HAL
open science

The binomial interpolated lattice method fro step double barrier options

Elisa Appolloni, Gaudenzi Marcellino, Antonino Zanette

To cite this version:

Elisa Appolloni, Gaudenzi Marcellino, Antonino Zanette. The binomial interpolated lattice method fro step double barrier options. International Journal of Theoretical and Applied Finance, 2014, 17 (6), pp.1450035. 10.1142/S0219024914500356 . hal-01096581

HAL Id: hal-01096581

https://hal.science/hal-01096581v1

Submitted on 12 Jan 2015

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Copyright

The Binomial Interpolated Lattice Method for Step Double Barrier Options

Elisa Appolloni [*], Marcellino Gaudenzi[†], Antonino Zanette[‡]

Abstract

We consider the problem of pricing step double barrier options with binomial lattice methods. We introduce an algorithm, based on interpolation techniques, that is robust and efficient, that treats the "near barrier" problem for double barrier options and permits the valuation of step double barrier options with American features. We provide a complete convergence analysis of the proposed lattice algorithm in the European case.

Keywords: step double barrier options, American options, binomial method, interpolation, rate of convergence.

Introduction

Double Barrier options have become quite popular especially in the foreign exchange markets. A double barrier option has a lower barrier and an upper barrier which control the option. Once either of these barriers is breached, the status of the option is immediately determined: either the option comes into existence if the barrier is of knock-and-in type, or ceases to exist if the barrier is of knock-and-out type. Step double barrier options are more flexible contracts that allow investors to set knock-and-out or knock-and-in levels they want. The feature of these contracts is that the double barrier is not constant as in the standard case, but it evolves as a step function of time. Guillame [12] presents closed-form formulae for different types of two-step double barrier options in the Black-Scholes model, but no analytical expressions are given for more general step barrier options. In the latter case the author proposes a conditional Monte Carlo method scheme enhanced with control variate.

In this article we deal with numerical tree methods because they permit to easily treat general multi-step double barrier options, including early exercise features.

The classical CRR approach may be problematic when applied to barrier options because the convergence is very slow compared with the standard case. A possible solution widely shared in literature is to feed the algorithm with the right value of the barrier. In fact the convergence behavior improves when the barrier lies exactly (or is very close) on a layer of the

[*] *Dipartimento MEMOTEF, Sapienza Universitá di Roma, Italy, elisa.appolloni@uniroma1.it*
[†] *Dipartimento di Scienze Economiche e Statistiche, Universitá di Udine, Italy, gaudenzi@uniud.it*
[‡] *Dipartimento di Scienze Economiche e Statistiche, Universitá di Udine, Italy, antonino.zanette@uniud.it*

tree nodes. Boyle-Lau [2] choose the number of time steps in order to minimize the distance between the barrier and a layer of nodes. Figlewky-Gao [6] introduce an Adaptive Mesh Model that refine the tree mesh near the barrier. Ritchken [15] aligns a layer of nodes of the trinomial tree with each barrier. Later Cheuck-Vorst [3] present a modification of the trinomial method (based on a change of the geometry of the tree) which allows to set a layer of nodes exactly on the barrier for every choice of the number of time steps. Gaudenzi-Lepellere [8] introduce suitable interpolations of binomial values and Gaudenzi-Zanette [9] construct a tree where all the mesh points are generated by the barrier itself. However, all the previous methods are not able to price efficiently double barrier options.

In order to deal with double barrier options Dai-Lyuu [5] introduce the bino-trinomial tree that is constructed so that both barriers exactly hit two lines of the tree nodes. Numerical results show that this method is not able to treat the "near barrier" problem, occurring when the initial asset price is very close to one of the barriers.
To overcome this problem we introduce a method (called Binomial Interpolated Lattice) that generates the binomial tree points using the Dai-Lyuu [5] binomial mesh but forgets the trinomial part using just simple interpolations. Moreover, the extension of this method to multi-step double barrier options (including American features) is straightforward and it allows to obtain accurate estimates of the prices in a very short time.

We analyze the convergence of the proposed method and those of the Dai-Lyuu procedure. We show that the rate of convergence of the methods is the same, proving that the approximation error is equal in both cases and that it is $o(\frac{1}{N^{1-\alpha}})$, for all $\alpha \in (0,1)$ (where N is the number of steps of the binomial lattice). The analysis allows us to clarify the conditions in which the Dai-Lyuu method presents some drawbacks in the "near barrier" case.

The paper is organized as follows. In Section 1 we present the model and the bino-trinomial tree method for continuous double barrier options and in Section 2 we describe the new proposed lattice algorithm for double barrier options. In Section 3 we provide rate of convergence results of our proposed binomial method and we remark that it is the same as Dai and Lyuu algorithm. Moreover, we describe an analysis of the troubles of the bino-trinomial method in the "near barrier" case. In Section 4 we provide the extension of this method to step double barrier options. Finally, in Section 5, we compare the results obtained with our algorithm with Dai-Lyuu bino-trinomial tree (double barrier options), Guillaume closed-form formulae (two step double barrier options) and Monte Carlo method (multi-step double barrier options).

1 The model and the bino-trinomial method

In this paper, we consider a market model where the evolution of a risky asset is governed by the Black-Scholes stochastic differential equation

$$\frac{dS_t}{S_t} = rdt + \sigma dB_t, \quad S_0 = s_0, \tag{1}$$

where $(B_t)_{0\leq t\leq T}$ is a standard Brownian motion under the risk neutral measure Q. The non-negative constant r is the force of interest rate and σ is the volatility of the risky asset.

Let M be the number of steps of the binomial tree and $\Delta\tau = \frac{T}{M}$ the corresponding time-step.

The standard discrete binomial process is given by

$$S_{(i+1)\Delta\tau} = S_{i\Delta\tau} Y_{i+1}, \quad 0 \leq i \leq M-1,$$

where the random variables $Y_1, \ldots, Y_M$ are independent and identically distributed with values in $\{d, u\}$. Let us denote by $p = \mathbb{P}(Y_M = u)$. The Cox-Ross-Rubinstein tree corresponds to the choice $u = \frac{1}{d} = e^{\sigma\sqrt{\Delta\tau}}$ and

$$p = \frac{e^{r\Delta\tau} - e^{-\sigma\sqrt{\Delta\tau}}}{e^{\sigma\sqrt{\Delta\tau}} - e^{-\sigma\sqrt{\Delta\tau}}}.$$

Now, let us consider a continuous double barrier option with barrier levels L (lower barrier) and H (higher barrier). In order to treat the double barrier options pricing problem Dai-Lyuu [5] introduce the following bino-trinomial method. After a logarithmic change of the barriers $l = log(\frac{L}{s_0})$ and $h = log(\frac{H}{s_0})$ they first construct in the log-space a binomial CRR random walk with space step $\sigma\sqrt{\Delta T}$, where the new time step ΔT is defined as follows.

Considering the CRR choice of time step $\Delta\tau = \frac{T}{M}$, the new time step is defined such that

$$\Delta T = \left(\frac{h-l}{2k\sigma}\right)^2$$

where $k = \lceil \frac{h-l}{2\sigma\sqrt{\Delta\tau}} \rceil$. By this way, the layers coincide with down barrier L and up barrier H and the new number of steps is $M' = \lfloor \frac{T}{\Delta T} \rfloor$. Now, it is possible to build a binomial structure of M' time steps with binomial coefficient $u = \frac{1}{d} = e^{\sigma\sqrt{\Delta T}}$ and probability

$$p = \frac{e^{r\Delta T} - e^{-\sigma\sqrt{\Delta T}}}{e^{\sigma\sqrt{\Delta T}} - e^{-\sigma\sqrt{\Delta T}}}. \tag{2}$$

The remaining amount of time to make the whole tree span T years, that we denote with $\Delta T'$, is defined as

$$\Delta T' = T - \left(\left\lfloor \frac{T}{\Delta T} \right\rfloor - 1\right)\Delta T$$

and corresponds to the length of the first time step of the bino-trinomial tree. Finally, Dai-Lyuu construct a 1-step trinomial tree, using a moment matching procedure, starting from s_0 and reaching three nodes of the previous binomial CRR tree at time $\Delta T'$. Specifically, at time $\Delta T'$ they select the central node, that we call Y_l, such that it is the closest lattice point to the mean of the logarithmic stock price process $Y_t = \log S_t$. Once this point is chosen they consider two points, one below and one above Y_l: Y_{l-1} and Y_{l+1}, respectively. Then, in order to connect these three points to the starting one, they match the mean and the variance in $\Delta T'$ of the continuous process Y_t with the mean and the variance of the discrete process. We recall that the mean and the variance of Y_t are equal to

$$\mu = (r - \sigma^2/2)\Delta T',$$
$$Var = \sigma^2 \Delta T',$$

respectively. So, the branching probabilities (that we call p_{l-1}, p_l, p_{l+1}) can be derived by solving the following three equations

$$\begin{aligned}
&p_{l-1}Y_{l-1} + p_l Y_l + p_{l+1}Y_{l+1} = \mu, \\
&p_{l-1}(Y_{l-1} - \mu)^2 + p_l(Y_l - \mu)^2 + p_{l+1}(Y_{l+1} - \mu)^2 = Var, \\
&p_{l-1} + p_l + p_{l+1} = 1.
\end{aligned}$$

Then, the merge of the binomial tree of M' steps and the 1-step trinomial tree provide all the mesh structure. The pricing of European or American continuous double barrier options can be done by backward dynamic programming procedure using this bino-trinomial mesh structure. The numerical results in Section 5 show that this binomial-trinomial structure is not able to treat the "near barrier" problem. In order to overcome this, we introduce a simpler binomial structure called the "Binomial Interpolated Lattice" approach.

2 The Binomial Interpolated Lattice approach for double Barrier options

In the following, we will use the same binomial parameters ΔT, u, d and p of Dai-Lyuu [5], computed as described in the previous section. Moreover, we modify the number of steps considering a new number of time steps $N := M' + 2$ in order to perform a suitable interpolation in time.

First of all, we construct a binomial mesh structure where all the binomial nodes are generated by the barriers. Therefore we build a tree which nodes at maturity are indeed all of type

$$Lu^{2j}, \qquad j = 0, ..., k,$$

so that $Lu^{2k} = H$ (where, as in the previous section, $k = \lceil \frac{h-l}{2\sigma\sqrt{\frac{T}{M}}} \rceil$).

The underlying asset at a generic node (i, j), $\forall\, i = 0, ..., N-1$, is

$$S_{i,j} = \begin{cases} Lu^{2j}, & j = 0, ..., k \qquad \text{if } N - i \text{ is even} \\ Lu^{2j+1}, & j = 0, ..., k-1 \quad \text{if } N - i \text{ is odd} \end{cases}$$

We now proceed to the description of the pricing algorithm in the case of double barrier knock-and-out options. We shall denote by $v^N(t_i, S_{i,j})$ the option prices at time t_i depending on the underlying $S_{i,j}$.

The option prices at maturity are

$$v^N(t_N, S_{N,0}) = v^N(t_N, S_{N,k}) = 0 \text{ and } \; v^N(t_N, S_{N,j}) = \psi(S_{N,j}), \; \forall j = 1..., k-1,$$

where $\psi(x)$ is the payoff function. For call options $\psi(x) = \max\{x - K, 0\}$, while for put options $\psi(x) = \max\{K - x, 0\}$, where K is the strike price.

At time steps $i = N-1, ..., 0$ the option prices are backwardly computed by means of the formulas

$$v^N(t_i, S_{i,j}) = e^{-r\Delta T}[pv^N(t_{i+1}, S_{i+1,j+1}) + (1-p)v^N(t_{i+1}, S_{i+1,j})], \; j = 0, ...k-1, \quad \text{if } \; N-i \; \text{ is odd},$$

$v^N(t_i, S_{i,j}) = e^{-r\Delta T}[pv^N(t_{i+1}, S_{i+1,j}) + (1-p)v^N(t_{i+1}, S_{i+1,j-1})]$, $j = 1, ...k-1$, if $N-i$ is even.

The values at the barriers $v^N(t_i, L)$, $v^N(t_i, H)$, are set equal to 0 at every step i with $N - i$ even, in order to take into account the "out" feature of the barrier option.

At time steps $i = 0$ and $i = 2$ we choose four nodes (two less and two greater than s_0). In order to approximate the price of the double barrier option we first interpolate in time the points chosen so that we obtain four "precise" prices of the option at time 0. Then we proceed with a Lagrange four points interpolation in space, i.e. we interpolate the four prices at s_0. So, we provide the estimation of the option price at time 0 and initial underlying asset s_0. The procedure is illustrated in Figure 1.

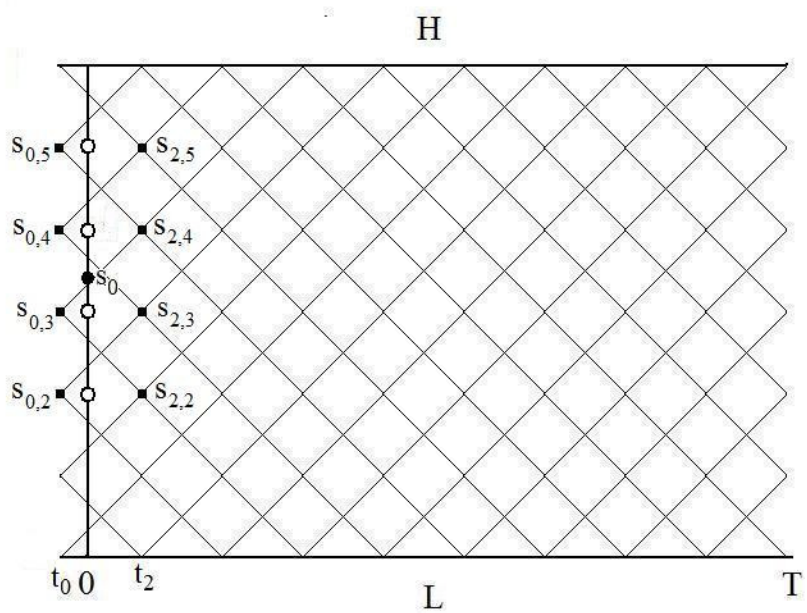

Figure 1: *Binomial Interpolated Lattice Method. Double Knock-Out Barrier Option. The price at s_0 is obtained by a Lagrange four points interpolation in space of the prices at the empty circles, such prices are obtained by a linear interpolation in time of the prices at the nodes denoted by squares.*

We remark that there are some cases in which we need to modify the choice of the interpolation points and this happens when s_0 is close to the barrier. Let us suppose that s_0 is near to the lower barrier L. Now we have two possible cases: there are no points between s_0 and the barrier L and there is only one point between s_0 and the barrier L. In the first case we select at times t_0 and t_2 the two points above s_0 and the point on the barrier. So we perform three interpolations in time using the chosen points and we obtain three different prices at time 0. Then we consider the polynomial passing through these three points and we evaluate it at s_0 (see Figure 2, cases a) and b)). We remark that in case a) the mesh constructed provides at times t_0 and t_2 a node on the barrier L, while in case b) there is not a node on the barrier by construction but we can always consider it in the interpolation procedure because here we

know that the price is equal to 0. In the second case we choose four points at t_0 and t_2: the two above s_0, the point below s_0 and the point on the barrier. So we interpolate linearly four times and then we evaluate at s_0 the polynomial passing through the four points obtained at 0. See Figure 2, cases c) and d). We observe again that in the case in which there is not a node on the barrier by construction we can always consider it in the interpolation procedure.

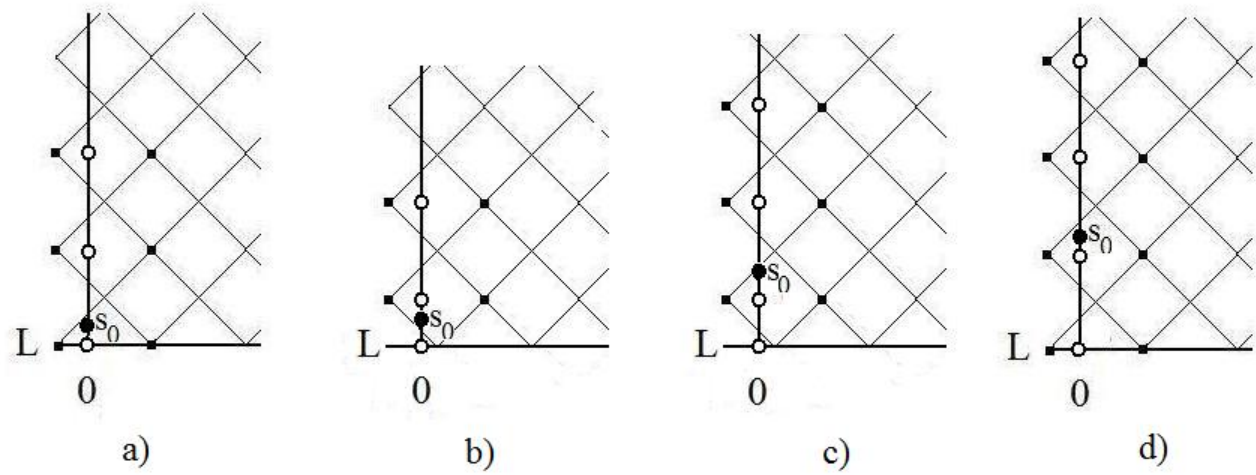

Figure 2: *Binomial Interpolated Lattice Method. Double Barrier Knock-Out Options in the "near barrier case". In cases a) and b) the interpolation in space involves three nodes: at times t_0 and t_2 we select the two nodes above s_0 and the node on the barrier. We observe that case a) occurs when $N-2$ is even and case b) when $N-2$ is odd. In cases c) and d), instead, we select four nodes at times t_0 and t_2. Case c) occurs when $N-2$ is odd and case d) when $N-i$ is even.*

In the American case the procedure is similar with suitable differences for the prices values on the barriers. In particular we set $v^N(t_i, L) = \psi(L)$ and $v^N(t_i, H) = \psi(H)$ for each time step $i = N, ..., 0$ with $N - i$ even (see Remark 5.1 in [8]). In the backward procedure, as usual, we need to compare the early exercise with the continuation value at each node of the tree.

The procedure previously described provides an efficient evaluation of double barrier options both in European and American case. We will show this in the last section, concerning the numerical results, where our method will be compared with the bino-trinomial algorithm.

3 Rate of convergence in the European case

In this section we study the rate of convergence of the Binomial Interpolated Lattice approach. Gobet in [10] gives an asymptotic expansion of the standard binomial tree error and proves that the main contribution term depends on the distance between the effective barrier and the tree overshoot of the barrier itself (see Theorem 3.3 in [10]). In the following we use this result and some properties of the Lagrange polynomials providing the final interpolations to show that the approximation error of the Binomial Interpolated Lattice scheme is $o(\Delta T^{1-\alpha})$,

for every $\alpha \in (0,1)$. We recall that $v^N(t_i, S_{i,j})$ is the approximated price on the lattice at time t_i and asset price $S_{i,j}$. Moreover, we call $v(t_i, S_{i,j})$ the corresponding price in the continuous model at time t_i and asset price on the lattice $S_{i,j}$ and $v(t,s)$ the continuous price at time t and underlying price s with $(t,s) \in [0,T] \times \mathbb{R}_+$. We fix now the starting point s_0. According to what developed in Section 2, at time step $i = 2$, i.e. at time $T - (N-2)\Delta T$, we choose four nodes (two less and two greater than s_0) and similarly we do at time step $i = 0$, i.e. at time $T - N\Delta T$. We observe that the mesh constructed provides the same four nodes at times t_0 and t_2 and we set them as follows

$$S_{j-2} < S_{j-1} \leq s_0 < S_j < S_{j+1},$$

where $S_k := S_{0,k} = S_{2,k}$, for all $k \in \{j-2, j-1, j, j+1\}$. At the chosen points the algorithm gives the prices

$$v^N(t_i, S_{j-2}), v^N(t_i, S_{j-1}), v^N(t_i, S_j), v^N(t_i, S_{j+1}), \quad i = 0, 2,$$

respectively. Now, as $k = j-2, j-1, j, j+1$, we write down the expressions of the approximated option prices $v^N(0, S_k)$ obtained by linearly interpolating in time the points

$$(t_0, v^N(t_0, S_k)), (t_2, v^N(t_2, S_k)), \quad k = j-2, j-1, j, j+1.$$

This means that we set

$$v^N(0, S_k) = \overline{q}_k(0), \quad k = j-2, j-1, j, j+1,$$

where $\overline{q}_k(t)$ are the linear interpolating polynomials given by

$$\overline{q}_k(t) = a_0(t) v^N(t_0, S_k) + a_2(t) v^N(t_2, S_k), \quad \text{with } a_0(t) = \frac{t - t_2}{t_0 - t_2}, \quad a_2(t) = \frac{t - t_0}{t_2 - t_0}.$$

Then, in order to define the precise price at time 0 we interpolate in space through a Lagrange polynomial the points

$$(0, v^N(0, S_{j-2})), (0, v^N(0, S_{j-1})), (0, v^N(0, S_j)), (0, v^N(0, S_{j+1})).$$

This means that we set

$$v^N(0, s_0) = q(s_0),$$

where $q(x)$ is the Lagrange polynomial given by

$$q(x) = b_{j-2}(x) v^N(0, S_{j-2}) + b_{j-1}(x) v^N(0, S_{j-1}) + b_j(x) v^N(0, S_j) + b_{j+1}(x) v^N(0, S_{j+1}),$$

with (for details see [16])

$$\begin{aligned}
b_{j-2}(x) &= \frac{(x - S_{j-1})(x - S_j)(x - S_{j+1})}{(S_{j-2} - S_{j-1})(S_{j-2} - S_j)(S_{j-2} - S_{j+1})}, \\
b_{j-1}(x) &= \frac{(x - S_{j-2})(x - S_j)(x - S_{j+1})}{(S_{j-1} - S_{j-2})(S_{j-1} - S_j)(S_{j-1} - S_{j+1})}, \\
b_j(x) &= \frac{(x - S_{j-2})(x - S_{j-1})(x - S_{j+1})}{(S_j - S_{j-2})(S_j - S_{j-1})(S_j - S_{j+1})}, \\
b_{j+1}(x) &= \frac{(x - S_{j-2})(x - S_{j-1})(x - S_j)}{(S_{j+1} - S_{j-2})(S_{j+1} - S_{j-1})(S_{j+1} - S_j)}.
\end{aligned}$$

So, by resuming, we set the approximated price at time 0 as

$$v^N(0,s_0) = \sum_{k=j-2}^{j+1} \sum_{i\in\{0,2\}} a_i(0) b_k(s_0) v^N(t_i, S_k).$$

Proposition 1 *The Binomial Interpolated Lattice error*

$$Err_{BIL}(N) = |v^N(0,s_0) - v(0,s_0)|$$

resulting from the algorithm behaves as follows:

$$Err_{BIL}(N) = o(\Delta T^{1-\alpha}),$$

for every $\alpha \in (0,1)$.

Proof. Since

$$\sum_{k\in\{j-2,j-1,j,j+1\}} b_k(s_0) = 1, \quad \sum_{i\in\{0,2\}} a_i(0) = 1,$$

we can write

$$\begin{aligned} Err_{BIL}(N) &= \left| \sum_k \sum_i a_i(0) b_k(s_0) (v^N(t_i, S_k) - v(0,s_0)) \right| \\ &\leq \left| \sum_k \sum_i a_i(0) b_k(s_0) (v^N(t_i, S_k) - v(t_i,S_k)) \right| \\ &+ \left| \sum_k \sum_i a_i(0) b_k(s_0) (v(t_i, S_k) - v(0,s_0)) \right|. \end{aligned}$$

Let us consider first the generic term in the second sums above: by applying Taylor's formula around the point $(0,s_0)$ we can write

$$v(t_i,S_k) - v(0,s_0) = \partial_t v(0,s_0) t_i + \partial_x v(0,s_0)(S_k - s_0) + \frac{1}{2} R(i,k),$$

where

$$\begin{aligned} R(i,k) &= \partial^2_{x,x} v(0, s_0 + \theta_{i,k}(S_k - s_0))(S_k - s_0)^2 + \partial_{t,t} v(\overline{\theta}_{i,k} t_i, s_0) t_i^2 \\ &+ 2\partial^2_{t,x} v(\overline{\theta}_{i,k} t_i, s_0 + \theta_{i,k}(S_k - s_0))(S_k - s_0) t_i, \end{aligned}$$

and $\theta_{i,k}, \overline{\theta}_{i,k}$ are suitable points in $[0,1]$. By using the global estimates in Gobet [10] (Lemma 3.1), we conclude that all the partial derivatives of $v(t,s)$ are bounded around $(0,s_0)$. So, we can immediately conclude that

$$\left| \sum_k \sum_i a_i(0) b_k(s_0) R(i,k) \right| \leq O(\Delta T).$$

Moreover we have

$$\sum_k \sum_i a_i(0) b_k(s_0)(\partial_t v(0,s_0) t_i + \partial_x v(0,s_0)(S_k - s_0))$$
$$= \partial_t v(0,s_0) \sum_i a_i(0) t_i + \partial_x v(0,s_0) \sum_k b_k(s_0)(S_k - s_0) = 0$$

because, by construction, $q_1(t) = \sum_i a_i(t) t_i$ is the linear polynomial which interpolates the points (t_i, t_i), $i = 0, 2$, so that $q_1(t) = t$ and then $q_1(0) = 0$. Similarly, the polynomial $q_2(x) = \sum_k b_k(x)(S_k - s_0)$ is the Lagrange polynomial which interpolates the points $(S_k, S_k - s_0)$, $k \in \{j-2, j-1, j, j+1\}$, so that $q_2(x) = x - s_0$ and again $q_2(s_0) = 0$. Therefore, we get

$$Err_{BIL}(N) \leq \left| \sum_k \sum_i a_i(0) b_k(s_0)(v^N(t_i, S_k) - v(t_i, S_k)) \right| + O(\Delta T).$$

In order to deal with the generic term in the above sums we define $Y_k := \log S_k$, for all $k \in \{j-2, j-1, j, j+1\}$, $u^N(t_i, Y_k) := v^N(t_i, e^{Y_k})$ and $u(t_i, Y_k) := v(t_i, e^{Y_k})$. We apply now Theorem 3.3 in Gobet [10]. We stress that the probability p of an up jump in (2) differs from the probability defined in Gobet for an $O(\sqrt{\Delta T})$, but it is easy to see that the result in [10] still remains valid in our case. Moreover, we remark that asymptotic expansion of the standard binomial tree error, that we call $Err(N)$, given in [10] is

$$Err(N) = C_1(H_N - H) + C_2(L - L_N) + o(\sqrt{\Delta T}),$$

where H_N is the first node of the tree over H, L_N is the first node of the tree lower than L and C_1 and C_2 are two positive constants. Actually, straightforward computations give that the error above can be written as follows

$$Err(N) = C_1(H_N - H) + C_2(L - L_N) + \mathcal{R}_{\Delta T},$$

where $\mathcal{R}_{\Delta T} \leq O(\Delta T \log(1/\Delta T))$ (for details see pag.11 of Gobet [10]). As a consequence, one has for every $\alpha \in (0,1)$ that

$$Err(N) = C_1(H_N - H) + C_2(L - L_N) + o(\Delta T^{1-\alpha}).$$

Now, the Binomial Interpolated Lattice is constructed so that two layers of nodes coincide with the down barrier L and the upper barrier H, therefore the main contribution term in the error expansion, that is of order $O(\sqrt{\Delta T})$, vanishes. So, we get

$$v^N(t_i, S_k) - v(t_i, S_k) = u^N(t_i, Y_k) - u(t_i, Y_k) = o(\Delta T^{1-\alpha}) \quad \text{for every } i, k.$$

The statement now follows. □

Remark 1 *We observe that Gobet convergence result is the only one that deals with the double barrier case and, moreover, it is applicable to a generic payoff function (for details see [10]). Lin and Palmer [14] have recently given explicit formulas for the coefficients of the asymptotic expansion of the CRR binomial price, but they treat single barrier European call options. In*

that case they obtain that the rate of convergence is of order ΔT if the barrier lies exactly on a node of the tree. For double barrier options it may be possible to derive similar formulas for the coefficients. In fact, we remark that in Gobet error expansion, since α can be taken arbitrarily close to zero, one essentially can assert that

$$Err(N) = C_1(H_N - H) + C_2(L - L_N) + O(\Delta T).$$

Remark 2 *We observe that the use of the interpolation technique is strategic in order to get that the rate of convergence of the Binomial Interpolated Lattice is $o(\Delta T^{1-\alpha})$, for $\alpha \in (0,1)$. In fact, we do not know a priori if the initial point s_0 is a point of the lattice and in general it is not: our procedure allows us to numerically compute the option price for every observed starting condition $s_0 \in (L, H)$. So if we want to directly approximate $v(0, s_0)$ with $v^N(t_1, \tilde{s})$, where $(t_1, \tilde{s})$ is a point of the lattice "close" to $(0, s_0)$, then one could have $|s_0 - \tilde{s}| = O(\sqrt{\Delta T})$. Therefore, in this case one could have*

$$v(0, s_0) - v^N(t_1, \tilde{s}) \simeq O(t_1) + O(|s_0 - \tilde{s}|) = O(\sqrt{\Delta T}).$$

Thus, it is only thanks to the interpolation rule that the error contribute of order $O(\sqrt{\Delta T})$ always vanishes.

Remark 3 *Let us consider the rate of convergence of the bino-trinomial tree. As described in Section 1, Dai and Lyuu build the grid in the* log*-space until time t_2 and then they construct a 1-step trinomial tree in the remaining amount of time $\Delta T'$ using a moment matching procedure. Specifically they select at time t_2 three nodes that are closer to the mean of the logarithmic process at time $\Delta T'$ and they define the three branching probabilities such that the first two moments of the logarithmic stock price process are matched (see Dai-Lyuu [5] for details). As in the proof of Proposition 1 we call $Y_{i,k} = \log S_{i,k}$ and $u^N(t_i, Y_k) = v^N(t_i, e^{S_k})$ for all i, k. Moreover we define $y_0 := \log s_0$ and we call $u(t, y) = v(t, e^y)$ for all $(t, y) \in [0, T] \times \mathbb{R}$. Now if we call the chosen points at time t_2*

$$Y_{2,l-1}, \quad Y_{2,l} \quad \text{and} \quad Y_{2,l+1},$$

then the Dai and Lyuu algorithm gives the prices

$$u^N(t_2, Y_{2,l-1}), u^N(t_2, Y_{2,l}), u^N(t_2, Y_{2,l+1})$$

and the option price of the bino-trinomial tree at time 0 is obtained by one more application of the backward induction that uses the trinomial approach, i.e.

$$u^N_{DL}(0, y_0) = e^{-r\Delta T'} \sum_{k=-1}^{1} p_k u^N(t_2, Y_{2,l+k}).$$

We can now proceed in the analysis of the convergence rate using arguments similar to the ones

in Proposition 1. In fact, we can write

$$\begin{aligned}
Err_{DL}(N) &= |u^N_{DL}(0,y_0) - u(0,y_0)| \\
&\leq O(\Delta T') + \left| \sum_{k=-1}^{1} p_k(u^N(t_2, Y_{2,l+k}) - u(t_2, Y_{2,l+k})) \right| \\
&+ \left| \sum_{k=-1}^{1} p_k(u(t_2, Y_{2,l+k}) - u(0,y_0)) \right| \\
&\leq O(\Delta T \log(1/\Delta T)) + \left| \sum_{k=-1}^{1} p_k(u(t_2, Y_{2,l+k}) - u(0,y_0)) \right|,
\end{aligned}$$

where the estimate on the right hand side follows by applying Gobet's result. Again by using Taylor's expansion, one gets for every $\alpha \in (0,1)$

$$Err_{DL}(N) = o(\Delta T^{1-\alpha}) + \left| \partial_x u(0,y_0) \sum_{k=-1}^{1} p_k(Y_{2,l+k} - y_0) \right|.$$

Now, the sum in the above r.h.s. is of order $O(\Delta T')$*. In fact we recall that the mean of the logarithmic process* $Y_t = \log(S_t)$ *at time* $\Delta T'$ *is* $y_0 + (r - \sigma^2/2)\Delta T'$ *and that the probabilities* p_k *are calculated such that it coincides with the mean of the discrete approximating process, so we can state that* $\sum_{k=-1}^{1} p_k(Y_{2,l+k} - y_0) = O(\Delta T')$*. Therefore, we obtain*

$$Err_{DL}(N) = o(\Delta T^{1-\alpha}),$$

and we conclude that the rate of convergence of the bino-trinomial tree is the same as our Binomial Interpolated Lattice.

Remark 4 *We observe that there are some cases in which the procedure of the bino-trinomial tree can bring to numerical problems. In fact, if we fix the number* N *of time steps, it may happen that for some values of the starting point* s_0 *one or two of the three nodes required at time* $\Delta T'$ *to build the 1-step trinomial tree fall out of the grid. We briefly recall that in the bino-trinomial tree procedure the central node (we call it* $Y_{2,l}$ *from Remark 3) is selected such that it is the closest to the mean of the process (i.e.* $\log s_0 + (r - \sigma^2/2)\Delta T'$*), while the nodes* $Y_{2,l+1}$ *and* $Y_{2,l-1}$ *are the two nodes adjacent to* $Y_{2,l}$ *(above and below* $Y_{2,l}$ *respectively). In the following we suppose that* $r - \sigma^2/2 \geq 0$*. Let us consider first the case in which the initial point* $\log s_0$ *is near the higher barrier* $\log H$*. We need to consider two different cases:*
i) the mean of the process is above the barrier $\log H$*:*

$$\log s_0 + (r - \sigma^2/2)\Delta T' \geq \log H;$$

ii) the mean of the process is below the barrier $\log H$*:*

$$\log s_0 + (r - \sigma^2/2)\Delta T' < \log H.$$

The case i) is verified when

$$He^{-(r-\sigma^2/2)\Delta T'} \leq s_0 < H, \tag{3}$$

so for this range of values for s_0 the node $Y_{2,l}$ lies on the barrier (i.e. $Y_{2,l} = \log H$), so that node $Y_{2,l+1}$ falls out of the grid. Also in case ii) it may happen the same phenomenon. It is easy to see that if

$$He^{-(r-\sigma^2/2)\Delta T' - \sigma\sqrt{\Delta T}} \leq s_0 < H, \tag{4}$$

then again $Y_{2,l} = \log H$. From (3) and (4) we deduce that for the values of s_0 such that

$$He^{-(r-\frac{\sigma^2}{2})\Delta T' - \sigma\sqrt{\Delta T}} \leq s_0 < H, \tag{5}$$

the bino-trinomial tree may degenerate in the above sense. As for the lower barrier, a similar discussion gives that if

$$L < s_0 \leq Le^{-(r-\sigma^2/2)\Delta T' + \sigma\sqrt{\Delta T}}, \tag{6}$$

then $Y_{2,l} = \log H$. We observe that we can write the above inequality because the exponent on the right side is greater than 0 for a sufficiently large value of N. And whenever $r - \sigma^2/2 < 0$, one can proceed similarly and obtain the same intervals. It is clear that for N large enough any $s_0 \in (L, H)$ does not satisfy both (5) and (6). Nevertheless, for fixed values of N (5) and/or (6) may hold, so that in practice the bino-trinomial approach converges slowly than our procedure. So we conclude that asymptotically the two methods behave the same, but when s_0 is a "near barrier" point the Binomial Interpolated Lattice has the advantage of converging faster than the bino-trinomial method.

Remark 5 *We remark that the proof of the rate of convergence in the case in which we take into account only three points in the space interpolation ("near to the barrier" case) can be treated similarly to the previous one ("far from the barrier" case), so we omit it. Moreover, we observe that in this specific case we always find the three points used in the interpolations.*

4 The Binomial Interpolated Lattice approach for step double Barrier options

In this section we apply the Binomial Interpolated Lattice algorithm introduced in Section 2 for pricing step double barrier options. Let us introduce the *regular step double barrier options* as explained in Guillaume [12]. Let $\{T_0, T_1, ..., T_{n-1}, T_n\}$ be a partition of the option lifetime $[0, T]$ with $0 = T_0 < T_1 < ... < T_n = T$. A *standard n-step double barrier option* is an option in which the barriers are constant in every interval $[T_i, T_{i+1}], i = 0, ..., n-1$. Hence, at each interval $[T_i, T_{i+1}]$ is associated a constant double barrier with down barrier L_i and up barrier H_i. A standard n-step double knock-out option with payoff function ψ, has this payoff at maturity provided that the underlying asset price stayed in (L_i, H_i) in every interval $[T_i, T_{i+1}]$, otherwise it expires worthless or provides a contractual rebate. It is possible to include in the step double barrier options the possibility to remove the knock-out barrier provision (partial-time step double barrier options). For example an *early ending n-step double knock-out option* with maturity T_n has the same payoff of a standard call or put on the condition that the underlying asset price stayed in (L_i, H_i) in every interval $[T_i, T_{i+1}], i = 0, .., n-2$ (hence there are no "out" condition on the last time interval). A *windows n-step double knock-out option* has the same

payoff of a standard call or put on condition that the underlying asset price stayed in (L_i, H_i) in every interval $[T_i, T_{i+1}], i = 1, .., n-2$ (hence there are no "out" condition on the first and last time interval).

A partial-time step double barrier option will always be more valuable that the corresponding standard step double barrier option. Moreover, it is possible to take into account knock-in features in all these contracts. In the European case the knock-in options prices are obtained by taking the difference between the prices of the corresponding vanilla option and the knock-out option.

We can apply the Binomial Interpolated Lattice approach to treat standard and partial-time step double barrier options in a straightforward way. Let us consider for example a *two-step double knock-out option*. We globally take M times steps and we consider $M_1' = \lfloor \frac{T_1 - T_0}{T} \rfloor$ time steps in the first interval $[T_0, T_1]$ and $M_2' = M - M_1'$ time steps in the second interval $[T_1, T_2]$. We first consider the time period $[T_1, T_2]$ and we apply the double barrier procedure used in the previous section. So we compute the binomial parameters $N_2 = M_2' + 2$, ΔT_2, u_2, d_2, p_2, k_2 in order to hit exactly the barriers L_2 and H_2. This leads to a new binomial mesh $\{S^2_{i,j}\}$ defined $\forall\, i = 0, ..., N_2$ as follows

$$S^2_{i,j} = \begin{cases} L_2 u_2^{2j}, \quad j = 0, ..., k_2 & \text{if } N_2 - i \text{ is even} \\ L_2 u_2^{2j+1}, \quad j = 0, ..., k_2 - 1 & \text{if } N_2 - i \text{ is odd} \end{cases}$$

We can then proceed using the backward procedure for $i = N_2, ..., 0$ as described in the previous section. With the linear interpolation in time at T_1 we can obtain at every node $S^2_{0,j}$ the corresponding option price $v^{N_2}(T_1, S^2_{0,j})$.

Now we proceed in the same way in time interval $[T_0, T_1]$. We compute the new binomial parameters N_1, ΔT_1, u_1, d_1, p_1, k_1 in order to hit exactly the barriers L_1 and H_1. This leads to a new binomial mesh structure $\{S^1_{i,j}\}$. In order to obtain the option prices on the new nodes with underlying $S^1_{N_1,j}$, $j = 0, ..., k_1$, we interpolate at every $S^1_{N_1,j}$, $j = 0, ..., k_1$ by a Lagrange interpolation using 4 suitable points in the set $\{(S^2_{0,j}, v^{N_2}(T_1, S^2_{0,j}))\}$, with $j = 0, ..., k_2$ if N_2 is even and with $j = 0, ..., k_2 - 1$ if N_2 is odd. In order to perform such interpolation we set $v^{N_2}(T_1, S^2_{0,j}) = 0$, for j such that $S^2_{0,j} \leq L_2$ or $S^2_{0,j} \geq H_2$. Moreover, the values $v^{N_1}(T_1, S^1_{N_1,j})$ will be set equal to zero if either $S^1_{N_1,j} \leq L_1$ or $S^1_{N_1,j} \geq H_1$.

Finally, we proceed backward for $i = N_1, ..., 0$ and we compute the price at s_0 by a Lagrange interpolation in space and a linear interpolation in time as described before. We represent the mesh described above in Figure 3.

In the *early ending two-step double knock-out option* we just need to add the treatment of the period $[T_2, T_3]$ where there are no "out" conditions. We start by considering the number of time steps M_3 and the corresponding $\Delta\tau_3$. Then we compute k_3 and ΔT_3 in order to hit exactly the barriers L_2, H_2, i.e. $k_3 = \lceil \frac{h_2 - l_2}{2\sigma\sqrt{\Delta\tau_3}} \rceil$ and $\Delta T_3 = \left(\frac{h_2 - l_2}{2k_3\sigma}\right)^2$. The parameters M_3', N_3, u_3, d_3, p_3 are computed as usual. Now, starting from the nodes evaluated at time T_2 we can consider a tree structure $\{S^3_{i,j}\}$ in the time interval $[T_2, T_3]$ of N_3 time steps. At maturity T_3 we obtain the underlying assets $S_{N_3,j} = L_2 u_3^j$, $j = -N_3, ..., 2k_3 + N_3$. Then we apply the backward CRR binomial procedure starting with the maturity condition at time T_3. The prices at the nodes $S_{N_2,j} = L_2 u_2^j$, $j = 0, ..., k_2$ at time T_2 are obtained with the interpolation in time and space.

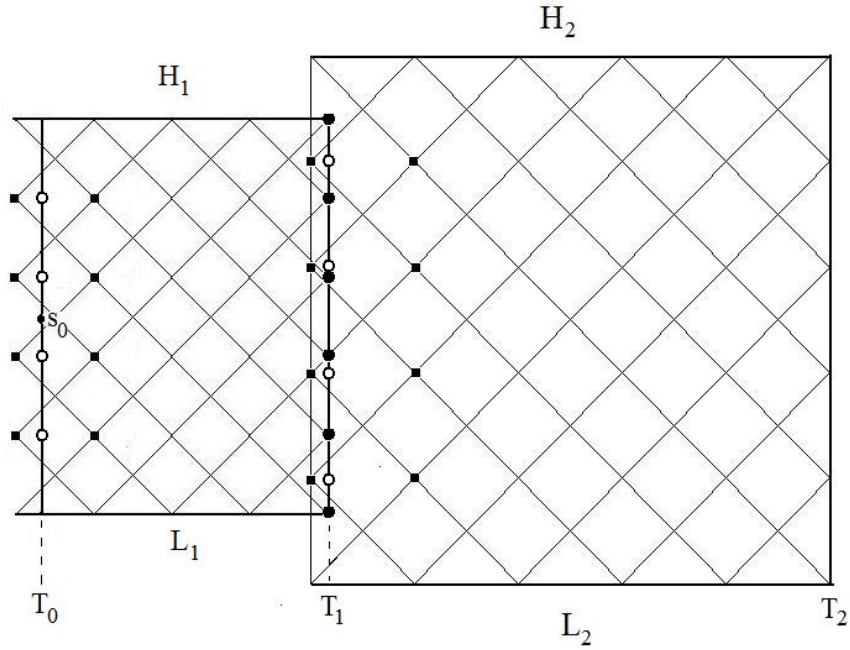

Figure 3: *Binomial Interpolated Lattice Method. Two Step Double Knock-Out Options. The prices at time T_1 are obtained by a Lagrange space interpolation of the prices at the nodes denoted by empty circles, such prices are obtained by a linear interpolation in time of the prices at the adjacent nodes denoted by squares. Similarly we obtain the prices at time T_0.*

Now the procedure is the same as in the standard two-step double barrier options (see Figure 4).

The treatment of the *windows two-steps double knock-out options* is similar. In the *n-step double barrier options* case we just apply the procedure for two-step double barrier options recursively.

5 Numerical results

We provide some numerical results of the algorithms presented in the Sections 2 and 4 in the case of double barrier options, two step double barrier options and multi-step double barrier options. All the computations presented in the tables have been performed in double precision on a PC with a processor Intel Core i5 at 1.7 Ghz.

5.1 Double Barrier Options

In order to test the efficiency of the Binomial Interpolated Lattice (BIL) approach we first consider the numerical experiments proposed in Day-Lyuu (DL) [5] for pricing knock-out double-barrier call options and then we propose other comparisons with different parameters. The volatility of the stock price is $\sigma = 0.25$, the interest rate is $r = 0.1$, the time to maturity is

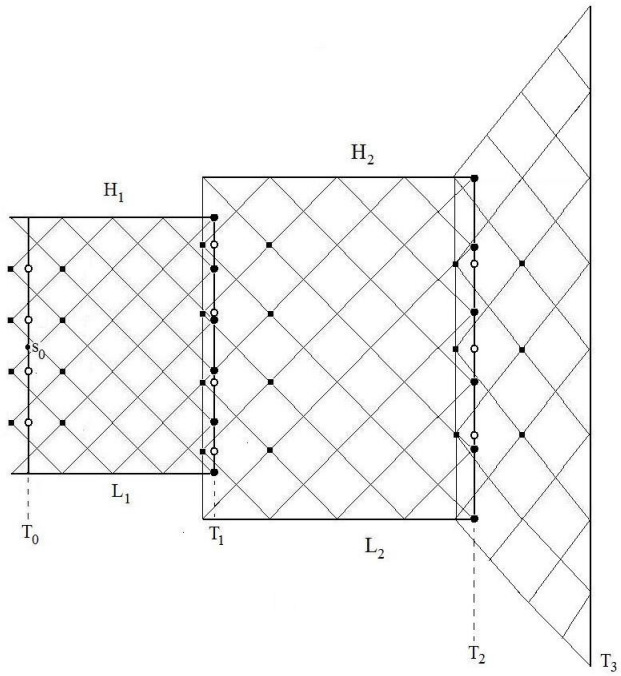

Figure 4: *Binomial Interpolated Lattice Method. Early Ending Knock-Out Options. The prices at time T_2 are obtained by a Lagrange space interpolation of the prices at the nodes denoted by empty circles, such prices are obtained by a linear interpolation in time of the prices at the adjacent nodes denoted by squares. Similarly we obtain the prices at time T_0 and T_1.*

$T = 1$, the strike price is $K = 100$ and the two barriers are $L = 90$ and $H = 140$. We consider three possible values for the initial stock price: $s_0 = 95, 90.05, 139.95$. We observe that in the tables below the number of time steps is M and it refers to the original CRR model, as defined in Section 1. We use as benchmark value the price computed with the closed formula provided by Kunitomo and Ikeda [13]. We observe that in the cases in which s_0 is a point near to the barrier, the Binomial Interpolated Lattice performs better than the bino-trinomial procedure. The results are given in Table 1.

	$s_0 = 95$			$s_0 = 90.05$			$s_0 = 139.95$		
M	DL	KI	BIL	DL	KI	BIL	DL	KI	BIL
100	1.423589		1.422126	0.313771		0.016004	0.196833		0.006562
200	1.440976		1.440586	0.279768		0.016235	0.176563		0.006688
400	1.449705	1.458435	1.450008	0.226927	0.016268	0.016267	0.144278	0.006656	0.006673
800	1.453358		1.453441	0.149755		0.016263	0.093742		0.006661
1600	1.456403		1.456350	0.084767		0.016254	0.051903		0.006650
3200	1.457183		1.457179	0.059701		0.016259	0.035926		0.006653

Table 1: *Knock-Out Double Barrier Call Options Prices with $T = 1$, $r = 0.1$, $\sigma = 0.25$, $K = 100$, $L = 90$, $H = 140$ and s_0 varying.*

In Table 2 we give the price of a knock-out double barrier call option in the "near to the barrier" case. Here $\sigma = 0.25$, $r = 0.1$, $T = 1$ and $K = 100$. Now we vary s_0, L and H so that we can consider both the case in which s_0 is close to L and the case in which s_0 is close to H.

In the table on the left we choose $s_0 = 95$, $L = 94.9$ and $H = 140$, so we have that the starting point s_0 is near to the lower barrier L. Instead, in the table on the right s_0 is near to the higher barrier H and we choose $s_0 = 139.9$, $L = 95$ ad $H = 140$. The results are given in Table 2.

	$s_0 = 95, L = 94.9, H = 140$			$s_0 = 139.9, L = 95, H = 140$		
M	DL	KI	BIL	DL	KI	BIL
100	0.274716		0.024774	0.211312		0.011449
200	0.227618		0.025281	0.085969		0.011148
400	0.109875	0.025305	0.025182	0.083269	0.011238	0.011188
800	0.128411		0.025305	0.087552		0.011253
1600	0.062742		0.025274	0.060058		0.011243
3200	0.053451		0.025286	0.048296		0.011239

Table 2: *Knock-Out Double Barrier Call Options Prices with* $T = 1$, $r = 0.1$, $\sigma = 0.25$, $K = 100$. *The values of* s_0, L *and* H *vary.*

We remark that in Table 1 and in Table 2 for each value of M the starting point s_0 belongs to the critical intervals (5) and (6) defined in Section 3. So, thanks to the sufficient condition given in Remark 5, we explain why the Binomial Interpolated Lattice method performs better than the bino-trinomial tree. In fact if s_0 belongs to the interval $(L, Le^{-(r-\sigma^2/2)\Delta T'+\sigma\sqrt{\Delta T}}]$ or to the interval $[He^{-(r-\frac{\sigma^2}{2})\Delta T'-\sigma\sqrt{\Delta T}}, H)$ the Dai and Lyuu procedure may degenerate. As pointed out in Remark 5, it is clear that if we choose M such that

$$s_0 > Le^{-(r-\sigma^2/2)\Delta T'+\sigma\sqrt{\Delta T}}, \tag{7}$$

(in the case in which s_0 is near to the lower barrier L), or if we choose M such that

$$s_0 < He^{-(r-\sigma^2/2)\Delta T'-\sigma\sqrt{\Delta T}}, \tag{8}$$

(when s_0 is near to the higher barrier H), then the bino-trinomial tree may not degenerate. But the values of M we need to consider in order to satisfy these conditions are very large. In fact in the case of Table 1 with $s_0 = 90.05$, we have to choose $M \geq 201840$. Instead, in the case in which $s_0 = 139.95$ we need to choose $M \geq 489104$ in order to satisfy condition (8). Similarly, in Table 2, we need to choose $M \geq 55987$ if $s_0 = 95$ and $M \geq 122107$ if $s_0 = 139.9$ to satisfy conditions (7) and (8) respectively.

In Table 3 we show two more examples of pricing a knock-out double barrier call option with $\sigma = 0.25$, $r = 0.1$, $T = 1$, $K = 100$, $L = 90$ and $H = 140$ in the "near to the barrier" case. Now we vary the starting point and we choose $s_0 = 92$ and $s_0 = 138$. The numerical results show that the BIL method converges faster than the bino-trinomial tree also in the case in which s_0 is chosen not so much close to the barriers. We also observe that when $s_0 = 92$ we need $M \geq 104$ to satisfy (7) and when $s_0 = 138$ we have to choose $M \geq 289$ to verify condition (8). Moreover, also if M is such that (7) or (8) is satisfied, the Binomial Interpolated Lattice converges faster than the bino-trinomial tree.

	$s_0 = 92, L = 90, H = 140$			$s_0 = 138, L = 90, H = 140$		
M	DL	KI	BIL	DL	KI	BIL
100	0.753689		0.611674	0.365983		0.265373
200	0.675166		0.620375	0.338188		0.270340
400	0.667178	0.626377	0.623008	0.316626	0.271825	0.270702
800	0.624209		0.624220	0.281821		0.270875
1600	0.625457		0.625476	0.271367		0.271286
3200	0.625861		0.625833	0.271534		0.271548

Table 3: *Knock-Out Double Barrier Call Options Prices with* $T = 1$, $r = 0.1$, $\sigma = 0.25$, $K = 100$. *The values of* s_0, L *and* H *vary.*

5.2 Two Step Double Barrier Options

Let us now consider the numerical experiments proposed in Guillame [12] in pricing two step double barrier knock-out put options. In the European case we compare our method with the benchmark value given by the closed formula (GUI) provided in Guillaume [12]. No benchmark is available in the American case. The volatility of the stock price is $\sigma = 0.3$, the interest rate is $r = 0.03$, the current stock price is $s_0 = 100$ and the strike price varies: $K = 90, 100, 110$. In Table 2 we report the values of two-step double knock-out put options with double barrier with parameters: $T_1 = 0.25$, $T_2 = T = 0.5$, $L_1 = 70$, $H_1 = 130$, $L_2 = 75$ and $H_2 = 125$.

	$K = 90$			$K = 100$			$K = 110$		
M	BIL-EU	GUI	BIL-AM	BIL-EU	GUI	BIL-AM	BIL-EU	GUI	BIL-AM
100	0.806457		3.549508	3.152257		7.694393	7.089792		13.516271
200	0.819128		3.541653	3.186487		7.694650	7.196594		13.574689
400	0.815156	0.821806	3.545095	3.187184	3.194080	7.703400	7.186754	7.186905	13.576866
800	0.820657		3.555169	3.191445		7.712047	7.182939		13.580487
1600	0.821165		3.555965	3.192845		7.713536	7.184522		13.581967
3200	0.821348		3.556271	3.192542		7.713324	7.184678		13.582068

Table 4: *Two-step Double Knock-Out Put Options Prices with* $s_0 = 100$, $r = 0.03$, $\sigma = 0.3$, $T_1 = 0.25$, $T_2 = T = 0.5$, $L_1 = 70$, $H_1 = 130$, $L_2 = 75$, $H_2 = 125$ *and* K *varying.*

In Table 3 we consider an early-ending two step double knock-out call with $K = 120$, $s_0 = 100$, $r = 0.03$ and double barrier parameters $T_1 = 0.125$, $T_2 = 0.25$, $T_3 = T = 0.5$, $L_1 = 75$, $H_1 = 125$, $L_2 = 70$, $H_2 = 130$. The volatility varies: $\sigma = 0.15, 0.3$.

	$\sigma = 0.15$			$\sigma = 0.3$		
M	BIL-EU	GUI	BIL-AM	BIL-EU	GUI	BIL-AM
100	0.263380		9.962265	1.575926		9.728607
200	0.265933		9.962439	1.613757		9.741826
400	0.271830	0.2755	9.962422	1.605417	1.6165	9.745550
800	0.273739		9.962533	1.608395		9.750609
1600	0.275081		9.962435	1.612841		9.755035
3200	0.275201		9.962400	1.615489		9.758268

Table 5: *Early-ending Two Step Double Knock-Out Call Options Prices with* $s_0 = 100$, $r = 0.03$, $K = 120$, $T_1 = 0.125$, $T_2 = 0.25$, $T_3 = T = 0.5$, $L_1 = 75$, $H_1 = 125$, $L_2 = 70$, $H_2 = 130$ *and* σ *varying.*

The numerical results show that the method is accurate also in the two-step double knock-out option case.

5.3 Multi Step Double Barrier Options

Finally, in Table 4, we propose the results obtained with our method for a 16-steps knock out double barrier put option. The volatility of the stock price is $\sigma = 0.3$, the interest rate is $r = 0.03$, the current stock price is $s_0 = 100$, the strike price is $K = 110$, the time to maturity is $T = 2$ and the barrier parameters are: $T_i = i \cdot 0.125$, $L_i = 70 - i$, $H_i = 130 + i$, for all $i = 1, ..., 16$. In the European case we use as benchmark value the Monte Carlo method provided in Baldi-Caramellino-Iovino [1] with 10 millions simulations and 1000 Euler time discretization steps (with the confidence interval in parenthesis). We observe that the computation times are very fast. For example, for $M = 12800$ and $M = 25600$ they are respectively 0.028221 and 0.072933 seconds.

M	BIL-EU	MC	BIL-AM
100	6.585345		17.570376
200	6.399257		17.598079
400	6.288664	6.197331	17.623151
800	6.243341	[6.187387-6.207276]	17.627720
1600	6.233008		17.629993
3200	6.208183		17.633762
6400	6.203391		17.634432
12800	6.194374		17.635186
25600	6.191878		17.635486

Table 6: *16-step Double Knock-Out Put Options Prices with* $\sigma = 0.3$, $r = 0.03$, $s_0 = 100$, $K = 110$, $T = 2$. *The barrier parameters are* $T_i = i \cdot 0.125$, $L_i = 70 - i$, $H_i = 130 + i$.

Acknowledgment: The authors want to thank Professor Lucia Caramellino for her valuable advice.

References

[1] Baldi, P., Caramellino, L. and Iovino, M. G. (1999): Pricing general barrier options: a numerical approach using sharp large deviations, *Mathematical Finance* **4**, 293-321.

[2] Boyle, P.P., Lau, S.H. (1994) : Bumping up against the barrier with the binomial method, *The Journal of Derivatives* **1**, 6-14.

[3] Cheuk, T.H.F., Vorst T.C.F. (1996): Complex barrier options, *The Journal of Derivatives* **4**, 8-22 (1996).

[4] Cox J., Ross S.A., Rubinstein M. (1979): Option Pricing: A simplified approach, *Journal of Financial Economics* **7**, 229-264.

[5] Dai, T., Lyuu Y. (2010): The Bino-Trinomial Tree: A Simple Model for Efficient and Accurate Option Pricing, *The Journal of Derivatives* Vol.17, **4**, 7-24 (2010).

[6] Figlewki, S., Gao, B. (1999): The Adaptive Mesh model: a New Approach to Efficient Option Pricing, *Journal of Financial Economics* **53**, 313-351.

[7] Gao, B., Huang. J., Subrahmanyam, M.G. (2000) : The Valuation of American Barrier Options Using the Decomposition Technique, *Journal of Economic Dynamics and Control* Vol.24 **11-12**, 1783-1827.

[8] Gaudenzi, M., Lepellere, M.A. (2006): Pricing and hedging American barrier options by a modified binomial method. *International Journal of Theoretical and Applied Finance* Vol.9, **4** , 533-553.

[9] Gaudenzi, M., Zanette, A. (2009): Pricing American barrier options with discrete dividends by binomial trees, *Decisions in Economics and Finance* Vol.32, **2** , 129-148.

[10] Gobet, E., (1999): Analysis of the zizag convergence for barrier options with binomial trees. Technical Report (Prépublication 536 du laboratoire PMA Paris 6, France) (Available at: http:// www.proba.jussieu.fr/mathdoc/preprints/).

[11] Guillame, T. (2003): Window Double Barrier Options, *Review of Derivatives Research*, **6**, 47-75 (2003).

[12] Guillame, T. (2010): Step Double Barrier Options, *The Journal of Derivatives* **Fall**, 59-79 (2010).

[13] Kunitomo M., Ikeda N. (1992) : Pricing options with curved boundaries, *Mathematical Finance* Vol.2, 275-298.

[14] Lin J., Palmer K. (2012) : Convergence of barrier option prices in the binomial model, *Mathematical Finance* Vol. 23, 318-338.

[15] Ritchken, P. (1995): On pricing barrier options. *The Journal of Derivatives*, **3**, 19-28.

[16] William H., Brian P. Flannery, Saul A. Teukolsky, William T. Vetterling (1986): Numerical Recipes: The Art of Scientific Computing, *Cambrigde University Press*, 80-82.