On pricing arithmetic average reset options with multiple reset dates in a lattice framework

Massimo Costabile, Ivar Massabó, Emilio Russo

Department of Business Administration, University of Calabria, Ponte Bucci cubo 3 C, 87036, Rende (CS), Italy

ARTICLE INFO

Article history:
Received 20 November 2009
Received in revised form 19 May 2011

Keywords:
Reset options
Binomial algorithms
Discrete time models

ABSTRACT

We develop a straightforward algorithm to price arithmetic average reset options with multiple reset dates in a Cox et al. (CRR) (1979) framework. The use of a lattice approach is due to its adaptability and flexibility in managing arithmetic average reset options, as already evidenced by Kim et al. (2003). Their model is based on the Hull and White (1993) bucketing algorithm and uses an exogenous exponential function to manage the averaging feature, but their choice of fictitious values does not guarantee the algorithm’s convergence (cfr., Forsyth et al. (2002)). We propose to overcome this drawback by selecting a limited number of trajectories among the ones reaching each node of the lattice, where we compute effective averages. In this way, the computational cost of the pricing problem is reduced, and the convergence of the discrete time model to the corresponding continuous time one is guaranteed.

© 2011 Elsevier B.V. All rights reserved.

1. Introduction

The increasing popularity of Asian-style options in financial markets makes their pricing problem a hot topic that is being studied by many authors looking for developing efficient pricing models. Indeed, the averaging feature is often desirable since it mitigates the sensitivity of the contract payoff to large price movements.

We propose an algorithm for pricing arithmetic average reset options characterized by multiple reset dates. The payoff of these securities is similar to that of plain-vanilla options, but, in addition, at some pre-determined reset dates they allow to update the strike price with the value of the arithmetic average of the asset prices registered during fixed monitoring windows. This means that the strike price of such options is stochastic depending upon the arithmetic average.

To illustrate the update of the strike price, we refer to a European reset call option, characterized by \( t_1, \ldots, t_N \) reset dates, and time to maturity \( T \), written on an underlying asset with value \( S_t \) at time \( t \) and dynamics described by a geometric Brownian motion. The option time to maturity is divided into \( N+1 \) monitoring windows for which, without loss of generality, we may suppose a constant length \( t_l - t_{l-1} = \frac{T}{N+1} \), with \( l = 1, \ldots, N+1 \), \( t_0 = 0 \) and \( t_{N+1} = T \). Let us denote by \( A_l \), \( l = 1, \ldots, N \), the arithmetic average of the underlying asset prices registered during the \( l \)th monitoring window, i.e., in time period \( (t_{l-1}, t_l) \), by \( K_0 \) the initial strike price used in the first monitoring window, and by \( K_l = \min(K_{l-1}, A_l), l = 1, \ldots, N \), the strike price used in the \((l+1)\)th monitoring window \( (t_l, t_{l+1}) \). At the end of the \( N \)th monitoring window, it lies at the last reset date, \( t_N \), where the strike price is finally modified according to \( K_N = \min(K_{N-1}, A_N) \) and remains unchanged up to maturity. Indeed, after the \( N \)th reset date (in the interval \( (t_N, T) \)), the reset call option becomes a standard

Corresponding author. Tel: +39 0984493387; fax: +39 0984492277.
E-mail addresses: massimo.costabile@unical.it (M. Costabile), i.massabo@unical.it (I. Massabó), emilio.russo@unical.it (E. Russo).

In the case of a put option with \( t_1, \ldots, t_N \) reset dates, the strike price is reset according to \( K_l = \max(K_{l-1}, A_l), l = 1, \ldots, N \), where \( K_0 \) is the strike price fixed at inception.

0377-0427/$ – see front matter © 2011 Elsevier B.V. All rights reserved.
doi:10.1016/j.cam.2011.05.041
Fig. 1. The initial strike price is $K_0$. Average $A_1$ is computed considering the underlying asset prices attained in the 1st monitoring window, and the strike price is reset at level $K_1$ at the 1st reset date. Average $A_2$ is computed upon the asset prices registered in the 2nd monitoring window and the strike price is reset at level $K_2$ at the 2nd reset date. The latter strike price is used up to maturity.

Contrary to the case of European options which reset the strike price to the current asset value or to a pre-specified function of the asset prices with known distribution, the pricing problem of arithmetic average reset options must be tackled with numerical approximation methods because their payoff distribution is not known, even in the simple log-normal framework. Among them, lattice-based methodologies represent a valuable resource for their flexibility and efficiency in managing path-dependent options; they are really appreciated by practitioners for their simplicity in implementation.

In a lattice framework, the main obstacle to price reset options with payoff depending upon the arithmetic average of the underlying asset prices is relative to the fact that the number of alternative arithmetic averages which can be realized at a node grows very fast with the number of time steps. To handle this complexity, a lattice-based model has been proposed in [9] who adapt the forward shooting grid method for valuing path-dependent options developed in [5] by adding two augmented state variables to the standard Cox et al. [10] (CRR) model. The first one, used to generate a set of representative strike prices at each node of the lattice, has the form $K_0 e^{-h t}$, where $K_0$ is the initial strike price, $h$ is a positive real number, and $k$ ranges between 0 and a suitable integer so that the set contains the possible minimum strike price at that node. The second state variable associates with each node a set of representative averages computed as $A_m e^{-mh t}$, where $A_m$ is the maximum possible value of the arithmetic average at the considered node and $m$ ranges between 0 and an integer $m^*$ which allows the minimum average $A_{min}$ to be included into the set. These exponential functions produce fictitious values both for the strike price and the arithmetic average which make the performance of the model strongly dependent upon the value assumed by the parameter $h$ and the interpolation technique chosen. In order to keep the algorithm computationally efficient, they choose $h$ proportional to $\sigma \sqrt{\Delta t}$ (where $\sigma$ is the underlying asset return volatility, while $\Delta t$ is the step length) but this choice does not assure the convergence of their discrete time model to the corresponding continuous time one. Indeed, as evidenced in [11], to achieve convergence in a forward shooting grid model when a linear interpolation procedure is applied, $h$ must be chosen proportional to the step length of the lattice, $\Delta t$.

The essence of our approach is to overcome the drawback of the Kim et al. algorithm by choosing sets of representative averages made up of effective values computed upon selected actual paths reaching each node of the lattice. Contrary to Kim et al., the proposed procedure does not rely upon any external parameter. At the declared reset dates, these averages are used to eventually update the strike price values. The backward recursion scheme, coupled with linear interpolation, furnishes a way to compute the option price at inception. The result is a model which reduces the computational complexity of the option evaluation problem and assures the convergence to the corresponding continuous time model (the convergence analysis is carried out in Appendix A). Finally, one of the main feature of the model is its flexibility which allows an immediate application to determine the option replicating portfolio and accommodates early exercise for valuing American options.

The rest of the paper is organized as follows. In Section 2, we present the binomial model for pricing arithmetic average reset options with multiple reset dates. Section 3 is devoted to illustrate the performance of our algorithm. Section 4 concludes with a summary.

---

2 To cite a few, Gray and Whaley [1,2] derive a valuation formula for the case of one reset date while Cheng and Zhang [3] present a closed form pricing formula generalized to the case of multiple reset dates. Furthermore, Kwok and Lau [4] propose a lattice model based on the algorithm developed in [5] for pricing path-dependent options. Models for options with different reset features or developed in different frameworks have been also proposed. Among others, we recall the recent contribution of Li et al. [6], which works in a stochastic interest rate framework, and of Yu and Shaw [7] who consider the case of an option with a snapshot reset feature.

3 Among others, for geometric average reset options, Cheng and Zhang [3] provide an explicit formula in the case of one reset date while Dai et al. [8] derive an analytic formula in the case of multiple monitoring windows as a corollary of a general formula used for pricing a large class of path-dependent options.
2. The binomial model for arithmetic average reset options

In this section, we illustrate the algorithm for pricing an arithmetic average reset option characterized by \( t_1, \ldots, t_N \) reset dates written on an asset, \( S_0 \), with dynamics described by a CRR lattice. At first, we provide a preliminary intuitive description of the algorithm; then, we present the procedure to generate the averages and the strike prices; finally, we present the backward recursion scheme to compute the option price at the lattice inception.

2.1. A preliminary presentation

Consider a call option with pre-determined \( t_1, \ldots, t_N \) reset dates with terminal payoff \[^4^] \( c(T) = \max(S_T - \min(K_0, A_1, \ldots, A_N), 0) \), where \( A_i, \ i = 1, \ldots, N \), is the average of the underlying asset values registered in the \((i + 1)\)th monitoring window, and \( K_0 \) is the initial strike price. Since the final strike price is determined at the last reset date, we implement an algorithm based on a backward recursive scheme that includes information on the values \( A_1, \ldots, A_N \). More explicitly, the option time to maturity is divided into \( N + 1 \) monitoring windows. In the first \( N \) windows, \( (t_1, t_{i+1}], \ i = 0, \ldots, N - 1 \), we associate with each node \( (t, S) \) of the CRR lattice a vector of arithmetic averages which will be used to eventually update at time \( t_{i+1} \) the strike prices then considered in the next monitoring window. In the \((N + 1)\)th period \( (t_N, T) \), after the last reset at time \( t_N \), the option becomes a plain-vanilla one with time to maturity \( T - t_N \). Hence, at time \( t_N \), in correspondence to each strike price, the option value may be computed via the usual formulas for plain-vanilla options. \[^5^] Then, we can proceed backward using the recursive procedure. Fixing the step length equal to \( \Delta t \), at time \( t = t_N - \Delta t \) in correspondence to the asset value \( S_t \), for each value of \( K(t) \) and \( A(t) \) we are able to calculate the option price by considering that at time \( t + \Delta t = t_N \) we have two possible scenarios for the asset value \( S_t \):

- With risk-neutral probability \[^6^] \( p, S_t \) shows an upward movement leading to asset value \( S_{t+\Delta t} = uS_t \), to an updated average value \( A_u(t + \Delta t) \) and, eventually, to a reset strike price \( K_u(t + \Delta t) = \min(K(t), A_u(t + \Delta t)) \). Hence, \( c_u(t + \Delta t) \) is easily computed;
- With risk-neutral probability \( q = 1 - p, S_t \) shows a downward movement leading to asset value \( S_{t+\Delta t} = dS_t \), to an updated average value \( A_d(t + \Delta t) \) and, eventually, to a reset strike price \( K_d(t + \Delta t) = \min(K(t), A_d(t + \Delta t)) \). Then, \( c_d(t + \Delta t) \) is easily computed.

The option price at time \( t \) is computed through the following recursive formula,

\[
e^{-r\Delta t} \left[ pc_u(t + \Delta t) + qc_d(t + \Delta t) \right].
\]

At previous time \( t = t_N - 2\Delta t \), keeping track of the already found option values at all nodes at time \( t_N - \Delta t \), we shall compute the option prices taking into account the following scenarios at time \( t_N - \Delta t \):

- With risk-neutral probability \( p, S_t \) shows an upward movement leading to asset value \( S_{t+\Delta t} = uS_t \) and to an updated average value \( A_u(t + \Delta t) \). Whenever \( t + \Delta t \) is not a reset date, the vector of the strike prices is still \( K(t) \). Otherwise, if \( t + \Delta t \) is a reset date, \( K_u(t + \Delta t) = \min(K(t), A_u(t + \Delta t)) \). Then, pick the option price associated with node \( (t + \Delta t, S_{t+\Delta t}) \) in correspondence to \( A_u(t + \Delta t) \) and \( K_u(t + \Delta t) \). If there is no perfect match of the updated average \( A_u(t + \Delta t) \) with the representative averages at node \( (t + \Delta t, S_{t+\Delta t}) \), then linearly interpolate the option prices in correspondence to the two closest representative averages to \( A_u(t + \Delta t) \). Denote the interpolated value by \( c_u(t + \Delta t) \).
- With risk-neutral probability \( q = 1 - p, S_t \) shows a downward movement leading to asset value \( S_{t+\Delta t} = dS_t \) and to an updated average value \( A_d(t + \Delta t) \). Whenever \( t + \Delta t \) is not a reset date, the vector of the strike prices is still \( K(t) \). Otherwise, if \( t + \Delta t \) is a reset date, \( K_d(t + \Delta t) = \min(K(t), A_d(t + \Delta t)) \). Then, pick the option price associated with node \( (t + \Delta t, S_{t+\Delta t}) \) in correspondence to \( A_d(t + \Delta t) \) and \( K_d(t + \Delta t) \). If there is no perfect match for the updated average \( A_d(t + \Delta t) \) with the representative averages at time \( t + \Delta t \) at node \( (t + \Delta t, S_{t+\Delta t}) \), then interpolate as mentioned before. Denote the interpolated value by \( c_d(t + \Delta t) \).

The option price at time \( t \) is computed using (1) and this is the option price associated with \( (t, S_t) \) in correspondence to the proper \( A(t) \) and \( K(t) \). Proceeding in this way, we compute the option prices at any \( (t, S_t) \) with \( t = t_N - 3\Delta t, \ldots, \Delta t, 0 \). Clearly, when \( t \) coincides with a reset date, the strike price \( K(t) \) is reset only when the average value is smaller than the current strike price, in the call case, or when the average value is greater than the current strike price, in the put case.

\[^4^\] In the put case, the payoff is \( p(T) = \max(\min(K_0, A_1, \ldots, A_N), -S_T, 0) \).

\[^5^\] For European-style options, the Black-Scholes formula may be used while, for the American counterparts of the contracts, option values may be computed through the analytical approximation proposed in [12].

\[^6^\] The values assumed by \( p, u, \) and \( d \) mentioned here are specified at the beginning of the next section when the CRR model is briefly presented.
2.2. The average and strike price selection

In this subsection, we detail the algorithm to build up the vectors of the averages used to define the vectors of the strike prices. The dynamics of the underlying asset, \( S_t \), during the option time to maturity \( T \) is modeled by a CRR lattice based on \( n \) time steps of length \( \Delta t = T/n \). At each time step, the asset value \( S_t \) increases by factor \( u = e^{\sigma \sqrt{\Delta t}} \) if an upward jump occurs, or it decreases by factor \( d = 1/u \) if a downward jump takes place. Here, \( \sigma \) is the asset return volatility, \( p = e^{-\Delta t - \frac{1}{2} \sigma^2 \Delta t} \) is the risk-neutral probability of an up step, \( q = 1 - p \) is the probability of a down step, and \( r \) is the risk-free interest rate. Without loss of generality, we assume \( t_0 = 0 \) as the contract inception where the asset value is equal to \( S \) and denote the asset value by \( S(i, j) = S u^i d^j \) at node \( (i, j) \), \( i = 0, \ldots, n \) and \( j = 0, \ldots, i \), after \( j \) up steps and \( i-j \) down steps. Furthermore, for the sake of simplicity, we choose \( n \) as a multiple of \( N+1 \) so that \( \delta = \frac{n}{N+1} \) steps fall in each monitoring window, and the \( (\delta) \)th step, \( i = 1, \ldots, N \), coincides with the \( l \)th reset date, \( t_l \).

The main feature of this type of reset option, which updates the strike price with the arithmetic average at the pre-specified reset dates, induces a computational problem: the huge number of possible averages computed following all the trajectories reaching each node of the lattice. Indeed, in general, each path produces a different arithmetic average and, to reduce the computational complexity of the pricing problem, we select a limited number of representative trajectories in order to associate with each node of the lattice a subset of effective averages as we will explain hereafter. With this aim, we choose to work in the CRR framework because of its simplicity in presenting the proposed pricing algorithm, but the selection procedure may be easily adapted to multinomial lattice frameworks.

At first, we present the procedure to choose the representative averages associated with the nodes located at the first reset date \( t_1 = \delta \Delta t \). Starting from inception, i.e., node \( (0, 0) \), node \( (\delta, j) \) is reached after \( j \) up steps and \( \delta - j \) down steps in \( \binom{j}{\delta} \) possible ways. Among them, we select by an iterative procedure only \( \eta(\delta, j) = 1 + j(\delta-j) \) trajectories, and we compute the arithmetic average of the asset values upon each one of these trajectories. In this way, we associate with node \( (\delta, j) \) a vector, \( A(\delta, j) \), of representative effective averages with \( \eta(\delta, j) \) components.

At node \( (\delta, j) \), the selection procedure starts by computing the maximum average value, which is produced by the trajectory \( t_{\max}(\delta, j) \) with \( j \) consecutive up steps followed by \( \delta - j \) down steps. We denote it by \( A(\delta, j; 1) \) because it is assigned to the first component in the vector \( A(\delta, j) \):

\[
A(\delta, j; 1) = \frac{1}{\delta+1} \left( \sum_{h=0}^{j} S^h u^{\delta-j-1} + \sum_{h=0}^{\delta-j} S^h d^{j-2-\delta} \right).
\]

The minimum average, \( A(\delta, j; 0) \), is produced by the trajectory \( t_{\min}(\delta, j) \), represented by the path with \( \delta - j \) consecutive down steps followed by \( j \) up steps. It will be the last component in the vector \( A(\delta, j) \), and it is computed by

\[
A(\delta, j; \eta(\delta, j)) = \frac{1}{\delta+1} \left( \sum_{h=0}^{\delta-j} S^h d^j + \sum_{h=0}^{j-1} S^h u^{j-2-\delta} \right).
\]

The other representative averages, \( A(\delta, j; a) \), \( a = 2, \ldots, \eta(\delta, j) - 1 \), are computed recursively through a step by step procedure. Assume that the \( \eta \)th representative average, \( A(\delta, j; a) \), has already been computed on the generic trajectory \( \tau(\delta, j) = \{(x_{j_k}), x = 0, \ldots, \delta; j_0 = 0, j_k = j\}, \) not coinciding with \( t_{\min}(\delta, j) \), characterized by the asset values \( S(x_{j_k}) \). To compute \( A(\delta, j; a+1) \), we proceed as follows:

Step 1: Among all nodes \( (x_{j_k}) \) belonging to \( \tau(\delta, j) \) with \( x \in [1, \delta - 1] \), we detect only those where the underlying asset has registered the maximum value, \( S_{\max}(x_{j_k}) \).

Step 2: Among them, we select the node corresponding to the minimum value assumed by \( x_{j_k} \) (i.e., node \( (x_{\min}, j_{\min}) \)) such that \( (x_{\min}, j_{\min}) \) does not belong to \( t_{\min}(\delta, j) \) and the new path generated by substituting in \( \tau(\delta, j) \), node \( (x_{\min}, j_{\min}) \) with node \( (x_{\min}, j_{\min} - 1) \) still reaches node \( (\delta, j) \).

Step 3: The \((a+1)\)th representative average is computed on this new trajectory or, alternatively, it is simply obtained from the previous one, \( A(\delta, j; a) \), by substituting \( S_{\max}(x_{\min}, j_{\min}) \) with \( S_{\max}(x_{\min}, j_{\min}) \), i.e.,

\[
A(\delta, j; a+1) = A(\delta, j; a) - \frac{1}{\delta+1} S_{\max}(x_{\min}, j_{\min}) \left[ 1 - d^2 \right].
\]

The procedure continues as long as the last trajectory, \( t_{\min}(\delta, j) \), is reached and, since we start from \( t_{\max}(\delta, j) \), this happens after \( j(\delta - j) \) substitutions. Clearly, the number of paths detected to compute the representative averages at node \( (\delta, j) \) is in a one-to-one correspondence with the number of nodes that lay between the lowest path, \( t_{\min}(\delta, j) \), and the highest one, \( t_{\max}(\delta, j) \), including the nodes belonging to \( t_{\max}(\delta, j) \), and excluding \((0, 0), (\delta, j) \) and the other ones belonging to \( t_{\min}(\delta, j) \).
Such a correspondence is justified by the fact that each new trajectory generated by the iterative procedure has one and only one different node with respect to the antecedent trajectory.\footnote{The vectors of the representative averages that we propose are generated starting from the highest trajectory, $\tau_{\text{max}}(\delta, j)$, and ending at the lowest one, $\tau_{\min}(\delta, j)$. The same vectors of representative averages could be generated in a symmetrical way simply starting from the average associated with the lowest path, $\tau_{\min}(\delta, j)$, and ending at the one associated with the highest path, $\tau_{\text{max}}(\delta, j)$, but by substituting in each iteration only the minimum value $S_{\min}(\delta, j_{\text{min}})$ with value $S_{\min}(\delta, j_{\text{max}})$.}

The following example clarifies how the representative trajectories are selected for each node $(\delta, j), j = 0, \ldots, \delta$. Fig. 2 illustrates the binomial evolution of the underlying asset price during the first monitoring window where we suppose $\delta = 4$ time steps. Consider, at first, the trajectories reaching node $(4, 4)$. Since there is only one trajectory, there is only one arithmetic average computed upon $(S, Su, Su^2, Su^3, Su^4)$. For node $(4, 2)$, the vector of the averages has five components. The first component, $A(4, 2; 1)$, is computed upon trajectory $\tau_{\text{max}}(4, 2) = (S, Su, Su^2, Su^3, Su^4)$. Here, the maximum value is $S_{\max}(2, 2) = Su^2$ and, consequently, the second component, $A(4, 2; 2)$, is computed upon path $(S, Su, Su, S, S)$ obtained from the previous one by substituting $S_{\max}(2, 2)$ with $S_{\max}(2, 2)d^2 = S$. The maximum value over this trajectory is now reached two times, i.e., $S_{\max}(1, 1) = S_{\max}(3, 2) = Su$. In this case, the algorithm selects value $S_{\max}(1, 1)$ and substitutes it with $S_{\max}(1, 1)d^2 = Su$. Hence, the third component, $A(4, 2; 3)$, is computed using values $(S, Sd, S, Su)$. The remaining averages associated with node $(4, 2)$ are computed upon trajectories $(S, Sd, Sd, Sd)$ and $(S, Sd, Sd^2, Sd, S)$ = $\tau_{\text{min}}(4, 2)$. Following this procedure, the vector of the representative averages associated with node $(4, 2)$ contains all the effective averages except the one generated by path $(S, Su, Sd, S)$ (depicted in Fig. 2 by thick lines marked with terminal arrows).

Once the vector of representative averages is associated with each node $(\delta, j)$ of the lattice relative to the first reset date $t_1$, the next step of the algorithm consists in determining the option strike price at time $t_1$. For each average, $A(\delta, j; a), a = 1, \ldots, \eta(\delta, j)$, eventually let update $K(\delta, j; a) := \min(K_0, A(\delta, j; a))$. Consequently, the total number of different strike prices, $\eta(\delta, j)$, associated with each node, $(\delta, j), j = 0, \ldots, \delta$, would be less than or equal to $\eta(\delta, j)$ because $K(\delta, j; a)$ coincides with $K_0$ for all averages $A(\delta, j; a)$ such that $A(\delta, j; a) \geq K_0$.

Now, we need to identify the strike prices, $K(\delta, j; k), j = 0, \ldots, \delta, k = 1, \ldots, \eta(\delta, j)$, which influence the option values at each node $(i, j)$, in the second monitoring window $(t_1, t_2)$. They are simply all the strike prices associated with nodes $(\delta, x)$ lain on the first reset date with $x$ assuming all integer values in the interval $[\max(j - i + \delta, 0), \min(j, \delta)]$. Indeed, no other node at the first reset date may be touched by the paths reaching node $(i, j)$. Fig. 3 illustrates the nodes of the lattice that influence the choice of the strike prices at node $(7, 2)$ belonging to the second monitoring window when, as before, $\delta = 4$. They are the ones associated with nodes $(4, 0), (4, 1), (4, 2)$ because all paths reaching node $(7, 2)$ are enclosled into the quadrilateral $ABCD$, depicted by thick lines in Fig. 3. Consequently, the number of all possible strike prices, $\phi^*_2(i, j)$, considered at node $(i, j)$, belonging to the second monitoring window is given by:

$$\phi^*_2(i, j) = \min_{x = \max(j - i + \delta, 0)} \phi_1(\delta, x).$$

Among the $\phi^*_2(i, j)$ strike prices, some values may be repeated; thus the algorithm sorts only $\phi_2(i, j) \leq \phi^*_2(i, j)$ different ones.

At the second reset date, $t_2$, coinciding with the $(2\delta)$th step of the CRR lattice, the strike price may be reset depending upon the values of the arithmetic averages. In general, once the procedure is defined in the $l$th monitoring window, $l = 1, \ldots, N - 1$, the algorithm captures the option path-dependency by computing the selected averages $A(i, j; a)$ at
At moving toward reset date $t$, the minimum average associated with node $A_{\downarrow}$ down steps. Hence, the first component of the vector of averages, $A(i, j)$, is given by

$$A(i, j; 1) = \frac{1}{i - \delta} \left( \sum_{x=0}^{\min(i-j, i-\delta-1)} S^i u^d e^{-\delta x} + \sum_{x=\min(i-j, i-\delta-1)+1}^{i-\delta-1} S^i u^{i-x+\min(i-j, i-\delta-1)} \right).$$

The minimum average associated with node $(i, j)$ is the last component in the vector of averages:

$$A(i, j; \eta(i, j)) = \frac{1}{i - \delta} \left( \sum_{x=0}^{\min(j, i-\delta-1)} S^j u^d e^{-\delta x} + \sum_{x=\min(j, i-\delta-1)+1}^{i-\delta-1} S^j u^{j-x+\min(j, i-\delta-1)} \right).$$

Fig. 3. The nodes of the lattice that influence the choice of the strike prices at node $(7, 2)$. Each node $(i, j)$ in the next monitoring window $(t_1, t_{i+1})$. In particular, the algorithm selects from all possible trajectories belonging to the $(l + 1)$th monitoring window a fixed number of representative paths, $\eta(i, j)$, provided in the following proposition (the proof is given in Appendix B).

**Proposition.** In a binomial lattice characterized by $n$ time steps modeling the dynamics of the underlying asset for an option with $t_1, \ldots, t_N$ reset dates, the total number of representative trajectories, $\eta(i, j)$, belonging to the $(l + 1)$th monitoring window with $l = 1, \ldots, N - 1$ and reaching the node $(i, j)$, is

$$\eta(i, j) = 1 + \frac{1}{2} \left[ \min(i - \delta, \min(j, i - j)) (2i - 2\delta - \min(i - \delta, \min(j, i - j)) - 1) \right].$$

We depict now how to compute the arithmetic averages upon the selected paths reaching each node in the generic $(l + 1)$th monitoring window. The total number of underlying asset values upon each path reaching node $(i, j)$ is given by $i - \delta$ because the first asset value falling into the $(l + 1)$th monitoring window is registered at the $(l + 1)$th step of the lattice. The maximum average associated with node $(i, j)$ is produced by trajectory $\tau_{\text{max}}(i, j)$ which, starting from node $(i, j)$ and moving toward reset date $t_1$, is characterized by $\min(i - j, i - \delta - 1)$ up steps followed by $i - \delta - 1 - \min(i - j, i - \delta - 1)$ down steps. Hence, the first component of the vector of averages, $A(i, j; 1)$, is equal to
The other representative averages at node \((i,j), A(i, j; a), a = 2, \ldots, \eta(i, j) - 1\), are computed recursively as outlined below. Let \( \tau(i, j) = \{(x, j_a), x = l\delta + 1, \ldots, i; j = j\} \) be the trajectory, not coinciding with \( \tau_{\min}(i, j) \), upon which \( A(i, j; a) \) has been already computed. The \((a + 1)\)th average \( A(i, j; a + 1) \) is obtained following the iterative procedure presented from Steps 1 to 3 with two differences: in Step 1, we have to consider nodes \((x, j_a)\) belonging to \( \tau(i, j) \) characterized by \( x \in [l\delta + 1, i - 1] \) (the last node \((i, j)\) is fixed in all the selected trajectories); in Step 3, relation (2) must be replaced by

\[
A(i, j; a + 1) = A(i, j; a) - \frac{1}{i - l\delta} S_{\max}(x_{\min}, J_{\min}) \left[ 1 - d^2 \right].
\]

(3)

This procedure is applied recursively \( \eta(i, j) - 1 \) times as long as the last trajectory, \( \tau_{\min}(i, j) \), is reached.

To clarify the selection procedure, we consider the evolution of the underlying asset in the second monitoring window (for simplicity, \( \delta = 4 \) in Fig. 4). At first, consider nodes \((5, j), j = 0, \ldots, 5\). The averages are equal to the asset price values (i.e., \( A(5, j; 1) = S(5, j) \)) because we are just one step after the first reset date. Consider now node \((8, 8)\). Since there is only one trajectory, there is only one arithmetic average computed through values \((S^5, S^6, S^7, S^8)\). At node \((8, 6)\), instead, the components of the vector of averages are the following. The first one, \( A(8, 6; 1) \), is computed upon \( \tau_{\max}(8, 6) = (S^5, S^6, S^7, S^8) \). Being \( S_{\max}(8, 5, 6, 7) \) the maximum value on \( \tau_{\max}(8, 6) \), the second average, \( A(8, 6; 2) \), is computed on path \((S^5, S^4, S^5, S^8)\) obtained from the previous one by substituting \( S_{\max}(8, 5, 6, 7) = S^4 \). The maximum value on the latter trajectory is now reached two times, \( S_{\max}(5, 5, 6, 7) = S^2 \). Following this procedure, the vector of representative averages associated with node \((8, 6)\) contains all the effective averages except the one generated by path \((S^5, S^4, S^5, S^8)\) (depicted in Fig. 4 by thick lines marked with terminal arrows).

Now, we are in the position to compute the strike prices associated with each node \((l + 1)\delta, j) at reset date \( t_{l+1} \). For example, for generic node \((2\delta, j)\) we have to compare the strike prices associated with all nodes \((\delta, x), x = \max(j - \delta, 0), \ldots, \min(j, \delta)\), with each average \( A(2\delta, j; a) \) in order to obtain the strike prices at the second reset date. The total number of strike prices, \( \varphi_2'(2\delta, j) \), associated with each node \((2\delta, j)\) is consequently given by:

\[
\varphi_2'(2\delta, j) = \eta(2\delta, j) \left( \sum_{x = \max(j - 3, 0)}^{\min(j, \delta)} \varphi_1(\delta, x) \right).
\]
Again, among \( \varphi'_2(2\delta, j) \) strike prices, some values may be repeated and the algorithm sorts only \( \varphi_2(2\delta, j) \) different ones.

Generally, by considering the \((l + 1)\)th monitoring window with \( l = 1, \ldots, N - 1 \), all possible strike prices, \( K(i; j; k) \), \( k = \varphi'_2(2\delta, j) \), associated with each node, \((i, j), i = l + 1, \ldots, (l + 1)\delta - 1, j = 0, \ldots, i \), are the ones associated with nodes \((\delta, x)\) lain on the \(l\)th reset date, where \(x\) assumes all integer values in the interval \( [\max (j - i + \delta, 0), \min (j, \delta)) \).

The total number of possible strike prices, \( \varphi_{l+1}(i, j) \), is clearly given by:

\[
\varphi_{l+1}(i, j) = \min_{x=\max(j-l+l\delta,0)} \varphi_l(\delta, x),
\]

but in order to reduce the computational cost of the algorithm, only \( \varphi_{l+1}(i, j) \) different ones are considered with \( \varphi_{l+1}(i, j) \leq \varphi_{l+1}(i, j) \). At reset date \( t_{l+1} \), at each node \((l + 1)\delta, j \), \( j = 0, \ldots, (l + 1)\delta \), the strike prices are computed by comparing each average associated with that node, \( A((l + 1)\delta; j; a), a = 1, \ldots, \eta ((l + 1)\delta, j) \), with each strike price associated with nodes \((\delta, x), x = \max (j - \delta, 0), \ldots, \min (j, \delta) \), i.e., \( K(\delta; x; k) \), \( k = 1, \ldots, \varphi_l(\delta, j) \). The total number of strike prices, \( \varphi_{l+1}((l + 1)\delta, j) \), associated with node \((l + 1)\delta, j \) is consequently given by:

\[
\varphi_{l+1}((l + 1)\delta, j) = \min_{x=\max(j-l+l\delta,0)} \varphi_l(\delta, x),
\]

but the algorithm considers only \( \varphi_{l+1}((l + 1)\delta, j) \) different ones with \( \varphi_{l+1}((l + 1)\delta, j) \leq \varphi_{l+1}(l + 1)\delta, j \).

At this point, with the presented procedure, we have associated with each node \((i, j)\) belonging to the first \(N\) monitoring windows two vectors, the one containing a selected number of effective arithmetic averages and the other one made up of different strike prices.

In the \((N + 1)\)th monitoring window, during period \((T_N, T)\) no more resets happen and the option is similar to a plain-vanilla one. It is worth noting that the average and the strike price are the state variables involved in the backward induction scheme used on the lattice to compute the option price at inception.

2.3. The recursive backward scheme

We focus attention on an arithmetic average reset call option. In the last monitoring window, it reduces to a plain-vanilla one so that we can start the backward recursion scheme from nodes \((N\delta - 1, j; \delta, 0, 0, \ldots, N\delta - 1, j; \delta, 0)\), immediately before the last reset date \(T_N = N\delta\). Indeed, at \(T_N\) we can compute the option values through classical formulas. For each average value \(A(N\delta; j; a), a = 1, \ldots, \eta (N\delta - 1, j)\), associated with node \((N\delta - 1, j)\) there are the following possible scenarios:

- With probability \(p\), asset value \(S(N\delta - 1, j)\) shows an upward movement to \(S(N\delta, j + 1)\) so that the average at the next time step is \(A(N\delta, j + 1; a_k) = e^{-\delta t} A(N\delta - 1, j; a_k) + S(N\delta, j + 1)\).

  In such a case, since the next time step coincides with reset date \(T_N\), we compute the option value in correspondence to asset price \(S(N\delta, j + 1)\) and strike price \(min(K(N\delta - 1, j; k), A(N\delta, j + 1; a_k)); k = 1, \ldots, \varphi_N(N\delta - 1, j)\).

  In the European case, in order not to introduce further approximations in the pricing process, in correspondence to each strike price use the Black–Scholes [13] explicit formula\(^8\) to compute the price of an option with initial asset value \(S(N\delta, j + 1)\) and time to maturity \(T/T_N\). By considering the generic \(k\) strike price associated with node \((N\delta - 1, j)\), we denote the corresponding option price by \(c(N\delta, j + 1; a_k)\).

- With probability \(q\), asset value \(S(N\delta - 1, j)\) shows a downward movement to \(S(N\delta, j)\) so that the average at the next time step is \(A(N\delta, j; a_k) = e^{-\delta t} A(N\delta - 1, j; a_k) + S(N\delta, j)\).

  In such a case, we compute the option value in correspondence to asset price \(S(N\delta, j)\) and strike price \(min(K(N\delta - 1, j; k), A(N\delta, j; a_k)); k = 1, \ldots, \varphi_N(N\delta - 1, j)\) through the Black–Scholes [13] formula. By considering the generic \(k\) strike price, we denote the corresponding option price by \(c(N\delta, j; a_k)\).

The reset call option price \(c(N\delta - 1, j; a, k)\) associated with node \((N\delta - 1, j)\) in correspondence to the \(a\)th average and the \(k\)th strike price is computed by

\[
c(N\delta - 1, j; a, k) = e^{-\delta t} [pc(N\delta, j + 1; a_k) + qc(N\delta, j; a_k)].
\]

In order to calculate the reset option price at the lattice inception, we adopt the backward recursive scheme described in Section 2.1, which starts from the known values \(c(N\delta - 1, j; a, k)\) associated with nodes \((N\delta - 1, j)\). In general, reset

---

\(^8\) The pricing of a European standard call option for each strike price is very time consuming following a straightforward CRR approach. Indeed, following the method suggested in [12], we should build up a new binomial lattice which represents the dynamics of value \(S(N\delta, j)\) over the last monitoring window and, on that lattice, we have to price a standard option with strike price \(min(K(N\delta - 1, j; k), A(N\delta, j + 1; a_k)); k = 1, \ldots, \varphi_N(N\delta - 1, j)\), and time to maturity \(T/T_N\). Furthermore, the use of other numerical techniques, e.g., finite differences or Monte Carlo methods, or the use of the adaptive mesh model suggested in [14] may make the pricing problem more complex on a computational point of view even if they provide very precise option prices. In the case of American options, the analytical approximation proposed in [12] is suitable to compute the option prices for each strike price and it is not time consuming.
call option price \( c(i, j; a, k) \) associated with node \((i, j)\), where \( i = l\delta + 1, \ldots, (l + 1)\delta, j = 0, \ldots, i, a = 1, \ldots, \eta(i, j), k = 1, \ldots, \varphi_{i+1}(i, j) \) and \( l = 0, \ldots, N - 1 \), is computed by

\[
c(i, j; a, k) = e^{-\gamma dt}[pc(i + 1, j + 1; a_u, k_u) + qc(i + 1, j; a_d, k_d)],
\]

where quantities \( c(i + 1, j + 1; a_u, k_u) \) and \( c(i + 1, j; a_d, k_d) \) are calculated by linear interpolation according to the cases we will present hereafter. For a detailed description of the linear interpolation technique, it is useful to classify generic node \((i, j)\), \( i = l\delta + 1, \ldots, (l + 1)\delta, j = 0, \ldots, i, \) belonging to the \((l + 1)\)th monitoring window \( l = 0, \ldots, N - 1 \) into one of the following three categories: in the first one, we include nodes \((i, j)\) with \( i = l\delta + 1, \ldots, (l + 1)\delta - 2; \) the second one includes nodes \((i, j)\) immediately before each reset date (i.e., \( i = (l + 1)\delta - 1; \)) in the last one, we include nodes \((i, j)\) in correspondence to the \((l + 1)\)th reset date (i.e., \( i = (l + 1)\delta \)).

Case 1: \( i = l\delta + 1, \ldots, (l + 1)\delta - 2 \).

\[
c(i + 1, j + 1; a_u, k_u) \text{ is the call option price in correspondence to average}
\]

\[
A(i + 1, j + 1; a_u) = \begin{cases} 
(i + 1)A(i, j; a) + uS(i, j) & \text{if } l = 0 \\
(i - l\delta)A(i, j; a) + uS(i, j) & \text{if } l = 1, \ldots, N - 1,
\end{cases}
\]

and strike price

\[
K(i + 1, j + 1; k_u) = \begin{cases} 
K_0 & \text{if } l = 0 \\
K(i, j; k) & \text{if } l = 1, \ldots, N - 1
\end{cases}
\]

because the strike price remains unchanged from the \((l\delta + 1)\)th step to the \(((l + 1)\delta - 1)\)th one. As a matter of fact, when there is an upward movement in the asset price, the \( l \)th average at node \((i, j)\), \( A(i, j; a) \), leads to average value \( A(i + 1, j + 1; a_u) \) at node \((i + 1, j + 1)\). Since we consider only a selected subset of effective averages, \( A(i + 1, j + 1; a_u) \) could be in the vector of the representative averages associated with node \((i + 1, j + 1)\) and the option price would be immediately available. In the other cases, \( c(i + 1, j + 1; a_u, k_u) \) is computed using a linear interpolation technique that starts by selecting, among the representative averages associated with node \((i + 1, j + 1)\), the closest ones, \( A(i + 1, j + 1; a_1) \) and \( A(i + 1, j + 1; a_2) \), to \( A(i + 1, j + 1; a_u) \) such that \( A(i + 1, j + 1; a_1) < A(i + 1, j + 1; a_u) \leq A(i + 1, j + 1; a_2) \). Then, the quantity

\[
\omega(i + 1, j + 1; a_u) = \frac{A(i + 1, j + 1; a_u) - A(i + 1, j + 1; a_1)}{A(i + 1, j + 1; a_2) - A(i + 1, j + 1; a_1)},
\]

is computed and the option price, \( c(i + 1, j + 1; a_u, k_u) \), is given by

\[
c(i + 1, j + 1; a_u, k_u) = c(i + 1, j + 1; a_1, k_u) + \omega(i + 1, j + 1; a_u) [c(i + 1, j + 1; a_2, k_u) - c(i + 1, j + 1; a_1, k_u)].
\]

The same interpolation technique and the same observations may be addressed to \( c(i + 1, j; a_d, k_d) \) which is the call option price in correspondence to strike price

\[
K(i + 1, j; k_d) = \begin{cases} 
K_0 & \text{if } l = 0 \\
K(i, j; k) & \text{if } l = 1, \ldots, N - 1,
\end{cases}
\]

and average

\[
A(i + 1, j; a_d) = \begin{cases} 
(i + 1)A(i, j; a) + dS(i, j) & \text{if } l = 0 \\
(i - l\delta)A(i, j; a) + dS(i, j) & \text{if } l = 1, \ldots, N - 1,
\end{cases}
\]

because the \( l \)th average at node \((i, j)\), \( A(i, j; a) \), leads to average value \( A(i + 1, j; a_d) \) at node \((i + 1, j)\) when a downward movement in the asset value takes place.

Case 2: \( i = (l + 1)\delta - 1 \).

\( c(i + 1, j + 1; a_u, k_u) \) is the call option price in correspondence to \( A(i + 1, j + 1; a_u) \), defined by (5), and

\[
K(i + 1, j + 1; k_u) = \begin{cases} 
\min(K_0, A(i + 1, j + 1; a_u)) & \text{if } l = 0 \\
\min(K(i, j; k), A(i + 1, j + 1; a_u)) & \text{if } l = 1, \ldots, N - 1,
\end{cases}
\]

because the \((i + 1)\)th step of the lattice coincides with the \((l + 1)\)th reset date where the strike price is updated. If \( A(i + 1, j + 1; a_u) \) is an element of the vector of the representative averages associated with node \((i + 1, j + 1)\), then \( c(i + 1, j + 1; a_u, k_u) \) is the option price in correspondence to \( K(i + 1, j + 1; k_u) \). Otherwise, if \( A(i + 1, j + 1; a_u) > K(i, j; k) \) \( (A(i + 1, j + 1; a_u) > K_0 \text{ if we are in the first monitoring window}) \), the strike is not reset and \( c(i + 1, j + 1; a_u, k_u) \) is the option price in correspondence to \( K(i, j; k) \). In case \( A(i + 1, j + 1; a_u) \leq K(i, j; k) \) \( (A(i + 1, j + 1; a_u) \leq K_0 \text{ if we are in the first monitoring window}) \), the strike is reset and linear interpolation must
be used to compute \( c(i + 1, j + 1; a_u, k_u) \). In such a case, after selecting from the representative averages associated with node \((i + 1, j + 1)\), the closest ones, \(A(i + 1, j + 1; a_1)\) and \(A(i + 1, j + 1; a_2)\), to \(A(i + 1, j + 1; a_u)\) such that \(A(i + 1, j + 1; a_1) < A(i + 1, j + 1; a_u) \leq A(i + 1, j + 1; a_2)\), the linear interpolation is based on strike prices

\[
K(i + 1, j + 1; k_1) = \begin{cases} 
\min(K_0, A(i + 1, j + 1; a_1)) & \text{if } l = 0 \\
\min(K(i, j; k), A(i + 1, j + 1; a_1)) & \text{if } l = 1, \ldots, N - 1,
\end{cases}
\]

and

\[
K(i + 1, j + 1; k_2) = \begin{cases} 
\min(K_0, A(i + 1, j + 1; a_2)) & \text{if } l = 0 \\
\min(K(i, j; k), A(i + 1, j + 1; a_2)) & \text{if } l = 1, \ldots, N - 1.
\end{cases}
\]

The same considerations may be similarly addressed to \((i + 1, j; a_d, k_d)\) which is the call option price in correspondence to \(A(i + 1, j; a_d)\) and

\[
K(i + 1, j; k_d) = \begin{cases} 
\min(K_0, A(i + 1, j; a_d)) & \text{if } l = 0 \\
\min(K(i, j; k), A(i + 1, j; a_d)) & \text{if } l = 1, \ldots, N - 1.
\end{cases}
\]

**Case 3:** if \(i = (l + 1)\delta\).

\(c(i + 1, j + 1; a_u, k_u)\) is the call option price in correspondence to \(K(i + 1, j + 1; k_u) = K(i, j; k)\), because it is determined at the \((l + 1)\)th reset date coinciding with the \(l\)th step of the lattice, and \(A(i + 1, j + 1; a_u) = A(i + 1, j + 1; 1) = S(i, j, 1) = uS(i, j)\), because the \((l + 1)\)th step of the lattice is the first one falling into the \((l + 2)\)th monitoring window. Consequently, the unique average associated with that node equals the underlying asset price; thus \(c(i + 1, j + 1; a_u, k_u) = c(i + 1, j + 1; 1, k)\) and no interpolation is required in such a case. Similarly, \(c(i + 1, j; a_d, k_d) = c(i + 1, j; 1, k)\).

To complete the description of the algorithm, it is worth observing that once the backward procedure reaches nodes \((1, j), j = 0, 1\), there is only one average and the initial strike price associated with each node, and the option price at inception is computed by

\[c(0, 0, 0, 1) = e^{-r\cdot T} \left[ p C(1, 1, 1, 1) + q C(1, 0, 0, 1) \right].\]

Recalling that a lattice approach is straightforward applicable to American-style options, the model is easily adapted by two simple devices. The first one concerns the option price computation on the last monitoring window where the reset option becomes a plain-vanilla one. In the American case, a numerical scheme to approximate the American option price at the last reset date \(t_T\) has to be used. An efficient way is the use of the Barone Adesi–Whaley [12] analytical approximation that is easily applicable in our framework and not time consuming. Then, starting from \(t_T\) and proceeding backward, the iterative formula may be modified by simply taking into account the early exercise option value. As an example, in the case of an American reset put option, \(P\), the recursive formula \((4)\) is modified as

\[
P(i, j; a, k) = \max \left\{ e^{-r\cdot T} \left[ p P(i + 1, j + 1; a_u, k_u) + q P(i + 1, j; a_d, k_d) \right], K(i, j; k) - S_u d^j \right\}.
\]

It is also worth emphasizing that an American arithmetic average reset call option with multiple reset dates written on a non-dividend paying asset presents the same price as its European counterpart. Simple replicating portfolio arguments may prove that the early exercise of such an option is never optimal. No convenience in early exercising the American version of the call contract means that the early exercise feature is valueless and, consequently, the American call price equals the European one because the option-holder prefers to keep it up to maturity.

### 3. Numerical results

We test the pricing model presented in Section 2 by computing the prices of different European arithmetic average reset options, both of call and put type, characterized by one, two and three reset dates, respectively.

At first, to assess the goodness of the model, we provide a comparison between the results provided by our algorithm and the ones provided by in [9]. To this end, we consider the case of a European arithmetic average reset call option with maturity \(T = 2\) years written on an underlying asset with initial value \(S = 50\). The risk-free continuously compounded interest rate is equal to \(r = 0.1\). In Tables 1–3, we report the option prices for different values of \(K_0\) and \(\sigma\). Increasing values of \(n\) are considered in order to show the convergence behavior of the option prices computed by the proposed model with respect to the benchmark Monte Carlo (MC) values. The values assumed by \(n\) are different in each table because \(n\) must be a multiple of \(N + 1\) to obtain monitoring windows characterized by an integer number of time steps. In this way, the reset dates coincide with lattice time layers thus avoiding biases occurring whenever the reset dates fall between consecutive steps of the lattice. In Table 1, where we consider options characterized by one reset date, we compute option prices with \(n\) ranging between 50 and 300 time steps. Clearly, if the number of monitoring windows increases, the option price calculation is more time consuming. This is the reason why in Tables 2 and 3 we consider \(n = 120\) steps in the case of \(N = 2\) reset dates, and \(n = 80\) time steps when \(N = 3\) reset dates, respectively.

In the last two rows of Tables 1–3, we report the prices supplied by the Kim et al. model (KCB) and by the MC simulations. KCB compute the call prices by considering a lattice based on 120 steps and different values of \(h\). Particularly, we report...
their best results which are obtained by considering $h = 0.005$ when the volatility value is $\sigma = 0.15$, and $h = 0.01$ when $\sigma = 0.3$ and $\sigma = 0.45$. The MC method is based on an Euler–time stepping scheme with one million trials (including 500 000 antithetic) and 120 time steps (the round brackets contain the MC standard error). It is worth noting that our model provides more accurate prices in comparison to the ones supplied by the KCB method. To give evidence of this aspect, in Table 4, we present a comparison in terms of the average relative deviation (ARD) from MC simulation of the results provided by our model (BIN) and by the KCB model. For each volatility level, we consider the BIN option values with $n = 120$ steps when $N = 1$ and $N = 2$, and $n = 80$ when $N = 3$, and the KCB prices as reported in Tables 1–3. In all the examined cases, BIN is characterized by an average relative deviation from the MC simulation that is smaller than the KCB one, despite we consider only 80 time steps when $N = 3$. In the last row of Table 4 (Overall Average), we also report the ARD for all the examined cases in Tables 1–3, without differentiating with respect to the volatility level. Such numerical comparison further assesses

Table 1
The prices of European arithmetic average reset call options with $N = 1$ reset time. For different values of $n$, we report option prices for three different volatility levels $\sigma = 0.15$, $\sigma = 0.3$, and $\sigma = 0.45$. For each volatility value, we consider three different strike prices $K = 45$, $K = 50$, and $K = 55$. The other initial parameters are: $S = 50$, $r = 0.1$, and $T = 2$. In the last two rows, we report the best option values provided by KCB and the benchmark values computed by MC simulations.

<table>
<thead>
<tr>
<th>$n$</th>
<th>$\sigma = 0.15$</th>
<th>$\sigma = 0.3$</th>
<th>$\sigma = 0.45$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$K_0 = 45$</td>
<td>$K_0 = 50$</td>
<td>$K_0 = 55$</td>
</tr>
<tr>
<td>50</td>
<td>13.482204</td>
<td>10.287595</td>
<td>8.465587</td>
</tr>
<tr>
<td>100</td>
<td>13.483757</td>
<td>10.293445</td>
<td>8.480482</td>
</tr>
<tr>
<td>120</td>
<td>13.484019</td>
<td>10.294433</td>
<td>8.482968</td>
</tr>
<tr>
<td>150</td>
<td>13.484284</td>
<td>10.295432</td>
<td>8.485480</td>
</tr>
<tr>
<td>200</td>
<td>13.484549</td>
<td>10.296433</td>
<td>8.487986</td>
</tr>
<tr>
<td>250</td>
<td>13.484708</td>
<td>10.297037</td>
<td>8.489496</td>
</tr>
<tr>
<td>300</td>
<td>13.484816</td>
<td>10.297441</td>
<td>8.490506</td>
</tr>
<tr>
<td>KCB</td>
<td>13.4875</td>
<td>10.3020</td>
<td>8.4859</td>
</tr>
<tr>
<td>MC</td>
<td>13.4860</td>
<td>10.2972</td>
<td>8.4924</td>
</tr>
</tbody>
</table>

Table 2
The prices of European arithmetic average reset call options with $N = 2$ reset times. For different values of $n$, we report option prices for three different volatility levels $\sigma = 0.15$, $\sigma = 0.3$, and $\sigma = 0.45$. For each volatility value, we consider three different strike prices $K = 45$, $K = 50$, and $K = 55$. The other initial parameters are: $S = 50$, $r = 0.1$, and $T = 2$. In the last two rows, we report the best option values provided by KCB and the benchmark values computed by MC simulations.

<table>
<thead>
<tr>
<th>$n$</th>
<th>$\sigma = 0.15$</th>
<th>$\sigma = 0.3$</th>
<th>$\sigma = 0.45$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$K_0 = 45$</td>
<td>$K_0 = 50$</td>
<td>$K_0 = 55$</td>
</tr>
<tr>
<td>30</td>
<td>13.556728</td>
<td>10.546596</td>
<td>9.110903</td>
</tr>
<tr>
<td>60</td>
<td>13.557920</td>
<td>10.555745</td>
<td>9.137285</td>
</tr>
<tr>
<td>90</td>
<td>13.558342</td>
<td>10.559149</td>
<td>9.146249</td>
</tr>
<tr>
<td>120</td>
<td>13.558567</td>
<td>10.560862</td>
<td>9.150798</td>
</tr>
<tr>
<td>KCB</td>
<td>13.5616</td>
<td>10.5716</td>
<td>9.1511</td>
</tr>
<tr>
<td>MC</td>
<td>13.5606</td>
<td>10.5633</td>
<td>9.1609</td>
</tr>
</tbody>
</table>

Table 3
The prices of European arithmetic average reset call options with $N = 3$ reset times. For different values of $n$, we report option prices for three different volatility levels $\sigma = 0.15$, $\sigma = 0.3$, and $\sigma = 0.45$. For each volatility value, we consider three different strike prices $K = 45$, $K = 50$, and $K = 55$. The other initial parameters are: $S = 50$, $r = 0.1$, and $T = 2$. In the last two rows, we report the best option values provided by KCB and the benchmark values computed by MC simulations.

<table>
<thead>
<tr>
<th>$n$</th>
<th>$\sigma = 0.15$</th>
<th>$\sigma = 0.3$</th>
<th>$\sigma = 0.45$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$K_0 = 45$</td>
<td>$K_0 = 50$</td>
<td>$K_0 = 55$</td>
</tr>
<tr>
<td>20</td>
<td>13.613912</td>
<td>10.728335</td>
<td>9.538806</td>
</tr>
<tr>
<td>40</td>
<td>13.612932</td>
<td>10.736857</td>
<td>9.570105</td>
</tr>
<tr>
<td>60</td>
<td>13.613020</td>
<td>10.741051</td>
<td>9.583266</td>
</tr>
<tr>
<td>80</td>
<td>13.613106</td>
<td>10.743485</td>
<td>9.589834</td>
</tr>
<tr>
<td>KCB</td>
<td>13.6162</td>
<td>10.7588</td>
<td>9.5936</td>
</tr>
<tr>
<td>MC</td>
<td>13.6153</td>
<td>10.7484</td>
<td>9.6069</td>
</tr>
</tbody>
</table>

Table 4
Overall Average of ARD for BIN and KCB methods in Tables 1–3.
The convergence of the KCB model is not assured when $h$ is chosen proportional to $\sqrt{\Delta t}$ as the authors suggest in order to keep the algorithm computationally feasible.

- As suggested in [11], the choice of $h$ proportional to $\Delta t$ allows the KCB algorithm to produce accurate prices in comparison to the MC simulations and really close to the ones provided by our model.

- The computational cost, measured in terms of number of strike prices used by the two models, shows that our algorithm outperforms the KCB model.

More in detail, in Tables 5–8, for different volatility levels, we report the prices of European arithmetic average reset call options with maturity $T = 1$ year and characterized by $N = 1$ reset date, written on an underlying asset with initial value $S = 50$, and $r = 0.1$. Different initial strike prices, $K_0$, are considered in order to take into account at-the-money and out-of-the-money options.\(^9\) We present option values computed by the KCB algorithm when $h$ is chosen equal to $\frac{1}{5} \sigma \sqrt{\Delta t}$.

\(^9\) We concentrate on at-the-money and out-of-the-money call options to show better the effects due to the reset feature. Indeed, in these cases the number of paths generating averages smaller than the initial strike price is greater than the ones obtained in the case of in-the-money options. The values

---

### Table 4

Average relative deviation from MC simulation of the KCB and BIN model. For each volatility value, $\sigma = 0.15$, $\sigma = 0.3$, and $\sigma = 0.45$, we compute the average relative deviation from MC simulation of both the KCB and BIN model. For the KCB model, we consider the option prices reported in Tables 1–3. For the BIN model, we consider the option prices reported in Tables 1–3 in correspondence to $n = 120$ steps when $N = 1$ and $N = 2$, and $n = 80$ when $N = 3$. In the last row, we report the average relative deviation for all the examined cases in Tables 1–3, without differentiating with respect to the volatility level.

<table>
<thead>
<tr>
<th>$\sigma$</th>
<th>KCB</th>
<th>BIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.15</td>
<td>0.06322</td>
<td>0.05948</td>
</tr>
<tr>
<td>0.3</td>
<td>0.11233</td>
<td>0.04478</td>
</tr>
<tr>
<td>0.45</td>
<td>0.05445</td>
<td>0.05382</td>
</tr>
<tr>
<td>Overall Average</td>
<td>0.07667</td>
<td>0.05269</td>
</tr>
</tbody>
</table>

### Table 5

Accuracy and computational cost comparisons in terms of number of strike prices between KCB and BIN. The table reports the option prices computed by the KCB algorithm with 120 time steps both when $h = \frac{1}{5} \sigma \sqrt{\Delta t}$ and $h = \frac{1}{5} \sigma \Delta t$, and the BIN model with $n$ ranging between 50 and 300. The initial option parameters are: $S = 50$, $r = 0.1$, $T = 1$, $K_0 = 55$, $\sigma = 0.5$, and $N = 1$.

<table>
<thead>
<tr>
<th>$n$</th>
<th>KCB with $h = \frac{1}{5} \sigma \sqrt{\Delta t}$</th>
<th>KCB with $h = \frac{1}{5} \sigma \Delta t$</th>
<th>BIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>50</td>
<td>11.540943</td>
<td>121520</td>
<td>11.511311</td>
</tr>
<tr>
<td>100</td>
<td>11.530562</td>
<td>152099</td>
<td>11.499147</td>
</tr>
<tr>
<td>120</td>
<td>11.532413</td>
<td>215791</td>
<td>11.511015</td>
</tr>
<tr>
<td>150</td>
<td>11.526886</td>
<td>405665</td>
<td>11.511015</td>
</tr>
<tr>
<td>200</td>
<td>11.546399</td>
<td>2023888</td>
<td>11.511730</td>
</tr>
<tr>
<td>250</td>
<td>11.546857</td>
<td>3378068</td>
<td>11.51183</td>
</tr>
</tbody>
</table>

### Table 6

Accuracy and computational cost comparisons in terms of number of strike prices between KCB and BIN. The table reports the option prices computed by the KCB algorithm with 120 time steps both when $h = \frac{1}{5} \sigma \sqrt{\Delta t}$ and $h = \frac{1}{5} \sigma \Delta t$, and the BIN model with $n$ ranging between 50 and 300. The initial option parameters are: $S = 50$, $r = 0.1$, $T = 1$, $K_0 = 50$, $\sigma = 0.7$, and $N = 1$.

<table>
<thead>
<tr>
<th>$n$</th>
<th>KCB with $h = \frac{1}{5} \sigma \sqrt{\Delta t}$</th>
<th>KCB with $h = \frac{1}{5} \sigma \Delta t$</th>
<th>BIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>50</td>
<td>16.534503</td>
<td>18354</td>
<td>16.502219</td>
</tr>
<tr>
<td>100</td>
<td>16.538439</td>
<td>129337</td>
<td>16.523181</td>
</tr>
<tr>
<td>120</td>
<td>16.562893</td>
<td>216198</td>
<td>16.526732</td>
</tr>
<tr>
<td>150</td>
<td>16.568416</td>
<td>405665</td>
<td>16.530287</td>
</tr>
<tr>
<td>200</td>
<td>16.573049</td>
<td>909883</td>
<td>16.533860</td>
</tr>
<tr>
<td>250</td>
<td>16.576251</td>
<td>1700027</td>
<td>16.536009</td>
</tr>
<tr>
<td>300</td>
<td>16.577754</td>
<td>2821379</td>
<td>16.537445</td>
</tr>
</tbody>
</table>

For each volatility level, $\sigma$, the choice of $h$ proportional to $\Delta t$ allows the KCB algorithm to produce accurate prices in comparison to the MC simulations and really close to the ones provided by our model.
as the authors suggest. Furthermore, we run the KCB model by setting \( h \propto \Delta t \), namely \( h = \frac{1}{3} \sigma \Delta t \). We also report the prices provided by the BIN model and compare the computational cost of the algorithms in terms of the number of strike prices used for price computations. The choice of measuring the computational cost through the number of strike prices is due to the fact that the two algorithms work in a similar fashion. Consequently, the number of strike prices to be considered makes the difference. The MC values, reported in the last row, are still computed on one million trials (including 500,000 antithetic) and 120 time steps (the round brackets contain the MC standard error).

In all the examined cases, it is evident that when \( h = \frac{1}{3} \sigma \sqrt{\Delta t} \), a reduced computational cost does not imply accurate option prices. Indeed, they do not show a convergent behavior with respect to the benchmark MC values. This evidence may be addressed to the fact that convergence of a forward shooting grid model is not assured when \( h \) is chosen in this way (cfr., Forsyth et al. [11]). Accurate prices are achieved if \( h = \frac{1}{2} \sigma \Delta t \) and this fact confirms the theoretical convergence of the algorithm when \( h \) is proportional to \( \Delta t \). On the other hand, it causes a huge increment in the computational cost of the KCB algorithm. Concerning the BIN model, it provides accurate prices with respect to the benchmark MC values. The option prices show a convergent behavior to the correct answer when we increase the number of lattice steps. These empirical evidences support the algorithm theoretical convergence. Furthermore, the BIN algorithm presents a smaller computational cost in terms of number of strike prices used for option valuations in comparison to the KCB model.

Finally, we propose a comparison between the delta (in Fig. 5) and the gamma (in Fig. 6) of a European arithmetic average reset call option and a European plain-vanilla one characterized by the same initial parameters. The delta of a European arithmetic average reset call option at each time step \( t \) of the CRR lattice \( u \) indicates an up step while \( d \) a down step) is computed by

\[
\delta(t) = \frac{c_d(t + \Delta t) - c_d(t + 2\Delta t)}{S_t(u - d)},
\]

while the gamma is given by

\[
\Gamma(t) = \frac{(c_{ud}(t + 2\Delta t) - c_{ud}(t + 2\Delta t)) / (S_t u^2 - S_t) - (c_{ud}(t + 2\Delta t) - c_{dd}(t + 2\Delta t)) / (S_t - S_t d^2)}{0.5S_t(u^2 - d^2)}.
\]

of \( \sigma \) considered here are different from the ones used in the other tables because the choice of \( h = \frac{1}{3} \sigma \sqrt{\Delta t} \), as suggested by KCB, despite \( h = \frac{1}{2} \sigma \Delta t \) (cfr., Forsyth et al. [11]) makes evident the non-convergent pattern of option prices when the volatility value \( \sigma \) is large.
The aim of such an analysis is to give an idea of the impact of the reset feature on a call option. In more detail, we consider a lattice based on 40 steps to price an arithmetic average reset call option with initial strike price $K_0 = 45$, risk-free interest rate $r = 0.1$, volatility $\sigma = 0.3$, maturity $T = 1$ years, and $N = 3$ reset dates. The initial asset price $S$ ranges in the interval $[20, 70]$. Fig. 5 shows that, when the option is deep out-of-the-money, the arithmetic average option delta is almost constant. To understand this effect, we observe that when the asset value at inception is very small with respect to the initial strike price, there is a high probability of reset. Hence, an increment in the underlying asset value has a double impact on the option value. It induces an increment in the option price (as in the plain-vanilla case) but, on the other side, it determines an increment in the strike price too because a higher initial underlying asset value will result in a higher realized average during each observation window. This second effect offsets the increment in the option value and makes the option value less sensitive to different asset price. Clearly, for options that are not deep out-of-the-money, the strike price is less likely to be reset and, as a consequence, the delta change is more similar to that of the corresponding plain-vanilla case. We also propose the graph in Fig. 6 showing a comparison between the gamma of an arithmetic average reset call option and the gamma of the corresponding plain-vanilla one. It is evident that the gamma of the reset option is approximatively constant when the option is deep out-of-the-money and initially smaller than the plain-vanilla one. Instead, when the initial asset value increases, the arithmetic average reset option gamma increases rapidly and assumes greater values than the gamma of the plain-vanilla option. This aspect confirms that the delta of the reset option increases faster than the delta of the plain-vanilla option when the initial asset price increases. In contrast, when the option is deep in-the-money, this effect vanished and the two gamma are really close to each other, as expected.

4. Conclusions

The proposed algorithm for pricing arithmetic average reset options with multiple reset dates is based on a CRR binomial lattice describing the evolution of the underlying asset price. In such a lattice framework, the main problem to look at is the large number of possible averages associated with each node. Indeed, in each monitoring window, the trajectories reaching a generic node produce different values for the arithmetic average and, when the number of time steps increases, the computational cost of the model grows exponentially. The main aspect of our model relies on the choice of the representative averages determining the strike price values at each reset date. In fact, instead of considering simulated averages as it happens in the Kim et al. [9] model, we propose to use vectors of effective representative averages computed on actual paths of the lattice selected following an easy scheme. This allows us to develop a binomial model which converges to the continuous time one and reduces the computational complexity of the pricing problem. Furthermore, the algorithm does
not depend on any exogenous parameter and, consequently, it allows to overcome the drawback affecting the Kim et al. [9] method, which converges to the continuous time model only when \( h \) (in their exponential functions for the averages and the strike prices) is chosen proportional to the lattice step length as suggested in [11].

The option price at inception is computed through a backward recursion scheme coupled with linear interpolation after that usual formulas provide the option prices associated with the nodes in correspondence to the last reset date. Finally, we propose a comparison between the prices supplied by the proposed model and the benchmark prices computed by the Monte Carlo method both for call and put cases. Furthermore, for the call case, we also provide a comparison with the option prices supplied by the Kim et al. model which evidences the greater accuracy of our algorithm.

The model presented here may be easily generalized for different types of reset options (e.g., options characterized by different types of path functions) or for American reset options. The algorithm is also flexible because it allows the use of other assumptions concerning the behavior of the underlying asset price, e.g., constant elasticity of variance specification, without much effort. Indeed, having chosen a procedure which makes recombining the discrete evolution of the underlying asset (cfr., Nelson and Ramaswamy [15], Costabile and Massabó [16], among others), the algorithm presented in this paper may be straightforwardly applied.

Acknowledgments

The authors thank the anonymous reviewers for their helpful comments, suggestions and remarks, which greatly improved the paper.

Appendix A. Convergence of the algorithm

The proposed model is based on a binomial discretization of the continuous time asset price process and uses a backward recursion scheme coupled with linear interpolation to compute the option price at inception. Consequently, the convergence analysis of the discrete model with the continuous time model has to be focused on the proof that both the truncation error (introduced by the binomial approximation) and the interpolation error tend to zero as the time step of the lattice \( \Delta t \to 0 \).

The continuous time model is based on a geometric Brownian motion for the asset value dynamics

\[
dS_t = rS_t dt + \sigma S_t dW_t,
\]

where \( r \) is the risk-free rate, \( \sigma \) is the volatility of the asset price and \( W_t \) is a Wiener process. We consider the case of a European arithmetic average reset call option\(^{10}\) characterized by \( t_1, \ldots, t_N \) reset dates with maturity \( T \), and initial strike price \( K_0 \). The option time to maturity results to be divided into \( N + 1 \) monitoring windows,\(^{11}\) \( (t_{l-1}, t_l), l = 1, \ldots, N + 1, \) which, without loss of generality, we suppose of equal length.

The arithmetic averaging feature makes such a reset option a path-dependent one and, in general, different trajectories for the asset price produce different values for the arithmetic average. Consequently, in addition to the underlying asset \( S \) and time \( t \), we have to introduce the average to measure its influence on the option price. Indeed, the European arithmetic reset call option presents a payoff at maturity \( T \) of the form \( \max(S_T - K(t_N), 0) \) where \( K(t_N) = \min(K_0, A_1, \ldots, A_N) \) and

\[
A_i = \frac{1}{t_i - t_{i-1}} \int_{t_{i-1}}^{t_i} S_i ds, \quad l = 1, \ldots, N,
\]

is the average computed on the asset prices falling into the \( l \)th monitoring window. It is worthwhile pointing out that the strike price \( K(t_l) = \min(K(t_{l-1}), A_i) \), \( l = 1, \ldots, N \) is piecewise constant within the monitoring window \( (t_{l-1}, t_l) \) and it may jump at \( t_{l+1} \).

During the monitoring window \( (t_{l-1}, t_l) \), we have to take into account the average variations because it evolves continuously \( \forall t \in (t_{l-1}, t_l) \) according to

\[
dA_i = \frac{1}{t_l - t_{l-1}} (S_t - A_i) dt,
\]

but we can simply treat the strike price as a dummy variable since it is fixed at the value assumed at the beginning of each monitoring window. On the contrary, the strike price variations have to be taken into account in correspondence to each reset date \( t_i \), \( l = 1, \ldots, N \), and this is done by imposing a jump condition on the option price as showed hereafter. Finally, the option value \( V \) at time \( t \in (t_{l-1}, t_l), l = 1, \ldots, N \) is a function of three independent variables represented by time \( t \), current asset value \( S_t \), and average \( A_i \), i.e., \( V = V(S_t, A_i, t) \), and we are now in a position to write down the Black–Scholes PDE for the European reset option into each monitoring window.

\(^{10}\) The extension to all the other cases is straightforward.

\(^{11}\) Clearly, \( t_{N+1} = T \).
By applying Ito’s lemma to \( V(S_t, A_t, t) \) and setting up the usual risk-free portfolio consisting of one option and a short position with a number \( \frac{\partial V}{\partial S} \) of the underlying asset, we obtain (cfr., Barraquand and Pudet [17], Forsyth et al. [11], and Jiang and Dai [18]) the PDE that is governing the option value into each monitoring window \( (t_{i-1}, t_i), i = 1, \ldots, N \):

\[
\frac{\partial V}{\partial t} + \frac{1}{2} \sigma^2 S^2 \frac{\partial^2 V}{\partial S^2} + rS \frac{\partial V}{\partial S} + \frac{1}{t - t_{i-1}} (S_t - A_t) \frac{\partial V}{\partial A} - rV = 0.
\]

Such a PDE is solved with a final condition given by \( V(S_T, A_T, T) = \max[S_T - K(t_0), 0] \) and, to capture the effects of the option reset feature, we have to consider at each reset date the update of the strike price. As explained above, it depends on the average value and is modeled by the jump condition \( V(S_{i-1}^-, A_{i-1}^-, t_i^-) = V(S_{i-1}^+, \min[K(t_{i-1}), A_{i-1}], t_i^-), i = 1, \ldots, N \). To summarize, in each monitoring window \( (t_{i-1}, t_i), i = 1, \ldots, N, \) we have to solve the following continuous time problem

\[
\begin{cases}
\frac{\partial V}{\partial t} + \frac{1}{2} \sigma^2 S^2 \frac{\partial^2 V}{\partial S^2} + rS \frac{\partial V}{\partial S} + \frac{1}{t - t_{i-1}} (S_t - A_t) \frac{\partial V}{\partial A} - rV = 0, & t_{i-1} < t < t_i, \\
V(S_{i-1}^+, A_{i-1}^-, t_i^-) = V(S_{i-1}^+, \min[K(t_{i-1}), A_{i-1}], t_i^-). &
\end{cases}
\]

which, on the last window \( [t_N, T] \), becomes

\[
\begin{cases}
\frac{\partial V}{\partial t} + \frac{1}{2} \sigma^2 S^2 \frac{\partial^2 V}{\partial S^2} + rS \frac{\partial V}{\partial S} - rV = 0, & t_N < t < T, \\
V(S_t, A_t, T) = \max[S_t - K(t_0), 0]. &
\end{cases}
\]

because after the last reset date \( t_N \), the arithmetic average reset option becomes a standard one with strike price \( K(t_0) = \min(K, A_1, A_2, \ldots, A_N) \), time to maturity \( T - t_0 \), and current underlying asset value \( S_{t_N} \). Hence, there is no dependence on the average.

The lattice model presented in Section 2 is obtained by chopping the option lifetime \( T \) into \( n \) equal subintervals of length \( \Delta t = \frac{T}{n} \), so that we have an equal number of observations \( \delta = \frac{n}{N+1} \) in each window. Recall that the option price over the last monitoring window \( [t_N, T] \) is computed by the standard Black–Scholes [13] formula which is the solution of the PDE in (7). A convergence analysis must be addressed to the first \( N \) monitoring windows where a backward recursion scheme coupled with linear interpolation is used.

Concerning the truncation error which affects the approximate solution of the valuation problem when the backward scheme is used in (6), we may follow the lines suggested in [18] relatively to the convergence of CRR models proposed for pricing path-dependent options and apply their findings to our algorithm in each monitoring window. Indeed, even if the binomial lattice method described in Section 2 (cfr., Eq. (4)) depends on four indexes, in each monitoring window \( (t_{i-1}, t_i) \), the dependence on the strike is fictitious since it cannot vary. Hence, \( c(i, j; a, k) \) (cfr., Eq. (4)) may be replaced by the analogous (cfr., Jiang and Dai [18]) option price

\[
V(i, j; a) = e^{-r\Delta t} \left[ pV(i + 1, j + 1; a_0) + qV(i + 1, j; a_0) \right],
\]

where it is supposed that, from the \( r \)th average value at node \( (i, j) \), an up step leads to average value \( A(i + 1, j + 1; a_0) \) at node \( (i + 1, j + 1) \) while a down step leads to average value \( A(i + 1, j + 1; a_0) \) at node \( (i + 1, j) \).

According to the Jiang–Dai analysis, the binomial lattice method defined by (8) is consistent with the PDE (6) (cfr., Jiang and Dai [18], page 1097 Theorem 3.1) and, moreover, by neglecting terms of high order of \( \Delta t \), is equivalent to an explicit difference scheme related to (6) (cfr., Jiang and Dai [18], Section 4). At this point, we can invoke the result obtained in [18] concerning the truncation error introduced by such a discretization which is of order \( O((\Delta t)^2) \).

Working into each monitoring window \( (t_{i-1}, t_i) \), where the strike price is constant, being it defined at time \( t_{i-1} \), an interpolation error may occur because our algorithm uses linear interpolation to compute the option price in correspondence to the averages not appearing in the vector. It is also worth mentioning that, at each reset epoch \( t_i, i = 1, \ldots, N \), no interpolation is required, as explained in Section 2.3 (cfr., Case 3). Consequently, the interpolation error has to be quantified only for the option values computed in correspondence to those nodes \( (i, j) \) falling into each monitoring window \( (t_{i-1}, t_i) \). We recall that at time \( t_i - \Delta t \) (cfr., Case 2), interpolation occurs only when the average value resets the strike price.

Quantity \( V(i + 1, j; a_0) \) is the option value when the asset is \( S(i + 1, j + 1) \) and the average value is

\[
A(i + 1, j + 1; a_0) = \frac{(i - (i - 1)\delta)A(i, j; a) + S(i + 1, j + 1)}{i - (i - 1)\delta + 1},
\]

while \( V(i + 1, j; a_0) \) is the option value when the asset is \( S(i + 1, j) \) and the average value is

\[
A(i + 1, j; a_0) = \frac{(i - (i - 1)\delta)A(i, j; a) + S(i + 1, j)}{i - (i - 1)\delta + 1}.
\]

Since the algorithm considers only a selected subset of effective averages at each node of the lattice, \( A(i + 1, j + 1; a_0) \) could appear in the vector of the representative averages associated with node \( (i + 1, j + 1) \) but this is not ever assured. Consequently, to compute the option value associated with \( A(i + 1, j + 1; a_0) \), sometimes we linearly interpolate the option
values in correspondence to the two known averages \( A(i + 1, j + 1; a_1) \) and \( A(i + 1, j + 1; a_2) \) at node \((i + 1, j + 1; a_u)\), such that \( A(i + 1, j + 1; a_1) < A(i + 1, j + 1; a_u) \leq A(i + 1, j + 1; a_2)\).

The key point now is the valuation of the error due to linear interpolation. To do this, we follow the lines suggested in [11] and work under their assumptions. Through the Taylor expansion series, \( V(i + 1, j + 1; a_u) \) may be rewritten in terms of the linear interpolation between the option prices associated with the known values \( A(i + 1, j + 1; a_1) \) and \( A(i + 1, j + 1; a_2) \):

\[
V(i + 1, j + 1; a_u) = V(i + 1, j + 1; a_1) + \omega(i + 1, j + 1; a_u) \left( V(i + 1, j + 1; a_2) - V(i + 1, j + 1; a_1) \right) + \beta(i + 1, j + 1; a_u),
\]

where

\[
\omega(i + 1, j + 1; a_u) = \frac{A(i + 1, j + 1; a_u) - A(i + 1, j + 1; a_1)}{A(i + 1, j + 1; a_2) - A(i + 1, j + 1; a_1)},
\]

and

\[
\beta(i + 1, j + 1; a_u) = \frac{1}{2} \left( (a_u + 1) - A(i + 1, j + 1; a_2) - A(i + 1, j + 1; a_1) \right) \frac{\partial^2 V(i + 1, j + 1; a_1)}{\partial A^2}.
\]

This similarly occurs for \( V(i + 1, j; a_u) \). Consequently, Eq. (8) may be written as

\[
V(i, j; a) = e^{-r\Delta t} \left[ p \left( V(i + 1, j + 1; a_1) + \omega(i + 1, j + 1; a_u) \left( V(i + 1, j + 1; a_2) - V(i + 1, j + 1; a_1) \right) \right) + q \left( V(i + 1, j; a_1) + \omega(i + 1, j; a_u) \left( V(i + 1, j; a_2) - V(i + 1, j; a_1) \right) \right) \right] + \text{interpolation error} + O \left( (\Delta t)^2 \right).
\]

Let \( E(i, j; a) \) be the difference between the exact solution \( V \) of the PDE in (6) and the approximate solution provided by our algorithm. An equation for the propagation of the interpolation and truncation error is given by

\[
E(i, j; a) = e^{-r\Delta t} \left[ p \left( E(i + 1, j + 1; a_1) + \omega(i + 1, j + 1; a_u) \left( E(i + 1, j + 1; a_2) - E(i + 1, j + 1; a_1) \right) \right) + q \left( E(i + 1, j; a_1) + \omega(i + 1, j; a_u) \left( E(i + 1, j; a_2) - E(i + 1, j; a_1) \right) \right) \right] + \text{interpolation error} + O \left( (\Delta t)^2 \right).
\]

From the recursion in (9), we can bound the cumulative effect of the interpolation and truncation error on the solution at inception. To do this, we may assume that

\[
\left| \frac{\partial^2 V(i, j; a)}{\partial A^2} \right| \leq M_{i,j}, \quad \forall i, j,
\]

where \( M_{i,j} \) is a constant independent on the step size \( \Delta t \) (cfr., Forsyth et al. [11], page 284 Eq. 4.13). As a consequence, the interpolation error can be bounded by

\[
\max \{ \beta(i + 1, j + 1; a_u) \} \leq \left[ M_{i+1,j+1} \right] \left[ A(i + 1, j + 1; a_2) - A(i + 1, j + 1; a_1) \right]^2 \leq \left[ M_{i+1,j+1} \right] \left[ A(i + 1, j; a_2) - A(i + 1, j; a_1) \right]^2.
\]

and, if we suppose \( i + 1 \) belongs to the \( l \)th window, i.e., \( (i + 1) \in ((l - 1)\delta + 1, l\delta) \), it may be rewritten as (cfr., Eqs. (2) and (3))

\[
\max \{ \beta(i + 1, j + 1; a_u) \} \leq M \left[ \frac{S_{\text{max}(x_{\text{min}}, \cdot)}, \beta_{\text{max}}(\cdot)}{i - (i - 1)\delta + 1} \right]^2 \left[ 1 - d^2 \right]^2.
\]

In (10), \( M \) assumes the value \( M_{i+1,j+1} \) or \( M_{i+1,j+1} \) according to the maximum in the right hand side of (10) and \( S_{\text{max}(x_{\text{min}}, \cdot)}, \beta_{\text{max}}(\cdot) \) is an asset value which, as known, does not explode on a finite horizon. If we define the maximum error at the \( l \)th step by

\[
\| E(l) \| = \max_{j,a} \| E(i, j; a) \|,
\]

given that the interpolation coefficients \( \omega(\cdot, \cdot, \cdot) \) and the probabilities \( p \) and \( q \) are all in the range [0,1], it follows from (9) that

\[
\| E(l) \| \leq e^{-r\Delta t} \left( \| E(i + 1) \| + \frac{M[S(i - 1, i - 1)]^2 \left[ 1 - d^2 \right]^2}{i - (i - 1)\delta + 1} \right) + O \left( (\Delta t)^2 \right).
\]
In (12), \( S(i−1, i−1) \) is the highest asset value at the \( i \)th time step with \( i < n \) (\( n \) number of time steps) which can be considered in the interpolation because at each time step no interpolation is required for the highest node (i.e., at the \( i \)th step no interpolation is required on node \((i, i))\).

Relation (12) means that during time step \( i + 1 \rightarrow i \) the error does not become amplified but propagates with non-increasing size. However, the cumulative error grows linearly due to the fact that a new interpolation error occurs at each step. In the worst case, given \( \delta \) steps in each monitoring window and \( N \) windows where we use backward recursion and linear interpolation, we may conclude that the cumulative effect due to the interpolation error is bounded by

\[
N \sum_{i=1}^{\lfloor \delta \rfloor-1} M[S(i−1, i−1)]^2 \left[ 1 - d^2 \right]^2 = N \left[ M \left[ 1 - e^{-2\sigma \sqrt{\Delta t}} \right]^2 \right] \sum_{i=1}^{\lfloor \delta \rfloor-1} \left( \frac{1}{i - (l - 1)\delta + 1} \right)^2.
\]

\[
\leq N \left[ M_1 \left[ 1 - e^{-2\sigma \sqrt{\Delta t}} \right]^2 \right] \sum_{i=1}^{\lfloor \delta \rfloor-1} \left( \frac{1}{i - (l - 1)\delta + 1} \right)^2 = N \left[ M_1 \left[ 1 - e^{-2\sigma \sqrt{\Delta t}} \right]^2 \right] \sum_{k=1}^{\delta} \frac{1}{k^2}
\]

\[
\leq 2NM_1 \left[ 1 - e^{-2\sigma \sqrt{\Delta t}} \right]^2 \simeq 8NM_1\sigma^2 \Delta t = O(\Delta t).
\]

In (13), we use the convergence properties of the harmonic series, \( M_1 = MS^2e^{2\Delta t} \sqrt{\Delta t} \) and we suppose that the asset value does not explode to infinity on a finite horizon to keep the problem financially consistent. Concerning the truncation error which at each step is \( O \left( (\Delta t)^{\frac{3}{2}} \right) \), the cumulative effect after \( N\delta \) steps is \( O(\sqrt{\Delta t}) \). Finally, we may conclude that the worst case error bound is

\[
\| E(0) \| \leq O(\sqrt{\Delta t}) + O(\Delta t),
\]

which guarantees the convergence of the numerical solution provided by our algorithm to the continuous time one as \( \Delta t \rightarrow 0 \). \( \square \)

### Appendix B. Proof of the Proposition

Our first task is to remark that the number of paths detected to associate a vector of representative averages with a generic node \((i, j)\) belonging to the \((l + 1)\)th monitoring window \( [t_i, t_{i+1}) \) \((l = 1, \ldots, N - 1)\) is in a one-to-one correspondence with the number of nodes lain between the lowest, \( \tau_{\min}(i, j) \), and the highest, \( \tau_{\max}(i, j) \), path including the nodes belonging to \( \tau_{\max}(i, j) \) and excluding the ones belonging to \( \tau_{\min}(i, j) \) and to the \( l \)th reset date. To clarify this aspect, suppose to consider an arithmetic average reset option with maturity \( T = 3 \) years characterized by two reset dates. In Fig. 7, we illustrate the first 8 steps of a binomial lattice based on \( n = 12 \) time steps discretizing the option time to maturity. The first reset date happens after \( \delta = 4 \) steps while the second one after \( 2\delta = 8 \) time steps. Suppose to consider node \((i, j) = (7, 5)\) belonging to the second monitoring window for which the lowest path, \( \tau_{\min}(7, 5) \), and the highest one, \( \tau_{\max}(7, 5) \), are depicted by thick lines. The nodes to be considered (i.e., the ones being in a one-to-one correspondence with the number of representative paths obtained by the iterative procedure presented in Section 2) are evidenced by big black circles and are enclosed in the “quadrilateral” ABCD. Such a “quadrilateral” would be a “triangle” when, for example, we consider nodes \((7, 4)\) or \((7, 3)\).

In order to count these nodes, we observe that, starting from the \( l \)th reset date, the number of steps needed to reach node \((i, j)\) belonging to the \((l + 1)\)th monitoring window is \( i - l \). Then, we follow the procedure outlined below:

**Step 1:** Starting from node \((i, j)\) (i.e., from the vertex \( B \) of the lattice), we consider the two adjacent sides \( BA \) and \( BC \). Between them, we fix the smallest one, i.e., the side with length (measured in terms of number of steps):

\[
\min (i - l\delta, \min(j, i - j)),
\]

which in Fig. 7 is \( BC \).

**Step 2:** We extend the other side, \( BA \), by a number of steps equal to

\[
i - l\delta - \min (i - l\delta, \min(j, i - j)),
\]

to obtain the side \( BE \) in Fig. 7 with length

\[
2i - 2l\delta - \min (i - l\delta, \min(j, i - j)).
\]

In such a way, we build up the “quadrilateral” \( BEFC \) (having two opposite sides \( BE \) and \( FC \)) with length given by (15) and build up the other two opposite ones \((BC \text{ and } EF)\) with length given by (14) which is divided into two equal parts by the \( l \)th reset date. One of these parts is exactly the “quadrilateral” ABCD. The total number of nodes belonging to \( BEFC \), excluding the ones lain on the lowest sides \( BE \) and \( EF \) is easily given by

\[
\min (i - l\delta, \min(j, i - j)) (2i - 2l\delta - \min (i - l\delta, \min(j, i - j))).
\]
In order to obtain the number of nodes belonging to $ABCD$, we first observe that the nodes lain on the $l$th reset date must not be considered and, consequently, the number of nodes belonging to $BEFC$ is given by

$$\min (i - l \delta, \min(j, i - j)) (2i - 2l \delta - \min (i - l \delta, \min(j, i - j)) - 1),$$

because the number of nodes lain on the $l$th reset date is given by $\min (i - l \delta, \min(j, i - j))$. Finally, the number of nodes belonging to $ABCD$ is

$$\frac{1}{2} \min (i - l \delta, \min(j, i - j)) (2i - 2l \delta - \min (i - l \delta, \min(j, i - j)) - 1),$$

and, by adding the nodes lain on the highest trajectory, the total number of paths reaching node $(i, j)$ is given by

$$\eta(i, j) = 1 + \frac{1}{2} \min (i - l \delta, \min(j, i - j)) (2i - 2l \delta - \min (i - l \delta, \min(j, i - j)) - 1).$$

□

References