International Journal of Theoretical and Applied Finance

Localized Monte Carlo Algorithm to Compute Prices of Path Dependent Options on Trees

Sebastian E. Ferrando.

Department of Mathematics, Physics and Computer Science,
Ryerson University, 350 Victoria St.,
Toronto, Ontario M5B 2K3, Canada.
ferrando@ryerson.ca

Ariel J. Benal.

Department of Mathematics, Mar del Plata National University,
Funes 3350, Mar del Plata, 7600, Argentina.
abernal@hotelcostagalana.com

Received (Day Month Year)
Revised (Day Month Year)

A new simulation based algorithm to approximate prices of path dependent European options is introduced. The algorithm is defined for tree-like approximations to the underlying process and makes extensive use of structural properties of the discrete approximation. We indicate the advantages of the new algorithm in comparison to standard Monte Carlo algorithms. In particular, we prove a probabilistic error bound that compares the quality of both approximations. The algorithm is of general applicability and, for a large class of options, it has the same computational complexity as Monte Carlo.

Keywords: Exotic Options; Monte Carlo; Binomial Trees.

1. Introduction

The Monte Carlo (*mc*) technique is an important computational technique in finance (see, for example, [9] and [3]), in particular it can be used to estimate prices of exotic options. Two key properties of this technique are: its general applicability and the fact that error bounds are readily available. Due to its slow convergence several speed up alternatives/improvements have also been extensively studied. In this paper we present a general way of improving *mc* to approximate prices of path dependent European options. Our approach is based on exploiting the lattice structure of certain tree approximations to the underlying diffusion process. For simplicity, we stay in the one dimensional case but indicate possible generalizations. In the literature there are several numerical procedures to approximate prices of exotic options, in general tree approximations have not been used for this task because of the exponential explosion in the number of paths. There are exceptions to this last remark, for example, the technique described in [8] is a dynamic programming technique, based on the space discretization of the state variables, that approximates the exact value on the binomial tree. There are other examples of the use of binomial trees tailored to particular class of options ([10], [5]). Our technique is completely different from these approaches, in particular, it offers a convenient way to estimate the error. On the other hand, our technique does not readily apply to American options.

Let n be the number of time steps discretizations, X will denote the payoff of a given option and $\mathbf{E}_n(X)$ the expectation (with respect to probability measure $p(\ )$) on the tree approximation. Assuming constant interest rate, the price in the continuos model will be given by the following limit

$$V_{t_0}(X) = e^{-r(t-t_0)} \lim_{n\to\infty} \mathbf{E}_n(X). \tag{1.1}$$

There are two problems with this approximation, the first one is how to compute $\mathbf{E}_n(X)$ efficiently. The issue being that the discrete probability space is of size 2^n and a naive Monte Carlo algorithm on the tree converges too slowly. We show by using examples that our main new algorithm reduces the standard error of the Monte Carlo algorithm by orders of hundreds. This is achieved, for a large class of options, without increasing the computational complexity. We also prove probabilistic inequalities to support this excellent performance. These facts make our techniques readily useful to compute efficiently and with error bounds the exact price in the discrete model. The second problem implicit in (1.1) is how large we should take n to get a good approximation to the left hand side. This problem has been only recently thoroughly treated in the literature ([7], [13] and [18]) for the case of path independent options. In addition, there are several proposals to speed up convergence for specific classes of path dependent options (see, for example, [4], [16], [17]). We do not study this problem directly in our paper, but given its crucial role we perform numerical experiments and indicate how our ideas could be combined with some specific speed ups treated in the literature.

This article is organized as follows, in Sections 2 and 3 we present two algorithms the second one builds on the first one and represents our main contribution. These algorithms are presented in the context of binomial trees. Section 4 presents many numerical examples some of them related to the speed of convergence of tree approximations to the continuos model. Section 5 elaborates some computational issues. Section 6 indicates generalizations beyond the binomial tree setting and indicates how to combine our new algorithm with improved lattice methods described in the recent literature. Section 7 summarizes the main features of our contributions. Finally, Section Appendix A proves a result needed in the paper and, for completeness, presents the expressions for the estimates of the standard errors corresponding to the new algorithms.

2. Localized Monte Carlo on Binomial Trees

For a fixed n let $w = \{w_0, w_1, \ldots, w_{n-1}\}$ be a path with $w_i \in \{d, u\}$ where $0 < d < 1 < u$, also let Ω be the space of all such paths. Ω is assumed to be a probability space with probability measure $p(\ )$ (a more precise notation would be $p_n(\ )$, but we will not use it for simplicity). The underlying process is $\mathbf{S}(w) = \{S_0, S_1, \ldots, S_n\}$, with S_0 fixed beforehand and $S_{i+1} = S_i\ w_i$, we may use the notation $S_i(w)$ when convenient. We will also need the following notation, for a given $A \subseteq \Omega$ define

$$\mathbf{E}_{n,A}(X) = \frac{1}{p(A)} \sum_{w \in A} X(w)p(w), \tag{2.1}$$

we set $\mathbf{E}_n(X) = \mathbf{E}_{n,\Omega}(X)$ and $p_A(w) = p(w)/p(A)$. In the present context, payoffs for path dependent european options are nonnegative functions $X : \Omega \to \mathbb{R}$. For example, the payoff function for an average asian strike is

$$X(w) = (S_n(w) - Average(w))_+, \tag{2.2}$$

$$Average(w) = \frac{1}{n+1} \sum_{k=0}^{n} S_k(w). \tag{2.3}$$

By defining the probability $p(\ )$ appropriately, the price in the continuos model $V_{t_0}(X)$ at time t_0 can be obtained by taking the limit of the following quantity,

$$V_{t_0}(X, n) = e^{r\ (t-t_0)} \mathbf{E}_n(X), \tag{2.4}$$

under the hypothesis of constant interest rates r. The Monte Carlo algorithm on binomial trees (*mcbt*) is based directly on the law of large numbers

$$\mathbf{E}_n(X) = \lim_{m\to\infty} \frac{1}{m+1} \sum_{q=0}^{m} X(w^q), \tag{2.5}$$

where the paths $w^q = \{w_0^q, w_1^q, \ldots, w_{n-1}^q\}$ are independent samples from $p(\ )$. For later reference, it will be useful to formalize this last statement; to achieve this end, assume there are independent random variables U_n, defined on $([0,1]^{\mathbb{N}}, \lambda)$ (where $\mathbb{N} = \{0, 1, 2, \ldots\}$), uniformly distributed with respect to the Lebesgue product probability measure λ. In particular, we could take U_n to be the coordinate projections

$$U_k : [0,1]^{\mathbb{N}} \to [0,1] \text{ where } U_k(x) = x_k \text{ and } x = \{x_0, x_1, \ldots, x_k, \ldots\}. \tag{2.6}$$

We will use W to denote a generic random variable implicit in the sampling of the paths. More formally

$$W : [0,1]^n \to \Omega \text{ and } \lambda(W^{-1}(A)) = p(A) \text{ for any } A \subseteq \Omega. \tag{2.7}$$

For given $x \in [0,1]^{\mathbb{N}}$, setting $w^q = W(U_{qn}(x), \ldots, U_{(q+1)n-1}(x))$, (2.5) becomes

$$\mathbf{E}_n(X) = \lim_{m\to\infty} \frac{1}{m+1} \sum_{q=0}^{m} X(W(U_{qn}(x), \ldots, U_{(q+1)n-1}(x))), \ \textit{a.e.} \text{ on } [0,1]^{\mathbb{N}}. \tag{2.8}$$

Let $\Omega^i = \{w \ : \ n_u(w) = i\}$ where $n_u(w) = |\{k : \ 0 \leq k < n, \ w_k(w) = u\}|$ and $|C|$ indicates the cardinality of a set C. The *localized monte carlo* algorithm (to be defined shortly) uses the obvious decomposition

$$\mathbf{E}_n(X) = \sum_{i=0}^{n} p(\Omega^i) \ \mathbf{E}_{n,\Omega^i}(X). \tag{2.9}$$

Similarly as we did above, sampling in Ω^i is formalized by the use of a generic random variable

$$W^i : [0,1]^n \rightarrow \Omega^i \ \subseteq \Omega \text{ and } \lambda((W^i)^{-1}(A)) = p_{\Omega^i}(A) = \frac{p(A)}{p(\Omega^i)} \text{ for any } A \subseteq \Omega^i. \tag{2.10}$$

For given $x \in [0,1]^{\mathbb{N}}$ we set $w^{q,i} = \{w_0^{q,i}, \ldots w_{n-1}^{q,i}\} = W^i(U_{qn}(x), \ldots, U_{(q+1)n-1}(x))$. We now give a precise description of the *localized monte carlo algorithm* (*lmc*):

• **Inputs**: n, m (number of simulations).

(1) Compute $m_i = m \times p(\Omega^i), \ \ i = 0, \ldots, n$.

(2) Loop over the $n+1$ sets Ω^i, in each of them sample paths independently, $w^{q,i} \in \Omega^i$ $q = 1, \ldots, m_i$. For each $w^{q,i}$ compute the monte carlo estimator on Ω^i

$$A_{mc,\Omega^i}(X, m_i) = \frac{1}{m_i} \sum_{q=1}^{m_i} X(w^{q,i}). \tag{2.11}$$

(3) Finally compute the *lmc* estimator on Ω by

$$A_{lmc,\Omega}(X, m) = \sum_{i=0}^{n} p(\Omega^i) \ A_{mc,\Omega^i}(X, m_i). \tag{2.12}$$

Alternative versions of the algorithm are discussed in Section 6. For completeness recall Chebychev's inequality for the *mcbt* estimator, let $\epsilon > 0$:

$$p(|\frac{1}{m} \sum_{q=1}^{m} X(w_q) - \mathbf{E}_n(X))| \geq \epsilon) \leq \frac{var_n(X)}{m \ \epsilon^2}. \tag{2.13}$$

An upper bound for the error of the *lmc* estimator is given by the following.

Proposition 2.1. *We have the following basic facts*

$$\lim_{m \rightarrow \infty} A_{lmc,\Omega}(X, m) = \mathbf{E}_n(X) \tag{2.14}$$

and the estimator is unbiased, that is $\mathbf{E}_n(A_{lmc,\Omega}(X, m)) = \mathbf{E}_n(X)$. *Moreover, for any* $\epsilon > 0$ *we have the following bound for the error,*

$$p(|A_{lmc,\Omega}(X, m) - \mathbf{E}_n(X)| \geq \epsilon) \leq \frac{1}{m \ \epsilon^2} \sum_{i=0}^{n} \left(p(\Omega^i) \ var_{n,\Omega^i}(X)\right) = \tag{2.15}$$

$$\frac{1}{m\ \epsilon^2}\left(var_n(X) - \frac{1}{2}\sum_{i=0}^{n}\sum_{j=0}^{n}\left[p(\Omega^i)\ p(\Omega^j)\ (\mathbf{E}_{n,\Omega^i}(X) - \mathbf{E}_{n,\Omega^j}(X))^2\right]\right). \tag{2.16}$$

Where $var_{n,\Omega^i}(X) = \mathbf{E}_{n,\Omega^i}((X - \mathbf{E}_{n,\Omega^i}(X))^2)$.

Proof. The convergence follows from the law of large numbers. We notice that $\mathbf{E}_n(A_{lmc,\Omega}(X,m)) = \mathbf{E}_n(X)$, we then apply Chebychev's inequality to the random variable $A_{lmc,\Omega}(X,m)$. This gives the first inequality after recolling that $m_i = m \times P(\Omega^i)$ and the fact that the random variables $X(W^i(U_{qn}(\ ),\ldots,U_{(q+1)n-1}(\ )))$ are independent and hence uncorrelated. We now notice

$$\sum_{i=0}^{n} p(\Omega^i)\ var_{n,\Omega^i}(X) = var_n(X) + \mathbf{E}_n^2(X) - \sum_{i=0}^{n} p(\Omega^i)\ \mathbf{E}_{n,\Omega^i}^2(X). \tag{2.17}$$

Finally, the equality in (2.15) follows directly by an application of (A.1) by taking in that equation $p_k = p(\Omega^k)$, $v_k = \mathbf{E}_{n,\Omega^k}(X)$ and $||v_k||^2 = v_k^2$, $k = 0,\ldots,n$ (i.e. $q = n$ in (A.1)). □

We will illustrate through numerical examples in the remaining of the paper that *lmc* offers better performance than *mcbt*. It follows from the above result that the reason for this improvement is that the options considered have small variances $\mathrm{var}_{n,\Omega^i}(X)$. The computational complexities of these two algorithms are the same.

3. Cyclic Shift Algorithm on Binomial Trees

Here we introduce our main contribution, the *cyclic shift* (*cs*) algorithm. It offers better error performance than *lmc* and for a class of options, to be described in Section 5, the number of function evaluations is of the same order as for *lmc* and *mcbt*. We now embark on a series of definitions and propositions, some of these will form the background of the *cs* algorithm. We provide partial arguments and references for the simple proofs; the results are specializations of well known constructions in ergodic theory and combinatorics.

Definition 3.1. A measurable transformation τ on a general measure space $(\Omega, \mathcal{F}, p)$ is called measure preserving if $p(\tau^{-1}A) = p(A)$ for all $A \in \mathcal{F}$.

In the present section, all sigma algebras $\mathcal{F}$ considered will be the power set of a given finite base space, hence we will not mention $\mathcal{F}$ explicitly. Moreover, in this setting, the condition $p(\tau w) = p(w)$ is equivalent to τ being measure preserving.

Definition 3.2. Define the circular *left* shift τ_c (shift on a discrete circle) acting on a path $w = (w_0,\ldots,w_{n-1})$ as follows

$$\tau_c(w) = (w_1, w_2, \ldots, w_{n-1}, w_0). \tag{3.1}$$

We denote with τ_c^k the composition of τ_c with itself k-times. Notice that this transformation is invertible, τ_c^{-1} is called the right shift.

To define the *cs* algorithm, we will need to assume that τ_c preserves the given probability measure on the tree. We will state explicitly when this condition is nedeed to clarify its role.

Definition 3.3. For any $w \in \Omega^i$ define the *orbit* of w by

$$\Omega^i_w = \{\tau_c^k w : k \in \mathbb{Z}\}. \tag{3.2}$$

Also, define θ^i_w to be the smallest positive integer such that $\tau_c^{\theta^i_w} w = w$

Lemma 3.1.

$$\Omega^i_w = \Omega^i_{w'} \text{ for any } w' \text{ in } \Omega^i_w \tag{3.3}$$

$$\Omega^i_w = \{\tau_c^k w : 0 \leq k < \theta^i_w\}. \tag{3.4}$$

$$|\Omega^i_w| = \theta^i_w = \theta^i_{w'} \text{ for any } w' \text{ in } \Omega^i_w, \tag{3.5}$$

$$\theta^i_w \times |\{\tau_c^k : \; \exists\, k;\; 0 \leq k < n, \tau_c^k w = w\}| = n. \tag{3.6}$$

Proof. The statements above are simple, well known, properties of orbits. See for example [14]. □

Proposition 3.1. *For each i the set Ω^i satisfies the following properties: there are unique sets Ω^i_j and integers J_i which satisfy $\Omega^i_j \subseteq \Omega^i$, $j = 1, \ldots, J_i$ and*

$$\Omega^i = \cup_j \Omega^i_j, \tag{3.7}$$

where

$$\Omega^i_j \cap \Omega^i_j = \varnothing \text{ if } j \neq j'. \tag{3.8}$$

Moreover,

$$\tau_c^k \Omega^i_j = \Omega^i_j \text{ for all } k \in \mathbb{Z}. \tag{3.9}$$

Proof. We first notice that $\Omega^i_w \cap \Omega^i_{w'} = \emptyset$ if $w \notin \Omega^i_{w'}$. Therefore we define the sets Ω^i_j as the collection of distinct orbits out of the collection of all orbits $\Omega^i_w, w \in \Omega^i$ □

For convenience we use the notation $\theta^i_j = |\Omega^i_j|$, this notation reflects the fact that θ^i_w is indepedent of $w \in \Omega^i_j$, this is due to (3.5).
The following proposition will be crucial for the definition of the *cs* algorithm.

Proposition 3.2. *Assume $p(\tau_c w) = p(w)$, then*

$$\mathbf{E}_{n,\Omega^i_j}(X) = \frac{1}{|\Omega^i_j|} \sum_{w \in \Omega^i_j} X(w) = \tag{3.10}$$

$$\frac{1}{\theta^i_j} \sum_{k=0}^{\theta^i_j - 1} X(\tau_c^k w) = \frac{1}{n} \sum_{k=0}^{n-1} X(\tau_c^k w). \tag{3.11}$$

Proof. First notice that $p(\tau_c w) = p(w)$ implies $p(\ )$ is constant on each Ω^i_j, hence for each $w \in \Omega^i_j$,

$$\frac{p(w)}{p(\Omega^i_j)} = \frac{1}{|\Omega^i_j|}. \tag{3.12}$$

This equation gives the first equality in (3.10). The second equality in (3.10) follows from (3.4) and (3.5). The third equality in (3.10) follows from (3.4) and (3.5) and (3.6). □

Is is important to notice that the above results can be seen from a different and more general perspective, to this end, lets introduce the following definition.

Definition 3.4. A measure preserving transformation τ is called ergodic if for all $A \in \mathcal{F}$ such that $\tau^{-1}A = A$ we have $A = \emptyset$ or $A = \Omega$.

The following Proposition follows easily from our definitions.

Proposition 3.3. *Assume τ_c is mesure preserving, then τ_c restricted to each probability space $(\Omega^i_w, p_{\Omega^i_w})$ is ergodic.*

The following is the well known Birkhoff theorem ([19]). In the present context it follows easily from Proposition 3.

Theorem 3.1. *For any ergodic transformation τ on a probability space $(\Omega, \mathcal{F}, p)$ and integrable function X we have*

$$\mathbf{E}(X) = \lim_{m\to\infty} \frac{1}{m} \sum_{k=0}^{m-1} X(\tau^k w) \text{ almost everywhere in } w. \tag{3.13}$$

In our present finite context *almost everywhere* is equivalent to *everywhere.* Notice that τ_c is not ergodic on the set Ω^i, by means of Proposition 3 ?????? we have decomposed this set into the ergodic components (see [19]) Ω^i_j and used ergodicity to compute the mean values $\mathbf{E}_{n,\Omega^i_j}(X)$.
Define the following function $X_s : \Omega \to \mathbb{R}$,

$$X_s(w) = \mathbf{E}_{n,\Omega^i_j}(X) = \frac{1}{n} \sum_{k=0}^{n-1} X(\tau^k_c w), \quad \text{if } w \text{ in } \Omega^i_j, \tag{3.14}$$

where we have made use of Proposition 3. Notice that X_s is well defined because of Proposition 3. In short, X_s is constant in each Ω^i_j and it is obtained by smoothing X.
We are now in a position to define the *cyclic shift (cs)* algorithm as the *lmc* algorithm applied to X_s. From now on we assume $p(\tau_c w) = p(w)$. Here are the details:

- **Inputs**: n, m (number of simulations).

(1) Compute $m_i = m \times p(\Omega^i)$, $i = 0, \ldots, n$.

(2) Loop over the $n+1$ sets Ω^i, in each of them sample paths independently, $w^{q,i} \in \Omega^i$ $q = 1, \ldots, m_i$. For each $w^{q,i}$ compute a *mc* estimator on Ω^i but this time for X_s,

$$A_{mc,\Omega^i}(X_s, m_i) = \frac{1}{m_i} \sum_{q=1}^{m_i} X_s(w^{q,i}). \tag{3.15}$$

(3) Finally compute the estimator on Ω by

$$A_{cs,\Omega}(X, m) = \sum_{i=0}^{n} p(\Omega^i)\ A_{mc,\Omega^i}(X_s, m_i). \tag{3.16}$$

Theorem ref algorithmJustification ???? allow us to compare the probabilistic error bounds for the *mcbt*, *lmc* and *cs* estimators. We will first need the following notation,

$$r_i = \frac{1}{2} \sum_j \frac{p(\Omega^i_j)}{|\Omega^i_j|^2} (\sum_{w \in \Omega^i_j} (\sum_{w' \in \Omega^i_j} (X(w) - X(w'))^2)). \tag{3.17}$$

The next lemma collects the relevant properties we will use from X_s.

Lemma 3.2.

$$\mathbf{E}_{n,\Omega^i}(X_s) = \mathbf{E}_{n,\Omega^i}(X), \tag{3.18}$$

$$\mathbf{E}_{n,\Omega^i}(X_s^2) = \mathbf{E}_{n,\Omega^i}(X^2) - \frac{r_i}{p(\Omega^i)}. \tag{3.19}$$

Proof. Equality (3.18) follows by noticing that the sets Ω^i_j are a disjoint covering of Ω^i and the fact that X_s is defined as the mean of X over each Ω^i_j. For (3.19) we argue as follows

$$\mathbf{E}_{n,\Omega^i}(X_s^2) = \frac{1}{p(\Omega^i)} \sum_j \sum_{w \in \Omega^i_j} p(w)\ X_s^2(w) = \frac{1}{p(\Omega^i)} \sum_j p(\Omega^i_j) \left(\frac{\sum_{w \in \Omega^i_j} X(w)}{|\Omega^i_j|} \right)^2 = \tag{3.20}$$

$$\frac{1}{p(\Omega^i)} \sum_j p(\Omega^i_j) \left(\frac{1}{|\Omega^i_j|} \sum_{w \in \Omega^i_j} X^2(w) - \frac{1}{2|\Omega^i_j|^2} \sum_{w \in \Omega^i_j} \sum_{w' \in \Omega^i_j} (X(w) - X(w'))^2 \right) = \tag{3.21}$$

$$\mathbf{E}_{n,\Omega^i}(X^2) - \frac{r_i}{p(\Omega^i)}. \tag{3.22}$$

Where, for each fixed Ω^i_j, we have made use of (A.1) by taking in that equation $p_k = p_w = \frac{1}{|\Omega^i_j|}$, $v_k = v_w = X(w)$. □

Theorem 3.2. *We have the following basic facts*

$$\lim_{m \to \infty} A_{cs,\Omega}(X, m) = \mathbf{E}_n(X). \tag{3.23}$$

and the estimator is unbiased, that is $\mathbf{E}(A_{cs,\Omega}(X,m)) = \mathbf{E}(X)$*. Moreover, for any* $\epsilon > 0$ *we have the following bound for the error,*

$$p(|A_{cs,\Omega}(X,m) - \mathbf{E}_n(X)| \geq \epsilon) \leq \tag{3.24}$$

$$\frac{1}{m\ \epsilon^2}\left(var_n(X) - \frac{1}{2}\sum_{i=0}^{n}\sum_{j=0}^{n}\left[p(\Omega^i)\ p(\Omega^j)\ (\mathbf{E}_{n,\Omega^i}(X) - \mathbf{E}_{n,\Omega^j}(X))^2\right] - \sum_{i=0}^{n} r_i \right). \tag{3.25}$$

Proof. Equation (3.23) follows from the law of large numbers and (3.18). We notice $\mathbf{E}(A_{cs,\Omega}(X,m)) = \mathbf{E}(X)$, this and Chebychev's inequality gives

$$p(|A_{cs,\Omega}(X,m) - \mathbf{E}_n(X)| \geq \epsilon) \leq \frac{1}{m\ \epsilon^2}\sum_{i=0}^{n}\left(p(\Omega^i)\ var_{n,\Omega^i}(X_s)\right).$$

Now using (3.19) and (3.18) we obtain

$$\sum_{i=0}^{n} p(\Omega^i)\ var_{n,\Omega^i}(X_s) = \sum_{i=0}^{n} p(\Omega^i)\ (\mathbf{E}_{\Omega^i}(X^2) - \frac{r_i}{p(\Omega^i)} - \mathbf{E}^2_{\Omega^i}(X)) = \tag{3.26}$$

$$var_n(X) + \mathbf{E}_n^2(X) - \sum_{i=0}^{n} p(\Omega^i)\ \mathbf{E}^2_{n,\Omega^i}(X) - \sum_{i=0}^{n} r_i. \tag{3.27}$$

We complete the proof by the same argument as the one used below (2.17). □

One could try to study the asymptotic behaviour, as a funtion of the number of simulated paths and/or n, of the above error bounds for specific options. In practice, one can easily estimate the variance of the cs, this is described in Section Appendix A. In Section 5 we indicate the time complexity of this algorithm.

4. Numerical experiments and Continuous Limit

We now offer some computational examples that indicate the quality of the approximations to $\mathbf{E}_n(X)$ offered by *mbct, lmc* and *cs*. For later comparison with standard discretizations of stochastic differential equations all of our numerical examples will consider the standard ([12]) binomial approximation to the Black Scholes model. This binomial model consists of,

$$u = e^{\sigma\sqrt{dt}} = d^{-1} \text{ and } dt = \frac{(t-t_0)}{n}, \tag{4.1}$$

and

$$p_d = \frac{u - e^{rdt}}{u-d},\ p_u = 1 - p_d. \tag{4.2}$$

We then have that $p(\ )$ is the product probability and

$$p(\Omega^i) = \frac{n! p_u^i p_d^{n-1}}{(n-i)!i!}, \tag{4.3}$$

this last expression can easily be computed numerically avoiding round off errors. Tl

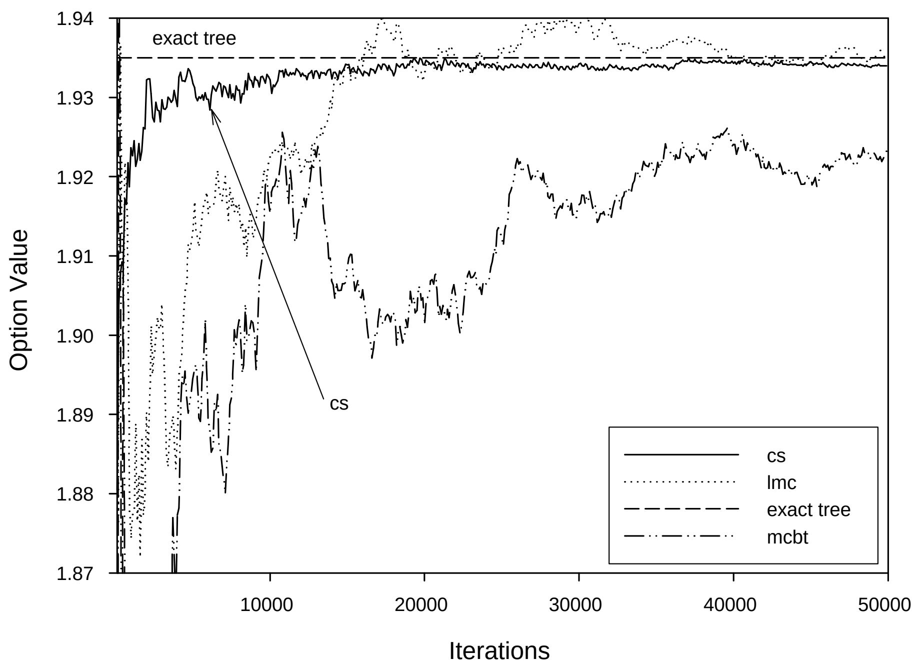

Fig. 1. *lmc, mcbt (LLAMADO BMC EN EL SOFT QUE TE MANDE), cs and exact value on tree* Here we compare *lmc, mcbt* and *cs* against the exact value on tree for the average strike. Values of parameters: $S_0 = 50$ $r = 0.05$ $\sigma = 0.2$ $n = 19$ $T = 0.5$.

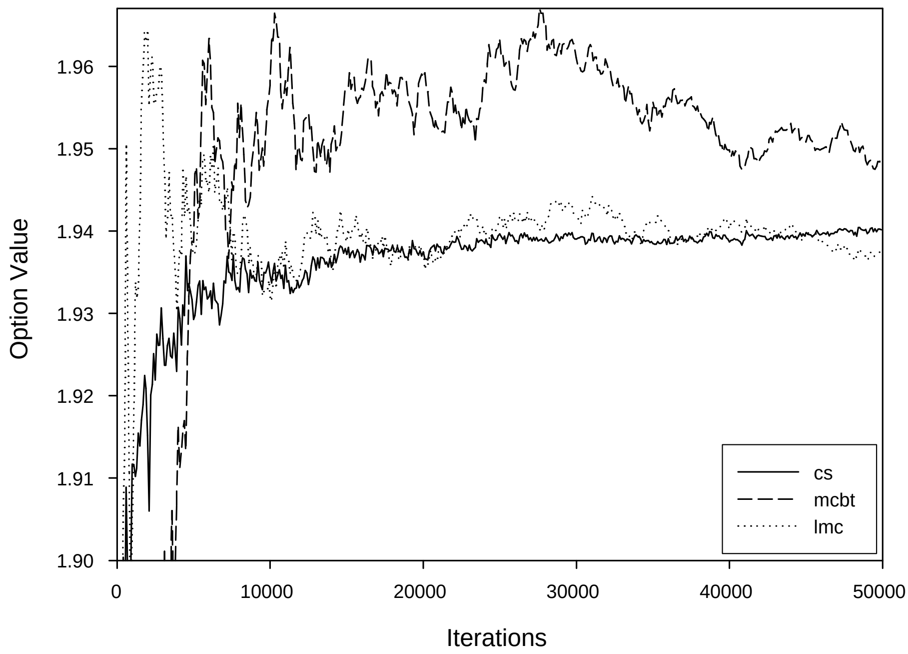

Fig. 2. *lmc, mcbt* and *cs* Here we compare *lmc, mcbt* and *cs* against each other. Values of parameters: $S_0 = 50$ $r = 0.05$ $\sigma = 0.2$ $n = 157$ $T = 0.5$.

As we mentioned in the introduction we do not offer any theoretical results for the speed of convergence of the binomial tree approximation to the Black and Scholes continuous limit (REFERENCES FOR THE PATH INDEPENDENT CASE). We do present some numerical evidence that our algorithms outperform the basic application of Monte Carlo to the basic discretization of the stochastic differential equation (REFERENCE). We refer to this application of Monte Carlo as the *cmc* (continuos Monte Carlo) algorithm. We implemented *cmc* with the following standard discretization

$$S_{t_{i+1}}(x) = S_{t_i}(x)\ e^{\nu dt - \sigma\sqrt{dt} Y(x_i)} \tag{4.4}$$

where $Y \sim \mathcal{N}(0,1)$ and $\nu = r - \sigma^2/2$.
We will compare our techniques and *cmc* for cases where the exact continuos values are known. For this reason we introduce the *fixed strike average option*, whith payoff given by:

$$X(w) = (Average(w) - K)_+, \tag{4.5}$$

where K is a fixed strike value. We also introduce the *lookback European call option* with payoff:

$$X(w) = (S_n(w) - m(w)_+, \tag{4.6}$$

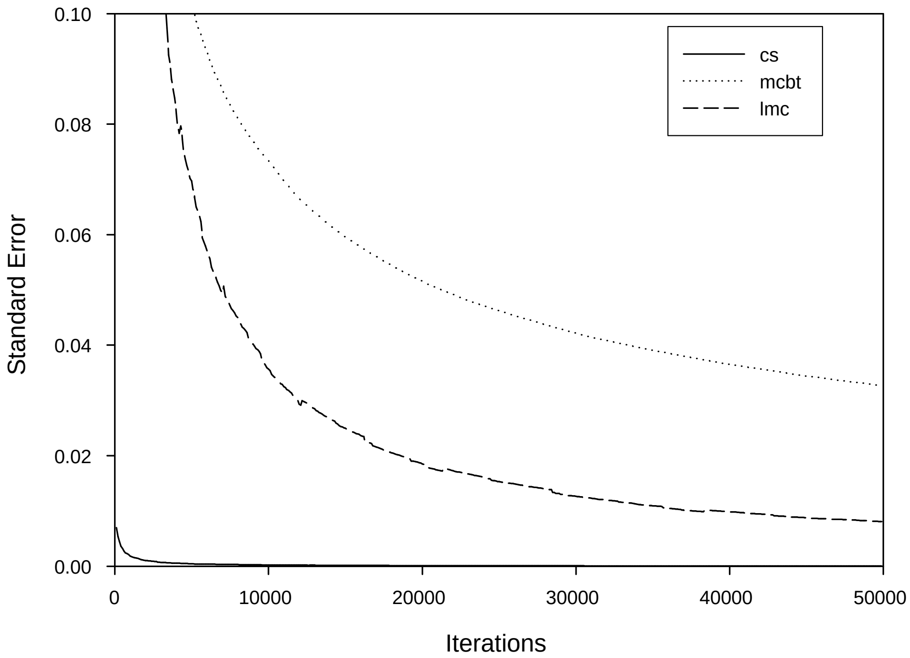

Fig. 3. Standard errors for *lmc, mcbt* and *cs* Values of parameters (as in previous figure): $S_0 = 50$ $r = 0.05$ $\sigma = 0.2$ $n = 157$ $T = 0.5$.

$$m(w) = \min_{0 \leq k \leq n} S_k(w). \tag{4.7}$$

1

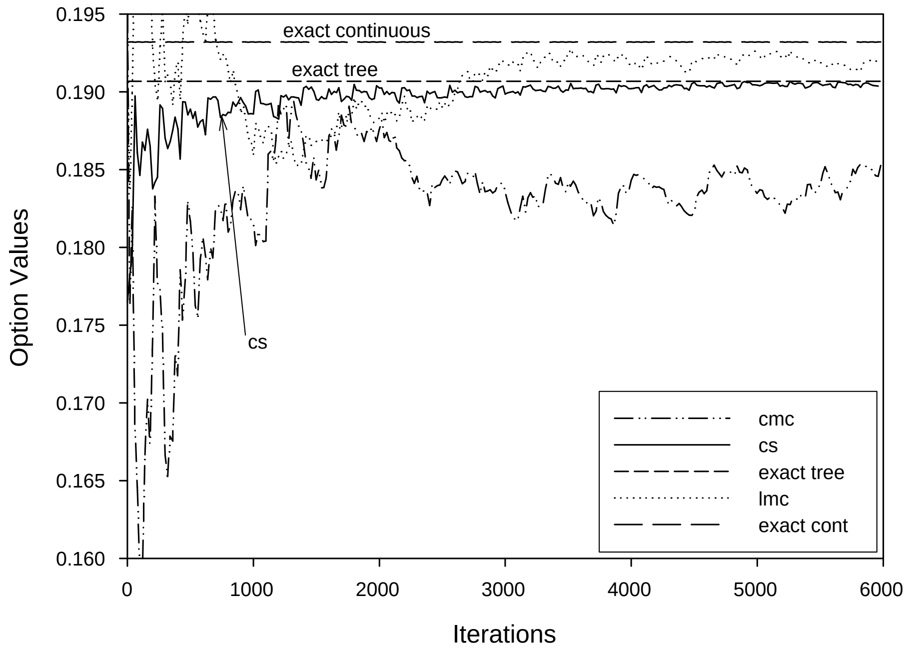

Fig. 4. *lmc, cs, cmc* and exact value on Black and Scholes. Here we compare *lmc, cs* and *cmc* against the exact value on the continuous model for fixed strike average European call. Values of parameters: $S_0 = 1.9$, $K = 2$, $r = 0.05$, $\sigma = 0.5$, $n = 7$, $T = 1.0$. The exact value (in the continuos model) para estos parametros es 0.193174, no tengo como ponerlo en archivo via software asi que tendras que armarlo vos.

5. Fast Updates

We present computational details on how to compute the arithmetic average and minimum value along a shifted path. These computations will readily apply to Asian options depending on the arithmetic average and to the *lookback* European call option. We basically show how the value of the payoff can be updated, after applying a shift transformation, in constant time. The conclusion will be that the *cs* algorithm requires the same order of function evaluations as *mc* and the ratios of the constants involved are very low. We will refer to this constant time updates as *fast updates*. Similar *fast updates* can be used to deal with other averages, maxima, etc., in particular one can apply the *cs* algorithm , with fast updates, to barrier options.
In this section we will use the following notation

$$S_k^j = S_0 w_0^j \; w_1^j \dots w_{k-1}^j \text{ for } k = 0, \dots, n. \tag{5.1}$$

After applying the *left* cyclic shift to w^j we obtain $\tau_c(w^j) = w^{j+1}$ where

$$w_k^{j+1} = w_{k+1}^j \text{ for } k = 0, \dots, n_1 \text{ and } w_{n-1}^{j+1} = w_0^j. \tag{5.2}$$

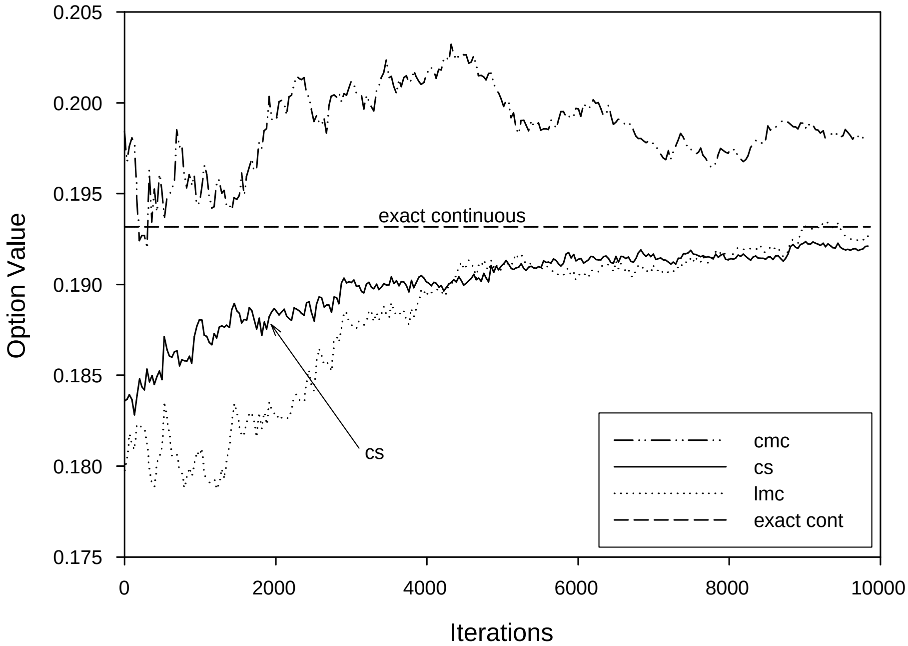

Fig. 5. *lmc, cs, cmc* and exact value on Black and Scholes. Here we compare *lmc, cs* and *cmc* against the exact value on the continuous model for fixed strike average european call. Aumento el valor de n para ver que pasa Values of parameters: $S_0 = 1.9$, $K = 2$, $r = 0.05$, $\sigma = 0.5$, $n = 157$, $T = 1.0$. The exact value (in the continuos model) para estos parametros es 0.193174, no tengo como ponerlo en archivo via software asi que tendras que armarlo vos.

After applying the *right* cyclic shift to w^j we obtain $\tau_c^{-1}(w^j) = w^{j+1}$

$$w_k^{j+1} = w_{k-1}^j \text{ for } k = 1, \ldots, n \text{ and } w_0^{j+1} = w_{n-1}^j. \tag{5.3}$$

Clearly $S_n^j = S_n$ for all j.
The fast update for the aritmetic average is obvious, let $q_j = \sum_{k=0}^n S_k^j$, then

$$q_{j+1} = \frac{1}{w_0^j}(q_j - S_0) + S_n. \tag{5.4}$$

The following proposition indicates a fast update for the minimum of the stock values along a path. For convenience let

$$m_j = \min_{0 \le k \le n} S_k^j = S_{i_j}^j, \tag{5.5}$$

so i_j is the index at which the minimum is attained.

Proposition 5.1. *For a given $i \in \{0, 1, \ldots, n\}$ let $S_n = S_0 u^i d^{n-i}$, if $S_n \le S_0$ let $w^{j+1} = \tau_c(w^j)$ and*

$$m_{j+1} = \min\left(\frac{m_j}{w_0^j}, S_n\right), \tag{5.6}$$

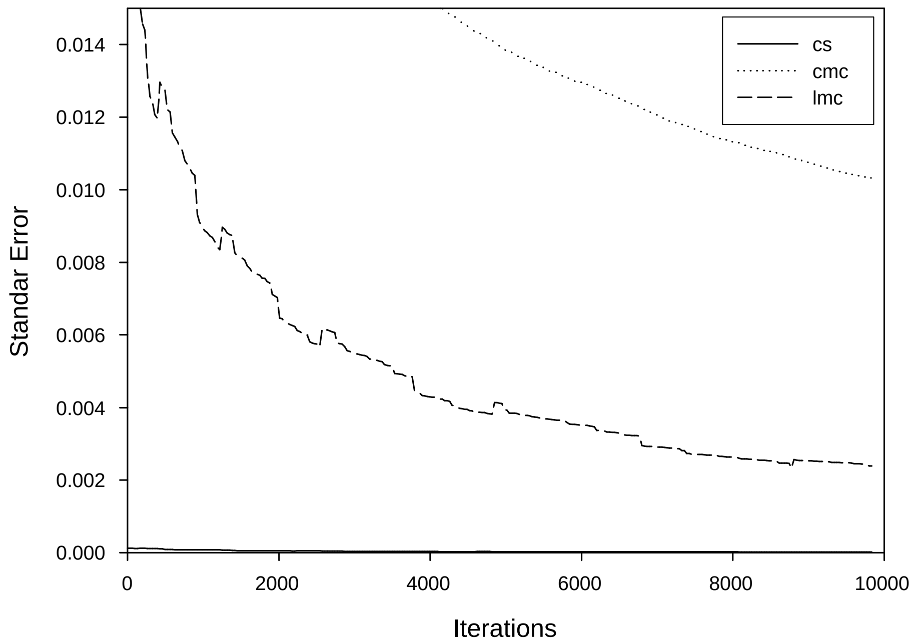

Fig. 6. Standard errors (para l asituacion de la figura anterior) for *lmc, cs, cmc* and exact value on Black and Scholes. Values of parameters (como en la figura anterior): $S_0 = 1.9$, $K = 2$, $r = 0.05$, $\sigma = 0.5$, $n = 157$, $T = 1.0$.

$$i_{j+1} = i_j - 1 \textit{ if } \frac{m_j}{w_0^j} < S_n \textit{ and } i_{j+1} = n \textit{ if } \frac{m_j}{w_0^j} \geq S_n. \tag{5.7}$$

If $S_n > S_0$ *let* $w^{j+1} = \tau_c^{-1}(w^j)$ *and*

$$m_{j+1} = \min\left(\frac{m_j}{w_{n-1}^j}, S_0\right) \textit{ and} \tag{5.8}$$

$$i_{j+1} = i_j + 1 \textit{ if } \frac{m_j}{w_{n-1}^j} < S_0 \textit{ and } i_{j+1} = 0 \textit{ if } \frac{m_j}{w_0^j} \leq S_0. \tag{5.9}$$

Proof. From the notation introduced we have

$$S_{i_j}^j \leq S_k^j \text{ for } k = 0, \ldots, n. \tag{5.10}$$

Consider first the case $S_n \leq S_0$ and notice

$$S_k^{j+1} = \frac{S_{k+1}^j}{w_0^j} \text{ for } k = 0, \ldots, n-1. \tag{5.11}$$

The inequality $S_n \leq S_0$ rules out the case $i_j = 0$, hence, consider $i_j \geq 1$, then

$$S_{i_j-1}^{j+1} \leq S_k^{j+1} \text{ for } k = 0, \ldots, n-1. \tag{5.12}$$

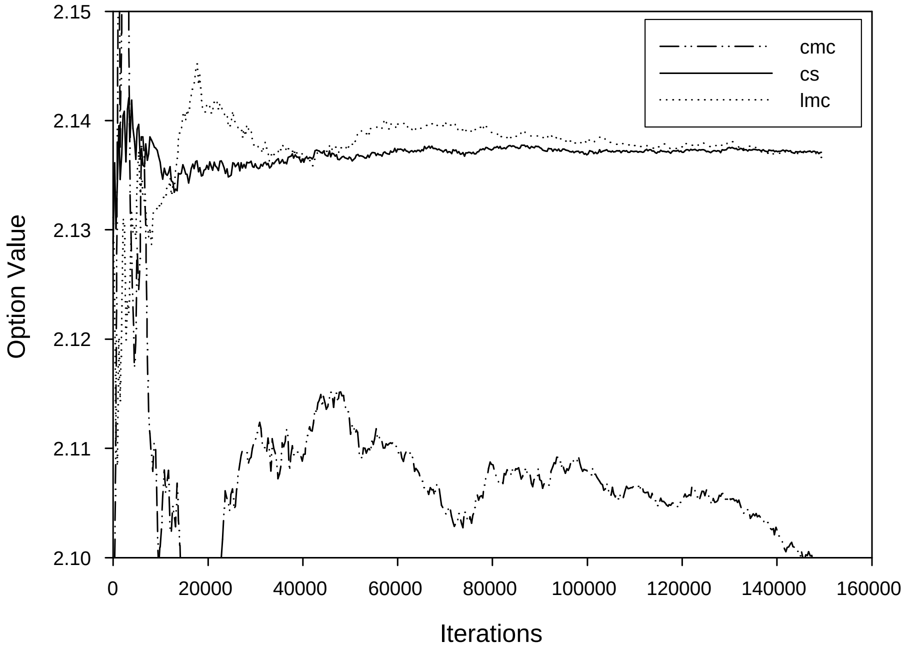

Fig. 7. *lmc, cs, cmc* and exact value on Black and Scholes. Here we compare *lmc, cs* and *cmc* against the exact value on the continuous model for the lookback european call. Cambie un poco los valores de la version anterior del paper para que no necesitemos n tan grande. Fijate que el valor exacto queda siempre arriba (ya que el minimo en el modelo continuo es menor, razonable, y por lo tanto el payoff es siempre mas grande y por lo tanto el valor de la opcion es siempre mas grande). Values of parameters: $S_0 = 10$, $r = 0.05$, $\sigma = 0.3$, $n = 19$, $T = 1.0$. The exact value (in the continuos model) para estos parametros es 2.37885. La version nueva que te estoy mandando del soft te permite calcular el valor exacto del modelo continuo del lookback, tenes que elegir 2, despues 3 y despues 5 en los "prompt" de la consola. Para que el grafico no quede tan desastroso tendras que toquetear un poco la escala en y para que el valor exacto quede un poco mas al medio

From these relationships we obtain (5.6). Consider now the case $S_n > S_0$, notice that this condition rules out $i_j = n$. We therefore assume $0 \leq i_j \leq n-1$, from the use of the right shift we obtain

$$S_{k+1}^{j+1} = \frac{S_k^j}{w_{n-1}^j} \text{ for } k = 0, \ldots, n-1. \tag{5.13}$$

Then,

$$S_{i_j+1}^{j+1} \leq S_k^{j+1} \text{ for } k = 1, \ldots, n. \tag{5.14}$$

From these relationships we obtain (5.8). □

The time complexity (CHECK WORDING) of *mcbt* and *lmc* is $n \times m$, a direct implementation of *cs* gives a time complexity of $n^2 \times m$, however, whenever *fast updates* are available, the actual time complexity of *cs* is $n \times m$. Our above compu-

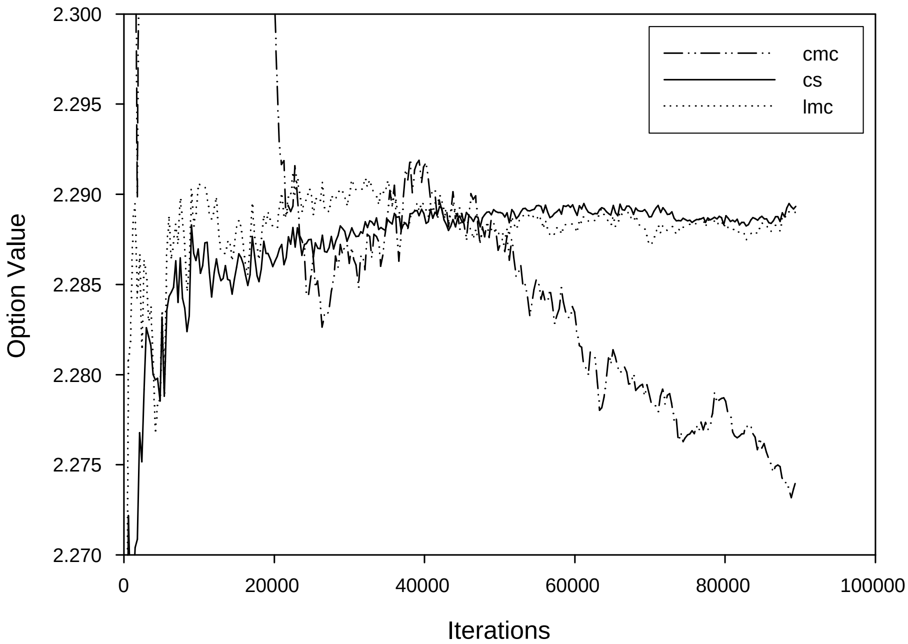

Fig. 8. *lmc, cs, cmc* and exact value on Black and Scholes. Here we compare *lmc, cs* and *cmc* against the exact value on the continuous model for the lookback european call. Los mismos parametros que en la figura anterior pero incremente n a $n = 157$.

tations and remarks show that fast updates are available for a large class of path dependent European options.

6. Extensions of the Cyclic Shift Algorithm

In this section we make clear the hypothesis needed to define a *cs* algorithm, how can be adapted to similar settings and how to combine it with improved lattice methods described in the recent literature. Moreover, we briefly describe how the idea of the algorithm originated. At a purely procedural level the *cs* algorithm needs to decompose the space of paths Ω into subsets Ω^i. A basic ingredient is to compute $p(\Omega^i)$ accurately, in the case of recombining binomial trees the number of these subsets is small, namely $n + 1$. A second key ingredient is to decompose the sets Ω^i further into subsets Ω^i_j in such a way that we can compute $\mathbf{E}_{\Omega^i_j}(X)$ fast. A more flexible way of implementing the *lmc* algorithm (and hence the *cs* algorithm) will be described now. Instead of sampling $m_i = m \times p(\Omega^i)$ paths from each Ω^i, we sample m independent paths w^q from Ω with probability $p(\ )$ and sort them according to which subset Ω^i they belong. Set $\hat{m}_i$ to be the number of paths belonging to Ω^i, if $w^{q,i}$, $q = 1, \ldots, \hat{m}_i$ are the sorted paths belonging to Ω^i, then

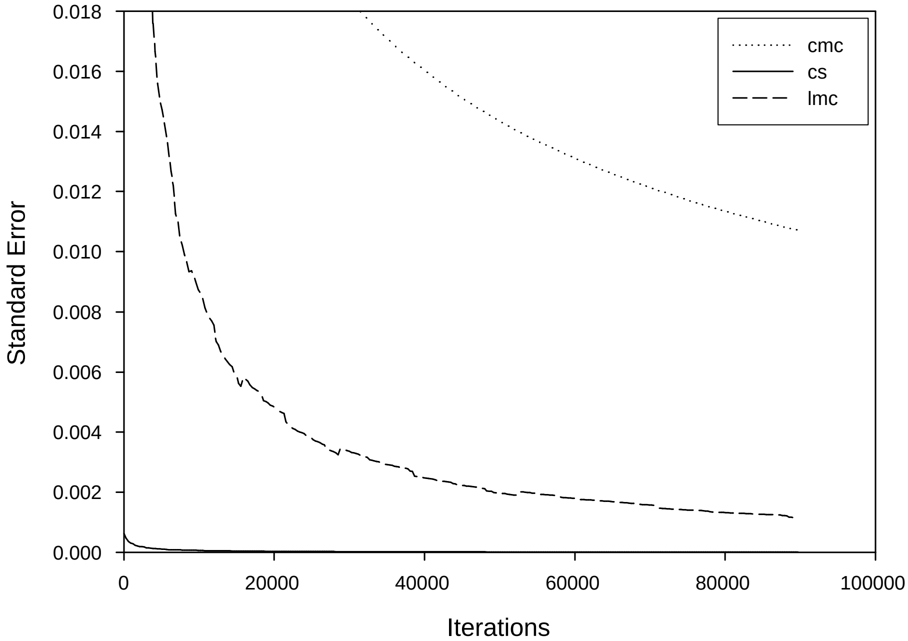

Fig. 9. Standard Errors for *lmc, cs, cmc*. Aqui irian los standard errors del caso de la figura anterior.

replace (2.12) by the following estimator

$$\hat{A}_{lmc,\Omega}(X,m) = \sum_{i=0}^{n} p(\Omega^i) \left(\frac{1}{\hat{m}_i} \sum_{q=1}^{\hat{m}_i} X(w^{q,i}) \right). \tag{6.1}$$

One should replace X by X_s for the analogous modification to the *cs* algorithm. This alternative version of the algorithm is more flexible for certain settings described below, it will incur into the extra costs of sorting m paths into the subsets Ω^i (INDICATE COST). This sorting was implicit in our description of the *cs* algorithm in the binomial setting, it did not incur into any extra omputations due to the use of a data structure (with constant access time to records) of size $n+1$.
We now indicate an extension to multi-trees (CHECK NAME, CONNECTION WITH MULTINOMIAL, BASIC DEFINITION OF PRODUCT SPACE AND BASIC PROPERTIES QUOTED) ans specialize to trinomial trees. For a given integer n the base space Ω is defined as a product space A^n where the finite probability space A is given by $A = \{r_0, r_1, \ldots, r_{b-1}\}$ with probabilities $p_k > 0$. Ω is made into a probability space through the product probability (we use $p(\ )$ to denote the product measure) and the power set sigma algebra. We use the notation $w = \{w_0, w_1, \ldots, w_{n-1}\}$ for a point (path) in Ω. We also introduce the multi-index notation $i = \{i_0, i_1, \ldots, i_{b-1}\}$ with $0 \le i_k \le n$ and $\sum_{k=0}^{b-1} i_k = n$. For a given path w we set $i(w) = \{i_0(w), i_1(w), \ldots, i_{b-1}(w)\}$, so w has a number $i_k(w)$ of children of

type k (RE-WRITE THIS). This notation allows to define, for a fixed multi-index i,

$$\Omega^i = \{w : \ i(w) = i\}. \tag{6.2}$$

We have

$$|\Omega^i| = \frac{n!}{i_0! \ i_1! \dots i_{b-1}!}. \tag{6.3}$$

The stock evolution is given by

$$S_{t_{i+1}}(w) = S_{t_i}(w) \ w_i, \tag{6.4}$$

where $w_i \in A$. To be specific we now specialize to the trinomial tree used in [4] and [16]. So we consider $b = 3$, $r_0 = d, \ r_1 = 1, \ r_2 = u$ and $p_0 + p_1 + p_2 = 1$. Setting the parameters as indicated in [4] we obtain improved convergence for a class of options including: discrete (i.e. with price fixings) barrier options, discrete lookbacks and continuos lookbacks. Our *cs* algorithm readily applies to this setting, we do notice that there are $(n+1)^2$ sets Ω^i. The *cs* algorithm still can be implemented with complexity $m \times n$ by making use of (6.1) and sorting the sampled paths w^q into the sets Ω^i by mantaining a data structure, with constant time access to records, of size $(n+1)^2$. There are several other papers dealing with speed improvements for lattice methods (see, for example, [17]) but we will not pursue these possibilities here.

Finally one could see the *cmc* algorithm as as a limit case of a high dimmensional tree, this can be done by discretizing the standard normal random variable $Y(\ )$ in (4.4). This can be done with the sampling algorithm described in [11]. EXPLAIN BETTER CONTINUE WITH OTHER EXAMPLES.

As a final remark, we mention that our original intention was to use the ergodic Bernoulli shift for computation, more specifically, the Monte Carlo algorithm can be seen as defined on the space $[0,1]^{\mathbb{N}}$ as indicated by (2.8). The key issue is that the paths w^q are independent because we shift by multiples of n. If we only shift by one unit we still have convergence,

$$\mathbf{E}_n(X) = \lim_{m\to\infty} \frac{1}{m+1} \sum_{q=0}^{m} X(W(U_q(x), \dots W(U_{q+(n-1)}(x)), \ a.e. \text{ on } [0,1]^{\mathbb{N}}, \tag{6.5}$$

the above equality holds because the ergodic theorem applies to the following ergodic transformation on $[0,1]^{\mathbb{N}}$ (called the Bernoulli shift): $\tau x = x'$ where $x'_i = x_{i+1}, i = 0, 1, \dots$. References [2] and [1] indicate advantages and strategies of how to apply this technique. Our *cs* algorithm can be seen as a way of combining the use of the shift transformation and making use of the *fast updates* available on the tree approximations.

7. Discussion

We introduced a Monte Carlo algorithm on tree approximations which makes use of the discrete tree structure to obtain impressive improvements with respect to a naive Monte Carlo. Our techniques are readily useful to compute efficiently and with error bounds the exact price in the discrete model. Under certain conditions on the path dependent option (namely, the existence of *fast updates*), there are fast implementations of our *cs* algorithm. We also indicate how it can be combined with recent improved lattice models to obtain faster convergence to the values of the continuous model. Our approach is basic and general, in particular it may be applicable to higher dimensional models like the ones described in [6] and [15].

Appendix A. Appendix

A.1. *Parallelogram Identity*

Here we present a simple equation which generalizes the parallelogram identity for inner product spaces. In the main text, we only apply this identity to the space $\mathbb{R}$ but it is convenient to prove the result on a more general setting. Let H be a vector space of dimension $q+1$ and $\langle\ ,\ \rangle$ be a real inner product defined on H. Letting $||\mathbf{v}||^2 = \langle \mathbf{v}, \mathbf{v}\rangle$ we have,

Proposition Appendix A.1. *Let $p_k \geq 0$ and $\sum_{k=0}^{q} p_k = 1$, then for any set of vectors $\mathbf{v}_k \in H$, $k = 0, \ldots, q$,*

$$\sum_{k=0}^{q} p_k\ ||\mathbf{v}_k||^2 - ||\sum_{k=0}^{q} p_k\ \mathbf{v}_k||^2 = \sum_{k=0}^{q-1} p_k \left(\sum_{j=k+1}^{q} (p_j ||\mathbf{v}_k - \mathbf{v}_j||^2) \right) = \tag{A.1}$$

$$\frac{1}{2} \sum_{k=0}^{q} \sum_{j=0}^{q} (p_k p_j ||\mathbf{v}_k - \mathbf{v}_j||^2). \tag{A.2}$$

Proof.

$$\sum_{k=0}^{q} p_k\ ||\mathbf{v}_k||^2 - \langle \sum_{k=0}^{q} p_k\ \mathbf{v}_k, \sum_{k=0}^{q} p_k\ \mathbf{v}_k \rangle = \tag{A.3}$$

$$\sum_{k=0}^{q} p_k\ ||\mathbf{v}_k||^2 (1 - p_k) - 2 \sum_{k=0}^{q-1} p_k \left(\sum_{j=k+1}^{q} p_j \langle v_k, v_j \rangle \right) = \tag{A.4}$$

$$\sum_{k=0}^{q} \left(p_k\ ||\mathbf{v}_k||^2 (\sum_{j=0, j\neq k}^{q} p_j) \right) - 2 \sum_{k=0}^{q-1} p_k \left(\sum_{j=k+1}^{q} p_j \langle v_k, v_j \rangle \right) = \tag{A.5}$$

$$\sum_{k=0}^{q-1} p_k \left(\sum_{j=k+1}^{q} p_j (||v_k||^2 + ||v_j||^2) \right) - 2 \sum_{k=0}^{q-1} p_k \left(\sum_{j=k+1}^{q} p_j \langle v_k, v_j \rangle \right) = \tag{A.6}$$

$$\sum_{k=0}^{q-1} \left(\sum_{j=k+1}^{q} p_k p_j ||v_k - v_j||^2 \right). \tag{A.7}$$

The last equality in (A.1) is clear. □

A.2. *Standard Errors*

Computation of Standard Error for LMC

$$\texttt{StError}_{lmc}(X, m) = e^{-r(t-t_0)} \sqrt{v\hat{a}r(A_{lmc,\Omega}(X, m))}, \tag{A.8}$$

$$v\hat{a}r(A_{lmc,\Omega}(X, m)) = \sum_{i=0}^{N} P(\Omega^i)^2 \; v\hat{a}r(A_{mc,\Omega^i}(X, m_i)) \tag{A.9}$$

where

$$v\hat{a}r(A_{mc,\Omega^i}(X, m_i)) = \frac{\hat{\sigma}^2_{mc,\Omega^i}(X, m_i)}{m_i}. \tag{A.10}$$

Finally

$$\hat{\sigma}^2_{mc,\Omega^i}(X, m_i) = \frac{1}{(m_i - 1)} \sum_{q=1}^{m} (X(w^{q,i}) - A_{mc,\Omega^i}(X, m_i))^2. \tag{A.11}$$

with $w^{q,i} \in \Omega^i$ sampled independently.

Computation Standard Error for CS

$$\texttt{StError}_{cs}(X, m) = e^{-r(t-t_0)} \sqrt{v\hat{a}r(A_{cs,\Omega}(X, m))}, \tag{A.12}$$

$$v\hat{a}r(A_{cs,\Omega}(X, m)) = \sum_{i=0}^{N} P(\Omega^i)^2 \; v\hat{a}r(A_{mc,\Omega^i}(X_s, m_i)) \tag{A.13}$$

where

$$v\hat{a}r(A_{mc,\Omega^i}(X_s, m_i)) = \frac{\hat{\sigma}^2_{mc,\Omega^i}(X_s, m_i)}{m_i}. \tag{A.14}$$

Finally

$$\hat{\sigma}^2_{mc}(X_s, m_i) = \frac{1}{(m_i - 1)} \sum_{q=1}^{m} (X(w^{q,i}) - A_{mc,\Omega}(X_s, m_i))^2. \tag{A.15}$$

with $w^{q,i} \in \Omega^i$ sampled independently.

Acknowledgments

We would like to thank P. Catuogno for useful suggestions, we also thank J. Cai and A. Korobchevsky for developing software used during the writing of this paper.

References

[1] M. B. Alaya, *On the simulation of expectations of random variables depending on a stopping time*, Stochastic Analysis and Applications, **11 (2)** , 133-153, 1993.
[2] N. Bouleau and D. Lepingle *Numerical Methods for Stochastic Processes*
[3] P. Boyle, M. Broadie and P. Glasserman *Monte Carlo methods for security pricing*, Journal of Econmic Dynamics and Control, Vol. **21**, 1267-1321, 1997.
[4] M. Broadie, P. Glasserman and S.G. Kou, *Connecting discrete and continuous path-dependent options*, Finance Stochat., Vol. **3**, 55-82, 1999.
[5] T. Cheuk and T. Vorst, *Currency lookback options and the observation frequency: A binomial approach*, J. International Money Finance, Vol. **16**, 173-187, 1997.
[6] H. He *Convergence from discrete to continuous-time contingent claims prices*, The Review of Financial Studies, Vol. **3**, No. 4, 523-546, 1990.
[7] S. Heston and G. Zhou *On the rate of convergence of discrete-time contingent claims*, Mathematical Finance, Vol. **10**, No. 1, 53-75, January 2001.
[8] J. Hull and A. White *Efficient procedures for valuing European and American path-dependent options*, The Journal of Derivatives, Fall 1993.
[9] P. Jackel, *Monte Carlo Methods in Finance*. Wiley Finance, 2002.
[10] H.M Kat, *Pricing lookback options using binomial trees: An evaluation*, J. Financial Eng., Vol. **4**, 375-397, 1995.
[11] D. Knuth, *The Art of Computing Programming. Volume 2*. 3rd edition.
[12] R. Korn and E. Korn, *Option Pricing and Portfolio Optimization. Modern Methods of Financial Mathematics*. Graduate Studies in Mathematics, Volume 31. American Mathematical Society, 2001.
[13] D. Leisen and M. Reimer *Binomial models for option valuation- examining and improving convergence*, Applied Mathematical Finance, **3**, 319-346, 1996.
[14] G. E. Martin, *Counting: The Art of Enumerative Combinatorics*. Springer-Verlag, New York, 2001.
[15] D.N. Nelson and K. Ramaswamy *Simple binomial processes as diffussion approximations in financial models*, The Review of Financial Studies,Vol. **3**, No. 3, 393-430, 1990.
[16] P. Ritchken *On pricing barrier options*, The Journal of Derivatives, 19-28, Winter 1995.
[17] L. C. G. Rogers and E. J. Stapleton *Fast accurate binomial pricing*, Finance and Stochastics, Vol. **2**, pp. 3-17, 1998.
[18] J. B. Walsh and O. D. Walsh, *Embedding and the convergence of the binomial and trinomial schemes*. Numerical Methods of Differential Equation, T. J. Lyons and T. S. Salisbury, editors. Fields Institute Communications, 2002.
[19] P. Walters, *An Introduction to Ergodic Theory*. Springer-Verlag, New York, 1982.