment consisting of a *cluster of processors connected by a fast communication network*.

2. The Financial Problem: Asian Options

Asian options involve a pay-off that depends on the average of the asset price over some prespecified period of time. If the average is computed using a finite sample of asset price observations taken at a set of regularly spaced time points, we have a *discrete Asian option*, as opposed to a *continuous Asian option*, which is obtained by computing tha average via the integral of the price path over an interval of time [11]. In addition the form of averaging can be either *arithmetic* (sum) or *geometric* (product). With the canonical assumption of a log-Gaussian distribution for stock prices, the geometric average is analytically tractable in closed form [13], whereas the arithmetic average is not. For this reason, in this paper we devote our attention to discretely sampled arithmetic average Asian options.

To formally describe the considered contract, let us introduce some notation. Let $S(t)$ denote the current stock price at time t. Fixed that the average is computed over the time interval $[t_0, t_N]$ and at the points on the interval $t_i = t_0 + ih$ for $i = 0, \ldots, N$, where $h = (t_N - t_0)/N$. Let $A(t)$ denote the average at the current time point, t, $t_m \leq t \leq t_{m+1}$, which is defined by:

$$A(t) = \frac{1}{m+1} \sum_{i=0}^{m} S(t_i)\,,$$

for a corresponding integer $0 \leq m \leq N$, and $A(t) = 0$ for $t < t_0$. Thus, $A(t_N)$ represents the simple arithmetic average of $N+1$ prices taken at equal intervals of time, h, between t_0 and t_N. Typically, this time interval is specified to be a day, a week, or a month [18]. The considered option is characterised by the pay-off function at time t_N given by $C[A(t_N), t_N, K, N] = \max[A(t_N) - K, 0]$ for a call option, or $P[A(t_N), t_N, K, N] = \max[K - A(t_N), 0]$ for a put option, where K denotes the specified strike price of the option.

The stock price process assumed is usually the familiar geometric diffusion:

$$dS(t) = \mu S(t)dt + \sigma S(t)dz\,, \tag{1}$$

where dz is a Wiener process (that is, it is distributed as normal with mean zero and unit standard deviation, $N(0,1)$), and μ and σ are, respectively, the constant drift and volatility parameters. Under (1) we can express $S(t_i)$ in terms of $S(t_{i-1})$ as:

$$S(t_i) = S(t_{i-1})e^{(\mu-0.5\sigma^2)h+\sigma\sqrt{h}y_i}\,, \tag{2}$$

where y_i is $N(0,1)$. To value this option, a frequently used approach is to adopt the risk-neutrality transformation of Cox and Ross [8]. Under the equivalent distribution, relative prices follow a martingale and the expected returns on the stock will be the risk-free rate of interest r, implying that $\mu = r$, in (2). Under these assumptions, the value of the (call) option is:

$$C[A(t), t_N, K, N] = e^{-r(t_N-t)}E_t[\max(A(t_N) - K, 0)]\,, \tag{3}$$

where E_t is the expectation operator with respect to the equivalent probability distribution, given the stock price at time t.

In particular, the equation (3) can be written in the following form:

$$C[A(t), t_N, K, N] = e^{-r(t_N-t)}\int_{-\infty}^{+\infty}\int_{-\infty}^{+\infty}\cdots\int_{-\infty}^{+\infty} g(y_1, y_2, \ldots, y_N) \times \phi(y_1)\phi(y_2)\ldots\phi(y_N)dy_1dy_2\ldots dy_N\,,$$

where g and ϕ depend on the assumptions (1) and (2). By change of variables, $x_i = \Phi(y_i)$, with

$$\Phi(y) = \frac{1}{2\pi}\int_{-\infty}^{y} e^{-s^2/2}ds$$

the normal cumulative distribution function, it is easy to see that

$$C[A(t), t_N, K, N]$$

$$= e^{-r(t_N-t)}\int_0^1\int_0^1\cdots\int_0^1 g(\Phi^{-1}(x_1), \Phi^{-1}(x_2), \ldots, \Phi^{-1}(x_N))dx_1dx_2\ldots dx_N$$

$$= e^{-r(t_N-t)}\int_0^1\int_0^1\cdots\int_0^1 f(x_1, x_2, \ldots, x_N)dx_1dx_2\ldots dx_N$$

$$= e^{-r(t_N - t)} \int_{[0,1]^N} f(\mathbf{x}) d\mathbf{x} \,. \tag{4}$$

Therefore, our problem is reduced to a problem of computing a *multidimensional integral* over the N-dimensional unit cube.

The MC methods, and even more, the QMC methods are very useful tools to calculate such kind of integral, that is for evaluating the Asian option, as evidenced by the voluminous literature on successful applications, especially on financial applications [2, 5, 11, 13, 15, 21, 23]. In the next section, we briefly describe the basic idea of the MC and QMC methods, and we propose a parallel algorithm, based on those methods, to evaluate the Asian option under consideration.

3. Numerical Methods and Algorithm

The idea underlying the Monte Carlo approach is to replace the integral of $f(\mathbf{x})$, which is a continuous average, by a discrete average over randomly chosen points. More precisely, the Monte Carlo approach approximates

$$I(f) = \int_{[0,1]^N} f(x_1, x_2, \ldots, x_N) dx_1 dx_2 \ldots dx_N \,,$$

by

$$Q(f) = \frac{1}{n} \sum_{i=1}^{n} f(\mathbf{t_i}) \,, \tag{5}$$

where $\mathbf{t_i} = (t_{i1}, \ldots, t_{iN})$, $i = 1, \ldots, n$ are n independent N-dimensional random points. It is well known that the expected error is

$$E_n(f) = \frac{\sigma(f)}{\sqrt{n}} \,, \tag{6}$$

where $\sigma^2(f)$ denote the variance of f. The expression (6) shows why, in comparison with other methods of numerical integration, MC becomes increasingly attractive as the dimension of the integral increases. Furthermore, the MC method is flexible and easy to implement and modify. In addition, the increased availability of powerful computers has significantly reduced the execution time required by the method.

However, the MC method suffers from the disadvantage that the rate of convergence is only proportional to $n^{-1/2}$. This motivated the search for methods with faster convergence. An improvement of the convergence can be sometimes obtained by using deterministic sequences rather than random sequences to evaluate the integral. These deterministic sequences are chosen to be more evenly dispersed through the region of integration than random sequences. *Discrepancy* measures the extent to which the points are evenly dispersed throughout the region. Deterministic sequences with this property are known as *low discrepancy* sequences or *quasi-random* sequences. The numerical integration methods based on those low discrepancy sequences are named *low discrepancy methods or Quasi Monte Carlo methods* (QMC). A complete description of these methods is outside the aim of this work, and we refer the reader to [19], in which the QMC methods are deeply reviewed. QMC methods provide deterministic error bounds $O(n^{-1}(\log n)^N)$ for suitably chosen deterministic sequences. There are several types of quasi-random sequences; here we confine ourselves to two of these low discrepancy sequences: the *Halton sequence* and the *Faure sequence*. Studies using these low discrepancy sequences in finance applications have found that the errors produced are substantially lower than the corresponding errors generated by crude Monte Carlo [2, 15, 20, 21]. However, other numerical experiments conducted on not financial applications seem to show QMC methods to offer no practical advantage over MC method for rather modest values of the dimension, say, $N = 30$ [3]. These experiences serves as a useful caution against assuming that QMC methods will outperform MC method in all situation [24]. This is the reason, why in this paper we take under consideration both methods.

Any of the two methods we used to calculate $Q[f]$ in (5), the algorithm to implement for evaluating the Asian option can be described as follows:

```
input(N; K; r; σ; t_N; t_0; t; S(t))
input(n)
Q[f] := 0
for i = 1,n
    generate(t_i)
    generate(y_i) with y_ij = Φ^{-1}(t_ij), for j = 1,...,N
    calculate f(t_i) = g(y_i)
    Q[f] := Q[f] + f(t_i)
endfor
Q[f] := Q[f]/n
C := e^{-r(t_N - t)} Q[f]
```

Algorithm 1: Algorithm to evaluate $C[A(t), t_N, K, N]$ in (4) using the formula (5)

In the previous algorithm, the first *input* is related to the Asian option parameters (see Section 2), while the second one is related to the number of N-dimensional points to generate for evaluating $Q[f]$. For each generated point $\mathbf{t_i} = (t_{i1}, t_{i2}, \ldots, t_{iN})$ of the sequence, we have to calculate $\mathbf{y_i} = (\Phi^{-1}(t_{i1}), \Phi^{-1}(t_{i2}), \ldots, \Phi^{-1}(t_{iN}))$, where Φ^{-1} denotes the inverse normal cumulative function. To do that, we use a version of the routine *dinvnr* from the package *DCDFLIB* written by B.W. Brown, J. Lovato and K. Russell (University of Texas), available through *NETLIB*. After that, we have to evaluate $g(\mathbf{y_i})$ and to execute a sum. At the end, the variable C contains an approximate value of the option.

In the next section, we illustrate a parallel version of Algorithm 1, based on the parallel multidimensional routine proposed in [16].

4. Parallel Asian Options Software and Numerical Experiments

To develop a mathematical software, based on the Algorithm 1, to evaluate the considered Asian option, in an high performance computing environment, we need to use an efficient parallel multidimensional quadrature routine. In [16] we proposed a high performance building block for multidimensional quadrature based on crude MC method and on two QMC methods, using respectively the Halton sequence and the Faure sequence. The main features of that routine are the following:

- the target computing environment is a distributed memory computer consisting of P processors. Each of them is identified by a number q, called *id-processor*, varying from 0 to $P-1$. The processors are connected by a communication network that allows to exchange messages;

- the parallelism in the rule (5) is introduced distributing the n function evalutions among the P processors, that is each processor generates only n/P points of the sequence and uses them to "locally" compute a quadrature rule. After that, the P processors have to combine the partial results to obtain the final result $Q[f]$;

- for the Halton and the Faure sequences, each processor can generate the "local" points without interaction with the other processors. For the random sequence used in the Monte Carlo method, each processor executes a version of the parallel generator introduced in [17];

- the programming model used is *Single Program Multiple Data* (SPMD), that is the same copy of the algorithm (the so called *node algorithm*), is executed by each processor on different data.

Using the described parallel multidimensional algorithm, we developed a SPMD parallel version of the Algorithm 1, as follows:

locinit(q;P)
input(N; K; r; σ; t_N; t_0; t; $S(t)$)
input(n)
locsum := 0
for $i = 0, n/P - 1$
  generate($\mathbf{t}_i^{(q)}$)
  generate($\mathbf{y}_i^{(q)}$) *with* $y_{ij}^{(q)} = \Phi^{-1}(t_{ij}^{(q)})$, *for* $j = 1, \ldots, N$
  calculate $f(\mathbf{t}_i^{(q)}) = g(\mathbf{y}_i^{(q)})$
  locsum := locsum + $f(\mathbf{t}_i^{(q)})$
endfor
compglobal(Q[f])
$Q[f] := Q[f]/n$
$C := e^{-r(t_N - t)} Q[f]$

Algorithm 2: Node algorithm of parallel version of the Algorithm 1

In the Algorithm 2, the function *locinit* inizializes the *id-processor* q, $\mathbf{t}_i^{(q)} = \mathbf{t}_{q+iP}$ is the i-th node generated by the processor q, and *compglob* denotes the function that computes the global sum $Q[f]$. The computation of the global sum is performed by a *cascade sum* in $\log_2 P$ steps.

Using the Algorithm 2, a first version of a parallel mathematical software, forward named *PQMCASIAN*, has been built. *PQMCASIAN* is written in *ANSI C* and uses the *Parallel Virtual Machine* (PVM) communication system in order to run on distributed parallel computers. The software also runs on a single computer. *PQMCASIAN* is designed to run under the operating system *UNIX* (or *UNIX*-like). The routine includes the following components:

- a driver, running on one or on P processors, whose input parameters are (1) the parameters related to the Asian option to evaluate, (2) the number of processors to use, (3) a parameter to select the method to be used, (4) various other parameters related to the accuracy of the computation;

- a parallel MC routine with a random points generator which use a combination of three *linear congruential generators*;

- a parallel QMC routine with low discrepancy points generators for the Halton sequence and the Faure sequence;

- all the necessary makefiles (for the sequential version and for the distributed version);
- some examples tests;
- documentation for the description of the software.

The software is written in a modular way so that other kinds of deterministic and random number generators can be easily include.

4.1. Numerical Experiments

We now present some results of testing the reliability and the efficiency of the proposed software *PQMCASIAN*. In order to do that, we use a set of Asian options drawn from [15, 18]. The options valued have $N = 52$ (52-week arithmetic average option with 53 reset points), $(t_N - t_0)$=one year, the riskless rate $r = 0.09$ and the volatility $\sigma = 0.5$. We have computed prices for this structure at the point time $t = t_0$ with a stock price $S(t) = 100$.

The first set of experiments evaluates the reliability on just one processor, by comparing the results of the three methods on the *call* option[1] for various strike price, in particular for $K = 90, 100, 110$ (Figures 1-3). In the pricing examples, the horizontal axis of the graphs denotes the number of points in the MC and QMC sequences (varying from 10^3 to 5×10^6). The vertical axis denotes the obtained prices of the option.

In each graph we also report two benchmark values, to evaluate the accuracy of the obtained results: the value of the *geometric solution*, that is the solution of the corresponding geometric Asian option, which closed-form solution exists, and the value obtained by *Levy and Turnbull* in [18] using MC method.

We summarize our conclusions:

- QMC methods converge faster than the MC method;
- QMC methods do not only reach the answer fast but mantain their performance, while the MC may fluctuate;
- the convergence of the QMC methods is smoother than the convergence of the MC method;
- the Faure method outperforms the Halton method;

[1] Given the average call option price, the value for the corresponding average put option, can subsequently be found using the put-call parity relation, as showed in [18].

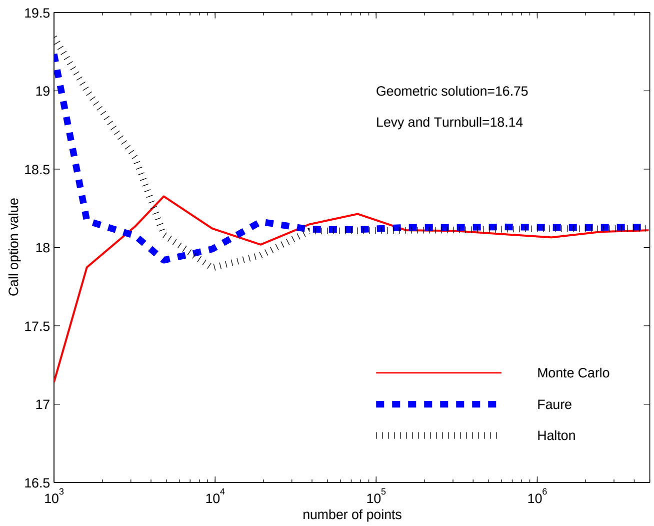

Figure 1: Call option values for $K = 90$

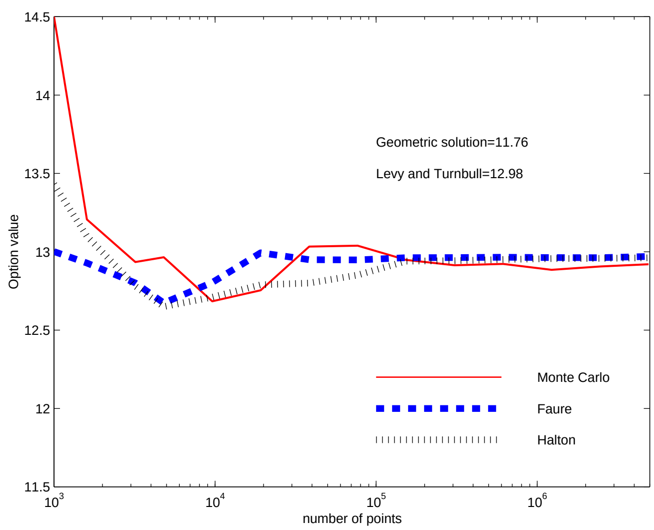

Figure 2: Call option values for $K = 100$

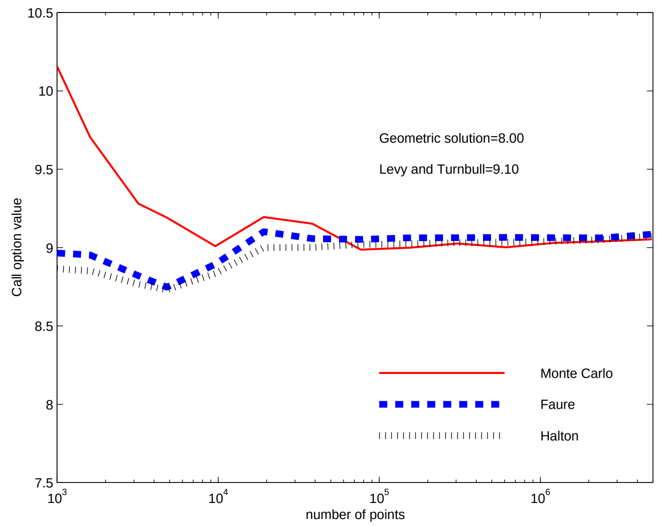

Figure 3: Call option values for $K = 110$

- the averaging period is quite long, this causes slower convergence for both QMC methods (see Section 3), but the Faure sequence is still an improvement.

All these results agree with the obtained results of analogous experiments carried out by other researchers to solve the same kind of problem [2, 15, 18].

The efficiency of the software to use is more and more importante for people, who evaluate Asian options. They need methods, which can evaluate a derivative in a matter of minutes. Rather low accuracy, on the order of 10^{-2} to 10^{-4}, is often sufficient. The integrals to evaluate are complicated and computationally intensive. We therefore evaluate the performance of the software when the number of the processors P increases, in order to estimate the gain deriving from the parallelization. To do that, we use the classical parameter:

$$\text{SPEED-UP} \quad S_P = \frac{T_1(s)}{T_P(s)},$$

where $T_P(s)$ is the elapsed execution time on P processors using s integrand function evaluations. The speed-up values reported in Figure 4, have been obtained testing the software on a cluster of 8 PC *AMD Athlon* at 1200 Mhz each operated by University of Naples "Parthenope". The processors are connected by a *FastEthernet* communication network, and they implement the PVM communication system. Even if such computational environment is not

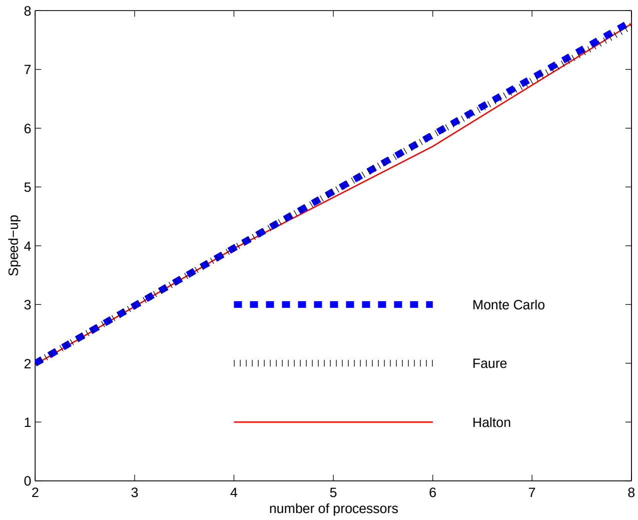

Figure 4: Speed-up values for $K = 100$

a particularly high performance computing environment, the reasons why we used it are the following: first of all, the architecture of any computers that, actually, can be defined an high performance computer is the same architecture of the used cluster, and then it constitutes a good test proof to evaluate the software; the second one is that this computational environment is more and more used by financial firms since it can achieve high speeds with a very low price-performance ratio.

Here we show the speed-up values obtained on the call Asian option only with $K = 100$ and $n = 10^5$, since the results obtained in the other cases are very similar.

It is not a surprise that both the MC and the QMC methods have a near perfect speed-up. From the execution times point of view, this means that, for example, the QMC method based on the Faure sequence with $n = 10^5$ points requires for the option valuation about 7.2 minutes on 1 processor and minus than 1 minute on 8 processors, as the execution times, reported in Table 1, show.

5. Concluding Remarks

Actually the simulation of innovative financial instruments, such as CMO, Asian options, and other derivatives, is highly complex and computational intensive,

	Methods		
P	MC	Halton	Faure
1	7.30	8.33	7.23
2	3.65	4.21	3.62
4	1.84	2.10	1.82
6	1.24	1.46	1.23
8	0.93	1.0	0.93

Table 1: Execution times in minutes on P processors for $K = 100$ and $n = 10^5$

putting more demands on computational speed and efficiency. The Monte Carlo approach has proved to be a valaublе and flexible computational tool in modern finance. In this paper we proposed a parallel mathematical software to evaluate discrete arithmetic Asian options, based on MC and QMC methods. The numerical experiments, carried out on a cluster of PC, seem to show that the QMC methods outperform the crude MC method, even if this improvement is not so far when the dimension of the integral increases. Different variance reduction techniques have been developed to speed-up the convergence of MC method. Two of the classical variance reduction techniques are the control variate approach and the antithetic variate method. In this work we have not included such techniques, since the first our aim has been to develop the kernel of a parallel routine based on MC and QMC methods to simulate innovative financial instruments and to carried out some first experiments to evaluate its efficiency. From the efficiency point of view, all the methods used reach very high values of speed-up. Then, these results encourage to go on in such kind of work. Further, the modularity of proposed routine allows to easily introduce some variance reduction techniques, so as to use new and different low discrepancy sequences. These are some directions for future work. In conclusion, we retain that the development of a reliable and efficient parallel routine to evaluate innovative financial instruments, such as Asian options, in high performance computing environment, can be an useful service to the finance community.

Acknowledgements

I am grateful to G. Laccetti, M. Lapegna, M. Marino, and A. Murli who developed togheter with me the parallel building block for multidimensional quadra-

ture described in [16] and used in this work.

References

[1] J. Andreasen, The pricing of discretely sampled Asian and lookback options: A change of numeraire approach, *Journal of Computational Finance*, **2**, No. 1 (1998), 5-30.

[2] P. Boyle, M. Broadie, P. Glasserman, Monte Carlo methods for security pricing, In: *Monte Carlo Methodologies and Application for Pricing and Risk Management* (Ed: Bruno Depire), Risk Books, (1998), 15-44.

[3] P. Bratley, B. Fox, H. Niederreiter, Implementation and tests of low-discrepancy sequences, *ACM Transactions on Modeling and Computer Simulation*, **2** (1992), 195-213.

[4] M. Broadie M., P. Glasserman, Estimating security price derivatives using simulation, *Management Science*, **42** (1996), 269-285.

[5] R. Caflisch, W. Morokoff, A. Owen, Valuation of mortgage-backed securities using Brownian bridges to reduce effective dimension, *The Journal of Computational Finance*, **1** (1998), 27-46.

[6] A. Carverhill, L. Clewlow, Flexible convolution, *Risk*, **3**, No. 4 (1990), 25-29.

[7] P. Chalasani, S. Jha, F. Egriboyan, A. Varikooty, A refined binomial lattice for pricing American Asian options, In: *Proc. 8-th Annual Derivatives Securities Conference*, Boston (1998).

[8] J. Cox, S. Ross, The Valuation of options for alternative stochastic processes, *Journal of Financial Economics*, **3** (1976), 145-166.

[9] P. Davis, P. Rabinovitz, *Methods of Numerical Integration*, Second Edition, Academic Press (1984).

[10] M. Dempster, J. Hutton, D. Richards, LP valuation of exotic American options exploiting structure, *Journal of Computational Finance*, **2**, No. 1 (1998), 61-84.

[11] M. Fu, D. Madan, T. Wang, Pricing continuous Asian options: A comparison of Monte Carlo and Laplace transform methods, *Journal of Computational Finance*, **2**, No. 2 (1998/9), 49-74.

[12] H. Geman, M. Yor, Bessel processes, Asian options and perpetuities, *Mathematical Finance*, **3** (1993), 349-375.

[13] A. Kemna, E. Vorst, A pricing method for option based average values, *Journal of Banking and Finance*, **14** (1990), 113-129.

[14] T. Klassen, Simple, fast an flexible pricing of Asian options, *Journal of Computational Finance*, **4**, No. 3 (2001), 89-124.

[15] C. Joy, P. Boyle, K. Tan, Quasi-Monte Carlo methods in numerical finance, In: *Monte Carlo Methodologies and Application for Pricing and Risk Management*, (Ed: Bruno Depire), Risk Books, (1998), 269-280.

[16] G. Laccetti, M. Lapegna, M. Marino, A. Murli, F. Perla, High performance mathematical software in finance: a case study, *Journal of Applied Mathematics*, **2**, No. 4 (2000), 441-457.

[17] P. L'Ecuyer, Efficient and portable combined random number generators, *Communications of the ACM*, **31** (1988), 742-774.

[18] E. Levy, S. Turnbull, Average intelligence, *Risk*, **5**, No. 2 (1992), 53-59.

[19]] H. Niederreiter, *Random Number Generation and Quasi Monte Carlo Methods*, SIAM (1992).

[20] S. Paskov, New methodologies for valuing derivatives, In: *Mathematics of Derivative Securities* (Ed-s: S. Pliska, M. Dempster), Isaac Newton Institute/Cambridge University Press, (1996).

[21] S. Paskov, J. Traub, Faster valuation of financial derivatives, *Journal of Portfolio Management*, Fall (1995), 113-120.

[22] L. Rogers, Z. Shi, The value of an Asian option, *Journal of Applied Probability*, **32** (1995), 1077-1088.

[23] E. Schwartz, W. Torous, Prepayment and the valuation of mortgage-backed securities, *Journal of Finance*, **44** (1989), 375-92.

[24] I. Sloan, H. Wozniakowsky, When are Quasi Monte Carlo algorithms efficient for high dimensional integrals? *Journal of Complexity*, **14** (1998), 1-33.

[25] S. Turnbull, L. Wakeman, A quick algorithm for pricing European average options, *Journal of Financial and Quantitative Analysis*, **26** (1991), 377-389.

[26] R. Zvan, P. Forsyth, K. Vetzal, Robust numerical methods for PDE models of Asian options, *Journal of Computational Finance*, **1**, No. 2 (1997/8), 39-78.

[27] R. Zvan, P. Forsyth, K. Vetzal, Discrete Asian barrier options, *Journal of Computational Finance*, **3**, No. 1 (1999), 41-69.