European Journal of Operational Research 236 (2014) 369–381

Contents lists available at ScienceDirect

European Journal of Operational Research

journal homepage: www.elsevier.com/locate/ejor

Innovative Applications of O.R.

Pricing exotic derivatives exploiting structure

Debora Sesana [a], Daniele Marazzina [b,*], Gianluca Fusai [a,c]

[a] Dipartimento S.E.I., Università del Piemonte Orientale A. Avogadro, via Perrone 18, I-28100 Novara, Italy
[b] Dipartimento di Matematica, Politecnico di Milano, Piazza Leonardo da Vinci 32, I-20133 Milano, Italy
[c] Cass Business School, City University London, 106 Bunhill Row, London EC1Y 8TZ, UK

ARTICLE INFO

Article history:
Received 10 January 2013
Accepted 5 December 2013
Available online 15 December 2013

Keywords:
CEV process
Discrete monitoring
Exotic derivatives
Matrix Factorization
Numerical quadrature
Option pricing

ABSTRACT

In this paper we introduce a new fast and accurate numerical method for pricing exotic derivatives when discrete monitoring occurs, and the underlying evolves according to a Markov one-dimensional stochastic processes. The approach exploits the structure of the matrix arising from the numerical quadrature of the pricing backward formulas to devise a convenient factorization that helps greatly in the speed-up of the recursion. The algorithm is general and is examined in detail with reference to the CEV (Constant Elasticity of Variance) process for pricing different exotic derivatives, such as Asian, barrier, Bermudan, lookback and step options for which up to date no efficient procedures are available. Extensive numerical experiments confirm the theoretical results. The MATLAB code used to perform the computation is available online at http://www1.mate.polimi.it/~marazzina/BP.htm.

1. Introduction

Over the years exotic options, such as Asian, barrier, and lookback contracts, have become more and more popular in equity markets and raised the attention in the academic research. Most of the articles in this literature price these contracts assuming a continuous monitoring, i.e., the payoff is triggered by events occurring continuously before expiry. Under this assumption, the option can often be priced in closed form. However, many real exotic contracts specify discrete monitoring instants, such as daily or weekly, see for example (Becker & Wystup, 2009) for a detailed description of the real market functioning of foreign exchange discrete barrier options. On the other hand, analytical formulae are available only when the underlying evolves according to a geometric Brownian motion (GBM) process and barrier and lookback options are considered, see (Fusai, Abrahams, & Sgarra, 2006), whilst other path-dependent options, such as Asian, do not admit closed form pricing formula. In addition, as at first noticed in Broadie, Glasserman, and Kou (1997), there can be substantial differences between discrete and continuous monitoring prices, even under very frequent monitoring. In fact, given the complexity of financial models and option contracts used in practice, we need a numerical method able to calibrate plain vanilla options, and at the same time price and hedge in a fast and accurate way exotic derivatives, such as the ones considered in this paper. As discussed in Bouzoubaa and Osseiran (2010), errors in computing sensitivities replacing the discrete feature with the continuous one (or a different discrete one) can cause large damages in the dynamic hedging strategy. As an example, in Fig. 1 we show how the Deltas of exotic contracts (barrier, Bermudan, Asian and lookback) vary according to different discrete monitoring features: with the exception of the Asian call, where the Deltas are very similar, for all the other contracts different monitoring features result in significant differences in Deltas, and thus in very different replicating portfolios. This problem is illustrated in detail in Castagna (2009): the author considers a market maker who is short a reverse knock-out option with a short time to maturity and the underlying spot price is around the barrier level and between two fixings dates, it is not possible to take the unequivocal decision whether to completely unwind the Delta-hedge. This uncertainty generates the so-called slippage cost that can be quite significant, and motivates the search for a fast and accurate algorithm for pricing the above mentioned contracts.

The discrete monitoring feature here considered is also relevant in real option application, for which the management decision making to continue or to abandon the project is determined on the basis of accounting reports published periodically. An interesting application using additive processes is given by Alexander, Mengija, and Stent (2012).

The discretely monitoring pricing procedure for exotic options is based on the standard backward recursion: at each monitoring date the option price is updated by taking the expectation of the derivative price at the previous date and checking if the underlying satisfies the monitoring condition. For example, in a down-and-out barrier option we set to zero the option value if the corresponding

* Corresponding author. Tel.: +39 02 2399 4630.
E-mail addresses: debora.sesana@eco.unipmn.it (D. Sesana), daniele.marazzina@polimi.it (D. Marazzina), gianluca.fusai@eco.unipmn.it (G. Fusai).

0377-2217/$ - see front matter © 2013 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.ejor.2013.12.009

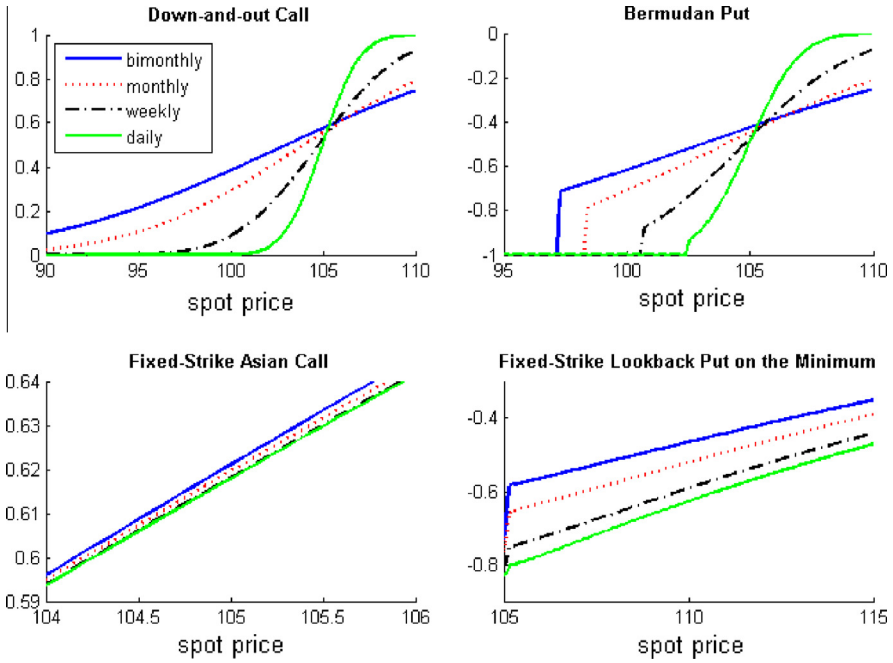

Fig. 1. Deltas of different exotic contracts. The maturity of all the contracts is $T = 1$ year, and the strike price is 105. For the down-and-out option, the barrier is equal to 90. The underlying asset is described according to a square-root process of parameter $\sigma = 2.5$, and the risk-free interest rate is equal to 0.1.

underlying price falls below the barrier. A possible numerical implementation consists of computing the nested (iterated) expectation via a recursive numerical quadrature (integration), as shown for example in Andricopoulos, Widdicks, Duck, and Newton (2003), Fusai and Recchioni (2007). This approach allows us to move directly from one date to the following one without any intermediate time discretization. In contrast lattice and finite difference or element methods are affected by errors due to the time discretization between monitoring dates. A considerable speed-up in the recursion can be obtained when the underlying asset evolves according to an exponential Lévy model. In this case the iterated expectation is a convolution between the transition density and the contract price at the previous monitoring date. The convolution can be computed efficiently via Fourier transform and exploiting the Fast Fourier Transform (FFT) algorithm: at each monitoring date we apply a Fourier transform-convolution-Fourier inversion, see for example (Fusai & Recchioni, 2007; Fusai, Marazzina, Marena, & Ng, 2012; Lord, Fang, Bervoets, & Oosterlee, 2008) and references therein. In this case, the computational cost is linear in the number of monitoring dates and of order $m \log(m)$ with respect to the number of discretization points m of the quadrature scheme. However the convolution structure of the transition density is due to the independent and identically distributed (i.i.d.) assumption on log-price increments. Unfortunately, if this assumption does not hold, the numerical integration becomes computational intensive with a cost proportional to m^2.

Therefore the main contribution of the present paper is to devise an efficient algorithm to price discretely monitored options previously described exploiting the structure of the probability transition density of the underlying asset and of its sampling matrix when log-price increments are not i.i.d. and the FFT algorithm cannot be used. More precisely, we introduce the concept of cluster of eigenvalues of a sequence of matrices arising from the numerical quadrature of the backward recursion as we increase the number of nodes m. We formally prove (Theorem 3) that the number of significant eigenvalues (i.e., larger than a fixed tolerance ϵ) is approaching a constant r_ϵ independent of $m, r_\epsilon \ll m$, as we take larger values of m. This result can be exploited to factorize the iteration matrix, giving a computational cost of $O(k_\epsilon m)$ operations, $k_\epsilon \approx r_\epsilon$, for the matrix–vector multiplication, instead of the standard $O(m^2)$. Given that the cost of the factorization is nearly independent on the number of monitoring dates, the advantage of our approach will be the greatest, greater the number of monitoring dates. The algorithm is general in the sense that it can be applied to stochastic processes for which the transition density is known in closed form. However, to make concrete our analysis, we examine it in detail with reference to the Constant Elasticity of Variance (CEV) model, introduced by Cox (1996), Cox and Ross (1976). This dynamics is interesting allowing for very different transition densities and implied volatility shapes. Very few option pricing models yield fully analytical results, and most require numerical evaluations. The CEV model is not an exception. Numerical methods for pricing derivatives under the CEV process are presented, for example, in Boyle and Tian (1999), Boyle, Tian, and Imai (1999), Costabile (2006), Peng (2006). All the mentioned articles refer to the continuous monitoring case and do not admit a simple implementation for exotic derivatives, whilst our approach can easily deal with a variety of path-dependent options.

Our algorithm is suitable to fast and accurate computation of prices and sensitivities of discretely monitored path-dependent options and therefore makes a real novel contribution for pricing real life contracts, see (Becker & Wystup, 2009), and risk-management decision making, (Castagna, 2009). For this reason our research is relevant to the methodology of operational research and to the practice of decision making. From the methodological point of view, we would like also to stress that the proposed factorization has potential and immediate applications to the study of properties of discrete time Markov Chains as well, a very important topic in operational research for modeling queuing sequences and many other practical systems, see (Norris, 1997).

To confirm the efficiency of the proposed methodology, theoretical results are proved in appendix, and extensive numerical experiments are conducted to compare the accuracy and the computational cost of our algorithm with respect to a standard backward recursive quadrature and to Monte Carlo simulation. In

particular, numerical experiments confirm that the greatest benefits are achieved for a large number of monitoring dates.

The structure of the paper is as follows. First of all, Section 2 introduces the general setup to price exotic derivatives with the discrete monitoring feature. Section 3 deals with the quadrature approach to solve the recursive pricing formulas, and we introduce the factorization idea, based on the structure of the pricing matrix. Finally, in Section 4 we validate the pricing procedure with numerical results assuming a CEV dynamics for the underlying. Accuracy and computational cost of our pricing algorithm are compared with the above mentioned benchmarks, i.e., the classical quadrature approach and Monte Carlo simulation. Theoretical results supporting the proposed methodology are provided in appendix.

2. The option pricing problem

Let us consider a derivative contract with a payoff $\phi(\cdot)$ at maturity $T = N\Delta$, where N is the number of Δ-equally spaced monitoring dates, and let S be the underlying asset price. The standard backward procedure computes the derivative price $V(S,n)$ at time $n\Delta$ through the following recursion (eventually with a modification to deal with the early exercise feature):

$$V(S,n) = e^{-r\Delta}\int_{\Omega} p(S,\xi;\Delta)V(\xi,n+1)d\xi, \quad n = N-1,\dots,0, \tag{1}$$

where r is the risk-free rate, and $p(S,\xi;\Delta)$ is the transition density from S at time t to ξ at time $t+\Delta$. Ω refers to the integration domain and can vary depending on the trigger event. The above recursion starts with the payoff condition at maturity $V(S,N) = \phi(S)$. We are interested in computing $V(S_0,0), S_0$ being the current spot price.

In the following subsections, we show how the above framework fits different exotic contracts.

2.1. Barrier options

If we deal with barrier options, the pricing recursion (1) starts from the payoff function $\phi(S) := (\varphi(S-E))^+$, where E is the strike price and φ is a binary variables taking value 1 for calls, and -1 for puts. If we denote with $L(U)$ the lower (upper) barrier, the domain Ω is $(L,+\infty)$ for down-and-out, (L,U) for knock-and-out, and $(0,U)$ for up-and-out barrier options.

For numerical purposes, the integration interval in (1) is truncated to $(L,\mathcal{U})$ – for down-and-out options – or $(\mathcal{L},U)$ – for up-and-out options – with $\mathcal{L} < S_0$ ($\mathcal{U} > S_0$). The truncation is chosen such that the probability of moving from S_0 to $\mathcal{L}$ ($\mathcal{U}$) is less than a preassigned tolerance.

2.2. Bermudan options

A Bermudan option gives the holder the right to early exercise at each monitoring date. This option is worth more than the corresponding European version, but less than the American counterparty, for which the exercise occurs continuously. To take into account the early exercise possibility we modify (1) into:

$$V(S,n) = \max\left\{e^{-r\Delta}\int_{\Omega} p(S,\xi;\Delta)V(\xi,n+1)d\xi,\ \phi(S)\right\},$$
$$n = N-1,\dots,0, \tag{2}$$

with $\Omega = (0,+\infty)$ for standard Bermudan options. If we have Bermudan contracts with a barrier trigger, then $\Omega = (L,+\infty)$, $\Omega = (0,U)$ or $\Omega = (L,U)$. Payoff function and domain truncation are as in Section 2.1.

2.3. Lookback options

The maturity settlement of lookback options is based on the minimum or the maximum value of the underlying asset as registered during the lifetime of the option. At maturity, the holder can "look-back" and select the most favorable figure of the underlying as occurring at the monitoring dates. If we let $S(n\Delta)$ to be the asset price at the nth monitoring date, we can define the discretely observed minimum price as

$$J_n := \min\{S(0),\dots,S(n\Delta)\}.$$

The payoff function of a fixed-strike lookback on the minimum is given by $(E-J_N)^+$. The lookback option price at time $n\Delta$ depends on the underlying asset price S, and on the up-to-date minimum $J_n = J$ and we denote it by $V(S,J,n)$. Clearly it must be $J \leqslant S$. Similar considerations hold for payoffs written on the maximum.

Respect to the GBM dynamics, where a change of numeraire argument reduces the number of state variables, under a more general process specification we must keep track of both state variables, underlying price and running minimum. Given that $J_{n+1} = \min\{J_n, S((n+1)\Delta)\}$, the backward recursion becomes $V(S,J,N) = (\varphi(J-E))^+$ and for $n = N-1,\dots,0$

$$V(S,J,n) = e^{-r\Delta}\int_0^{+\infty} p(S,\xi;\Delta)V(\xi,\min\{J,\xi\},n+1)d\xi, \tag{3}$$

S being greater or equal to J. Here $\min\{J,\xi\}$ is the minimum value of the underlying asset at the $(n+1)$th monitoring date given that $J_n = J$ and $S((n+1)\Delta) = \xi$. The initial option price is then given by $V(S_0,S_0,0)$.

2.4. Asian options

Asian options are a very popular type of exotic derivative. Such as for lookback options, their pricing requires the introduction of a new state variable, i.e., the (arithmetic) average up to time $n\Delta$

$$A_n = \frac{1}{n+1}\sum_{i=0}^{n} S(n\Delta).$$

The arithmetic average follows the updating rule

$$A_{n+1} = \frac{n+1}{n+2}A_n + \frac{1}{n+2}S((n+1)\Delta),$$

so that the price of the arithmetic fixed-strike Asian option satisfies the following backward recursion:

$$V(S,A,n) = e^{-r\Delta}\int_0^{+\infty} p(S,\xi;\Delta)V\left(\xi,\frac{n+1}{n+2}A+\frac{1}{n+2}\xi,n+1\right)d\xi, \tag{4}$$

for $n = N-1,\dots,0$, with $V(S,A,N) = (\varphi(A-E))^+$.

2.5. Step options

Step options are similar to barrier options, but the knock-and-out feature operates only gradually. To this aim we define the occupation time I_n of the subset $\mathcal{I}, \mathcal{I} \subset \mathbb{R}^+$,

$$I_n = \sum_{i=1}^{n} \mathbf{1}_{\{S(i\Delta)\in\mathcal{I}\}},$$

where $\mathbf{1}_{\{S(i\Delta)\in\mathcal{I}\}}$ is the indicator function, i.e., it is equal to 1 if $S(i\Delta) \in \mathcal{I}$, 0 otherwise. Notice that I_n measures the time spent by the underlying asset in the set $\mathcal{I}$ up to time $n\Delta$. I_n takes values in $\{0,1,2,\dots,n\}$ and satisfies the updating rule $I_{n+1} = I_n + \mathbf{1}_{\{S((n+1)\Delta)\in\mathcal{I}\}}$.

Given $S(N\Delta) = S$ and $I_N = I$, the payoff of a step option with principal amortization below the barrier is

$$V(S,I,N) = \left(1 - \frac{\rho}{N} I\right)^{+} (\varphi(S-E))^{+},$$

where ρ is the knock-out killing rate. The introduction of the knock-out range has the advantage of regularize the barrier option by making the price and the delta continuous at the barrier, see for example (Linetsky, 1999). The price recursion for step options reads as: for $n = N-1,\dots,0$

$$V(S,I,n) = e^{-r\Delta} \int_0^{+\infty} p(S,\xi;\Delta) V\left(\xi, I + \mathbf{1}_{\{\xi\in\mathcal{I}\}}, n+1\right) d\xi.$$

We also refer to (Cai, Chen, & Wan, 2010) for further details.

3. The quadrature approach and the "Breaking into Pieces" algorithm

As shown in the previous section, the general pricing framework requires the numerical computation of the following recursive integral equation

$$W(x,n) = \int_a^b H(x,y;\Delta) W(y,n+1) dy, \quad \forall x \in (a,b), \tag{5}$$

for $n = N-1,\dots,0$, with $W(x,N)$ assigned. This recursion holds for European and barrier options. Bermudan options also require an early exercise clause.

If we have more than one state variable (such as for lookback, Asian and step derivatives), the function W depends on the time index n and on two state variables, so that we write $W(x,\cdot,n)$. The additional state variable is the minimum value J if lookback options are considered, the average value A for Asian contracts and the occupation time I for step options.

If we apply a quadrature formula to (5), with nodes s_i and weights $w_i, i = 0,\dots,m-1$, we obtain

$$W(s_i,n) = \sum_{j=0}^{m-1} w_j H(s_i,s_j;\Delta) W(s_j,n+1). \tag{6}$$

If we define the matrix $\mathbf{H}_m$ as $\mathbf{H}_m = \left[H(s_i,s_j,\Delta)\right]_{i,j=0}^{m-1}$, then (6) can be written as

$$\mathbf{W}_n = \mathbf{H}_m \mathbf{D}_m \mathbf{W}_{n+1}, \quad n = N-1,\dots,0, \tag{7}$$

where $\mathbf{W}_n = [W(s_i,n)]_{i=0}^{m-1}$, and $\mathbf{D}_m = \mathrm{diag}(w_0,\dots,w_{m-1})$. The matrix $\mathbf{H}_m$ is called the sampling matrix of the function H, while $\mathbf{H}_m\mathbf{D}_m$ is the iteration matrix of the backward procedure (in the following we omit the subscripts to lighten the notation). Due to additional contractual features, such as early exercise, lookback or time averages, the recursion in (7) needs additional changes. In particular, the iteration matrix is the same for European, Bermudan, Asian, lookback and step options. The knock-out trigger event in barrier options generates a different structure of the domain Ω and thus of the iteration matrix. This is discussed in Appendix A.

In the recursion (7) the size of the iteration matrix equals the number of discretization points (nodes). It is well-known that increasing the number of nodes improves the accuracy of the solution. More precisely, the speed of convergence of the quadrature error to zero can be determined by using results on the speed of convergence of the integration rule when it is applied to the integral $\int_\Omega H(\cdot,\xi;\Delta)d\xi$, as discussed in Atkinson (2009, Chapter 4). In this setting the backward recursion (7) has a cost proportional to Nm^2 operations.

Our aim is to reduce this cost substantially by exploiting the spectral properties of the iteration matrix **HD**, analyzed in details in Appendix B, as the matrix size m grows. In fact, in the present context, the iteration matrix has two interesting properties: (1) its eigenvalues are clustered at zero, i.e., only a small number of non-negligible eigenvalues can be retained, and (2) it is close, within a prefixed tolerance, to a banded matrix. Given these properties, the main idea is at first to replace the iteration matrix by its factorization $\mathbf{PJQ}^H$, where $\mathbf{P}$ and $\mathbf{Q}$ are unitary matrices, $\mathbf{Q}^H$ is the hermitian of matrix $\mathbf{Q}$, and $\mathbf{J}$ is a bidiagonal matrix.[1] The factorization gives a relevant computational advantage in performing the matrix–vector multiplications required at each step of the backward procedure. The factorization, performed via the bidiagonalization algorithm of Golub and Kahan (1965), is very fast given the clustering property of the eigenvalues of the iteration matrix. In fact, as proved in Appendix B.2, the number r_ϵ of eigenvalues greater than a fixed tolerance ϵ is small with respect to the size m of the given matrix (usually, in applications, we use matrices of size at most 4000×4000). To grasp the idea, if we consider an Asian option with 252 monitoring dates, and we let m to grow from 1000 to 4000, r_ϵ remains constant at 412 (this will be throughly discussed in the numerical section part).

Secondly, due to the fact that the matrix **HD** is obtained by the discretization of a transition probability density, it turns out to be 'nearly' banded: the significant entries having values larger than a prefixed tolerance are confined to a diagonal band. The banded behavior is illustrated by the coloured part of the iteration matrix in Fig. 2. Therefore we can exploit the banded structure by "breaking into pieces" the matrix **HD** as shown in the same figure. In practice, we factorize separately each piece via the bidiagonalization algorithm. This is possible because each sub piece of the sampling matrix **HD** inherits its cluster property, so that it is convenient to perform the factorization on matrices of a smaller size. This makes the procedure even faster (especially when the size of the original matrix is very large). It is clear that a too large band leads to consider "pieces of matrix" having a large size, so that we do not achieve any benefit from the suggested breakdown procedure. In this case, the standard quadrature will remain the preferred approach. However, if a band structure is detected, once each piece is factorized we can calculate the matrix–vector product as shown in Fig. 2: the original matrix (top left corner of Fig. 2) is "broken" into smaller pieces (top right corner of Fig. 2). Then each piece is factorized using the Golub-Reinsch algorithm (bottom left corner of Fig. 2) and the matrix–vector product is computed exploiting the Householder or bidiagonal structure of the matrices involved. Finally the resulting vectors are "summed up" taking into account the overlapping parts.

In conclusion, the combination of the clustering and bandwidth properties allows us to reduce the computational cost of the pricing procedure, thanks to the factorization, which can be performed in a fast way due to the spectral properties above mentioned. More precisely, since in the recursive quadrature Eq. (7) we have to compute N matrix–vector products, the classical approach requires $O(Nm^2)$ operations against $O(k_\epsilon m^2 + Nk_\epsilon m)$ operations, being $O(k_\epsilon m^2)$ ($O(k_\epsilon m)$) the cost of the factorization (matrix–vector multiplication). In practice $k_\epsilon \approx r_\epsilon$, and we have a cost reduction if $r_\epsilon < mN/(N+m)$. In particular, as the number of monitoring dates increases, the cost reduction is effective if $r_\epsilon < m$, that, in general, it is always the case for the examples here considered, such as the previously mentioned Asian option contract. Using the proposed "Breaking into pieces" algorithm, it is thus possible even to achieve a larger time reduction exploiting the structure of the matrix **HD**, since the factorization and the matrix–vector products are computed considering the smaller submatrices and not the whole (large) iteration matrix, as shown in Fig. 2. The idea here described find a theoretical support in Appendix B.

[1] For the sake of completeness, we recall that the hermitian matrix of a matrix $\mathbf{Q}$ is its transpose conjugate, and $\mathbf{Q}$ is unitary if and only if $\mathbf{QQ}^H$ is the identity matrix.

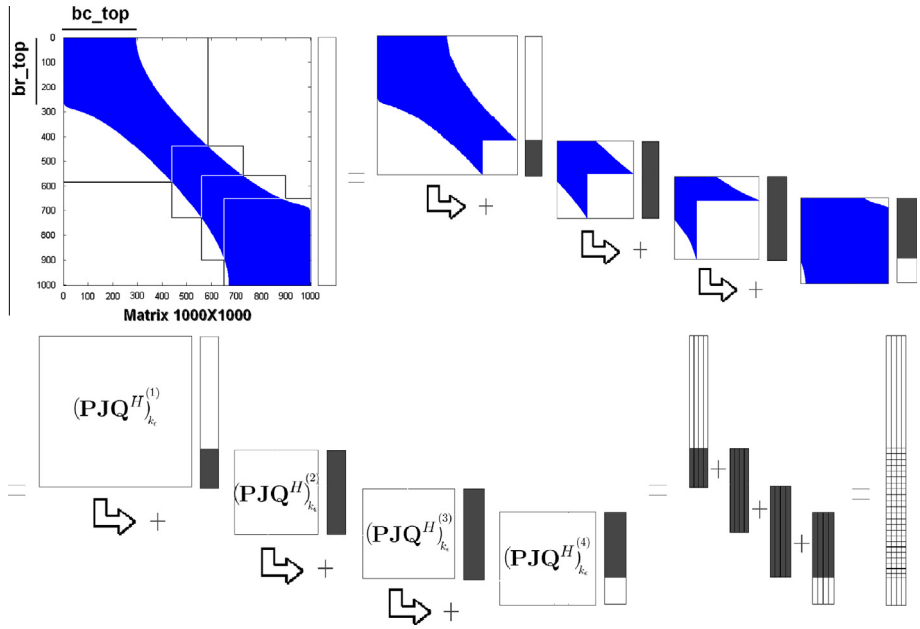

Fig. 2. "Breaking into pieces" a banded matrix. The black parts of the vectors overlap in the final sum. $b_{\text{top}} = \max\{bc_{\text{top}}, br_{\text{top}}\}$ is the bandwidth size.

4. Numerical results

In this section we validate with numerical experiments the ideas put forward in the previous section (theoretical results are in the appendix). The setup is absolutely general but for aim of clarity we consider the CEV process. This dynamics is indeed quite interesting allowing for very different transition densities and implied volatility shapes. For this reason, we describe it in some detail in Section 4.1. Then in Section 4.2 we show that the matrix **HD** obtained by the discretization of the CEV transition density exhibits eigenvalues strongly clustered at zero: if we increase the size (number of quadrature points) of the matrix, the number of eigenvalues greater than a given tolerance remains constant. Thus the CEV process admits the cluster property that allows us to improve the performance of the recursive approach. Finally, in Section 4.3 we apply our algorithm (hereafter denominated BP algorithm) to "break into pieces" the matrix **HD** and we apply it to price exotic derivatives. This algorithm is detailed with a pseudo-code in the electronic supplementary material.

For the numerical discretization of (5) we opt for a Gauss–Legendre quadrature (Quarteroni, Sacco, & Saleri, 2000). Numerical experiments not reported here have shown that the cluster of eigenvalues of the iteration matrix is independent of the adopted quadrature formulas, but the bandwidth of the matrix is larger for Gaussian quadrature respect to Newton–Cotes ones. However the results are similar, in terms of computational efficiency, for both classes of quadrature formulas.

All calculations were performed using Matlab R2008a on a PC Intel Core2 Quad 2.40 GHz with 3.24 GB RAM and Windows XP operating system.

4.1. The CEV process

Even if our algorithm is quite general, from now on, we assume that the underlying asset evolves according to a CEV process (Cox, 1996), i.e.,

$$dS(t) = rS(t)dt + \sigma S^{\beta+1}(t)dW(t), \quad S(0) = S_0, \tag{8}$$

and thus the transition probability density is given by

$$p(S,\xi;\Delta) := e^{-r\Delta}\, p_0\left(S,\ e^{-r\Delta}\xi;\ \frac{1}{2r\beta}\left(e^{2r\beta\Delta}-1\right)\right),$$

with

$$p_0(S,\xi;\Delta) = \frac{\xi^{-2\beta-\frac{3}{2}}S^{\frac{1}{2}}}{\sigma^2|\beta|\Delta}\ e^{-\frac{S^{-2\beta}+\xi^{-2\beta}}{2\sigma^2\beta^2\Delta}}\ I_{\frac{1}{2|\beta|}}\left(\frac{S^{-\beta}\xi^{-\beta}}{\sigma^2\beta^2\Delta}\right),$$

where I_ν is the modified Bessel function of the first kind of order ν. In particular, when $\beta = 0$ we have the classical geometric Brownian process (GBM), when $\beta = -1$ we have an arithmetic Brownian motion (ABM), while when $\beta = -0.5$ the Cox-Ingersoll-Ross square-root process (SR) is obtained. For details see also (Cox, 1996; Davydov & Linetsky, 2001; Fusai & Recchioni, 2007). In Fig. 3 we plot the density function for different values of the leverage parameter β (left panel) and the corresponding implied volatility curve (right panel). In particular, large negative values of β generate a skewed to the left density function and a very steep implied volatility curve, as often observed in the market. The CEV process, consistently with empirical studies, allows for the volatility to depend on the price level and in addition the two are negatively correlated (leverage effect); moreover, the model is able to generate the smirk effect often observed in the market implied volatility curve. See for example (Boyle & Tian, 1999; Costabile, 2006). Unfortunately, the transition density of the CEV process is not of convolution type, thus a fast computation of the recursion via the FFT is not feasible. For these reasons, the CEV dynamics turns out to be an interesting case to test our pricing procedure.

Numerical methods for pricing derivatives under the CEV process are presented, for example, in Boyle and Tian (1999), Boyle et al. (1999), Costabile (2006), Peng (2006). These articles price derivatives contracts, like barrier (Boyle & Tian, 1999), lookback (Boyle & Tian, 1999; Boyle et al., 1999; Costabile, 2006) and geometric Asian (Peng, 2006) options, assuming continuous monitoring and using a lattice approach, i.e., binomial or trinomial trees. In general, if we consider a non-Gaussian diffusion process, the diffusion coefficient is not constant and it is not possible to construct a recombining tree in the usual way. Therefore, the price process is transformed into another process having a constant diffusion coefficient, and the tree is built for the transformed process. In this

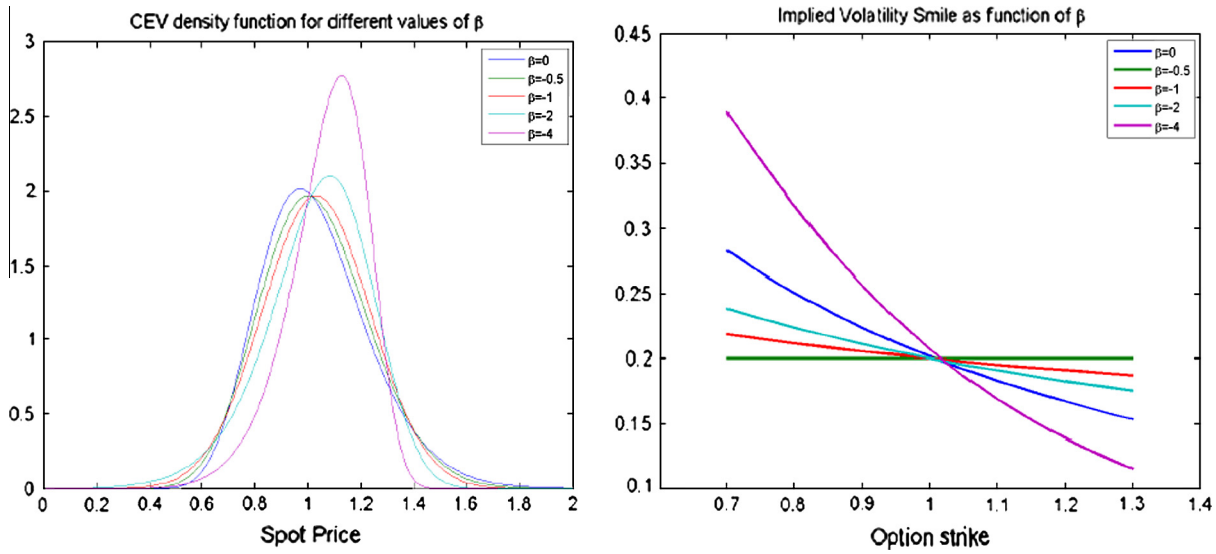

Fig. 3. Density function (left) and implied volatility (right) of the CEV model for different values of β.

case, advanced technique, like the adaptive averaging binomial method presented in Moon and Kim (2013) are available. Unfortunately, this procedure is reliable only for barrier options. For example, results in Boyle and Tian (1999) for lookback options are not accurate. To cope with this problem, in Costabile (2006) the author proposes a forward induction procedure that allows to compute the risk-neutral probability of each different payoff of the lookback option at maturity but with a computational cost that is cubic in the number of time step against the linear cost of the recursive quadrature. In addition, by using trees, we have slow and erratic convergence to the true price largely caused by the position of the barrier relative to the adjacent stock prices, see the theoretical results in Lin and Palmer (2013) with reference to the GBM dynamics. We can have large errors even with thousands of time steps and millions of node calculations. American options are considered in Nunes (2009), where the author proposes an alternative characterization of the early exercise premium that is valid for any Markovian and diffusive underlying price process. However, the author does not consider Bermudan options, for which early exercise can occur only on discrete dates. Finally, an analytical Laplace transform approach based on the scale function of a diffusion process is pursued in Davydov and Linetsky (2001). These authors obtain Laplace transform of barrier and lookback option prices involving Whittaker and Bessel functions of complex argument. Option prices are then obtained via a numerical inversion of the Laplace transform. Unfortunately, the procedure is quite computational intensive mainly for lookback options: this problem requires the numerical computation of an integral involving the inverse Laplace transform.

Unless otherwise specified, we consider the same parameter setting as in Davydov and Linetsky (2001): the initial asset price is $S_0 = 100$, the risk-free interest rate is 10% per annum ($r = 0.1$), the volatility is $\sigma = 0.25/S_0^{\beta}$. Moreover, we assume that the asset pays no dividends ($q = 0$), and all options have six months to expiration ($T = 0.5$). If necessary, we truncate the integration interval as stated in Section 2.1 with a 10^{-8} tolerance.

4.2. Cluster of the HD matrix

Table 1 provides the number of eigenvalues greater, in absolute value, than $\epsilon = 10^{-11}$ and the bandwidth size b_{top} of the matrix **HD**, see Fig. 2. The bandwidth size is fixed setting to zero the elements smaller, in absolute value, than 10^{-9}. The leverage parameter β in the CEV model is set equal to -0.5 (results for different values of β are reported in the electronic supplementary material). We stress that this table refers to pricing problems characterized by a different iteration matrix. We notice that:

- a strong cluster at zero always occurs. For example, let us consider down-and-out options in Table 1, given a number of monitoring dates $N = 52$. The number of significative eigenvalues (greater than ϵ) is 107, independently of the matrix dimension m;
- the cluster increases less than linearly with respect to the number of monitoring dates N. Thus our algorithm will achieve a large time reduction when N is large. In fact, we see in Table 1 that, given m, as the number of monitoring dates increases, the same happens to the number of eigenvalues greater than ϵ, but the ratio r_ϵ/N decreases;
- the presence of barriers strongly improves the cluster. This is evident if we compare the double barrier case in Table 1 to other contracts. In particular this suggests a relative better performance of the algorithm in pricing this kind of exotics;
- changing the value of the leverage parameter β in the CEV density does not affect the cluster. This is shown in the electronic supplementary material.

We can also make some additional considerations on the bandwidth of the matrix:

- we always have banded matrices; for example in Table 1 for European, lookback, Asian or step options with $N = 52$ and $m = 4000$, the bandwidth size is $b_{\text{top}} = 493$. However, the presence of barriers increases b_{top}. Indeed barriers cut the tails of the density so that we have to sample a function that does not approach zero on the frontier of the domain. In general, the ratio b_{top}/m remains constant as m varies.
- the bandwidth size decreases as we increase the number N of monitoring dates; this is, for example, confirmed for all contracts in Table 1 when $m = 4000$ and we let N to vary from 52 to 1008;

Since the performance of the algorithm is optimized when the cluster size r_ϵ and the bandwidth size b_{top} are both small, the above remarks suggest that this happens in all cases and the greatest benefit occurs as we increase N.

Table 1
Number r_ϵ of eigenvalues of **HD** greater than $\epsilon = 10^{-11}$. Legend: m is the matrix dimension, N is the number of monitoring dates, $\beta = -0.5$ is the leverage parameter in (8) and b_{top} is the bandwidth size of the matrix **HD**. Contracts are grouped according to the iteration matrix **HD**.

Contract	N	r_ϵ m				Bandwidth = b_{top} m			
		1000	2000	3000	4000	1000	2000	3000	4000
European	52	190	190	190	190	126	249	372	493
Lookback	104	266	267	267	267	105	208	310	411
Asian	252	411	412	412	412	83	165	247	328
Step	504	576	580	581	581	70	139	207	275
	1008	774	819	819	832	59	116	174	231
Down-and-out	52	107	107	107	107	209	414	617	818
	104	149	149	149	149	174	345	514	683
	252	230	230	230	229	139	275	410	544
	504	322	322	323	322	116	231	344	456
	1008	455	455	455	454	98	194	289	384
Up-and-out	52	115	115	115	115	183	362	540	716
	104	161	161	161	161	152	301	449	596
	252	247	247	247	247	121	239	357	474
	504	348	348	348	348	101	201	299	397
	1008	489	490	490	490	85	168	251	333
Double barrier	52	32	32	32	32	471	929	1382	1831
	104	44	44	43	43	383	757	1126	1493
	252	65	65	65	65	300	592	882	1170
	504	90	90	90	90	249	493	734	974
	1008	125	125	125	125	208	412	614	814

Table 2
Speed-up values for different contracts with $\beta = -0.5$ and $m = 4000$. The initial asset price is $S_0 = 100$, the volatility is $\sigma = 0.25/S_0^\beta$, the risk-free interest rate is 10% per annum ($r = 0.1$), the asset pays no dividends ($q = 0$), and all options have six months to expiration ($T = 0.5$). The strike price E is equal to 105. Additional payoff's parameters for step options are $\rho = 0.5$ and $A = [90, 110]$. ω is the parameter in Remark 1 in the appendix.

	$N = 252$	$N = 504$	$N = 1008$
European	1.1604	1.3270	1.4151
Barrier Down-and-out	1.1045	1.4104	1.7289
Barrier Up-and-out	1.1784	1.4872	1.7964
Double barrier	0.7263	0.9809	1.9535
Bermudan	1.1634	1.3476	1.4189
Bermudan Down-and-out	1.1102	1.4176	1.7126
Bermudan Up-and-out	1.1891	1.5065	1.7883
Bermudan Double barrier	0.7315	0.9867	1.9412
Lookback	2.3617	2.1688	1.9565
Lookback ($\omega = 4$)	3.0116	2.8686	2.5773
Asian	1.8654	1.7169	1.5609
Asian ($\omega = 4$)	1.9707	1.9769	1.9624
Step	4.3515	3.6113	3.1951

Table 3
Prices in Davydov and Linetsky (2001, Table 1).

	$\beta = 0$	$\beta = -0.5$	$\beta = -1$
European	7.0995	7.0170	6.9403
Down-and-out	6.3722	6.2554	6.1438
Up-and-out	0.6711	0.7734	0.8904
Double barrier	0.4418	0.5126	0.5945

Table 4
European call: m is the matrix dimension and β is the leverage parameter in (8). Parameters as in Table 2.

β	m	Prices		CPU times (second)	
		Rec.	Rec.+BP	Rec.	Rec.+BP
0	2000	7.099596	7.099596	1.01	4.60
	4000	7.099571	7.099571	3.90	18.20
−0.5	2000	7.017063	7.017063	6.09	10.16
	4000	7.016999	7.016999	23.63	39.66
−1	2000	6.940388	6.940388	8.09	12.96
	4000	6.940318	6.940318	31.40	50.87

In Table 2 we show the speed-up for different contracts. Here the speed-up is defined as the ratio between the CPU time for the classical recursive (Rec.) algorithm and the one for our pricing procedure (Rec.+BP). As expected, the speed-up is always greater than 1. Double barrier options are the exception when the number of monitoring dates is small, due to a too large bandwidth.

4.3. Pricing options

In this section we validate our pricing procedure comparing it to standard numerical quadrature and to Monte Carlo simulation. This has been implemented with an Euler discretization scheme with 300 steps between two consecutive monitoring dates, and 1,000,000 runs. In general Monte Carlo simulations applied to the CEV process achieve a two digits accuracy, but with a CPU time that turns to be higher than the recursive quadrature of a factor that varies from 4 to 10.[2] Extended numerical results are reported in the electronic supplementary material.

4.3.1. Barrier and Bermudan options

In Tables 4,5 we price European and barrier call options. Analytical formulas for European options are available in terms of the non-central chi-square distribution, see (Schroder, 1989). Prices of continuously monitored barrier options, i.e., $N = \infty$, are given in Davydov and Linetsky (2001) and reported here in Table 3.

Since European options are path-independent contracts, their pricing requires a single recursion. For this reason, in Table 4 we

[2] We also considered an exact Monte Carlo simulation by sampling from the known transition cumulative density function, but the procedure turns out to be too time consuming and of no practical relevance.

Table 5
Down-and-out and double barrier call: $\beta = -0.5$ is the parameter in (8), m is the matrix dimension and N is the number of monitoring dates. Parameters as in Table 2.

N	m	Down-and-out call				Double barrier call			
		Prices		CPU (second)		Prices		CPU (second)	
		Rec.	Rec.+BP	Rec.	Rec.+BP	Rec.	Rec.+BP	Rec.	Rec.+BP
52	2000	6.497277	6.497277	4.47	6.03	0.771025	0.771025	4.16	8.70
52	4000	6.497278	6.497278	17.13	23.74	0.771024	0.771024	16.12	34.37
104	2000	6.434699	6.434699	4.87	5.51	0.694140	0.694140	4.48	8.41
104	4000	6.434700	6.434700	18.73	22.22	0.694140	0.694140	17.21	32.49
252	2000	6.375374	6.375374	6.03	5.70	0.628248	0.628248	5.39	7.61
252	4000	6.375375	6.375375	23.24	21.04	0.628248	0.628248	20.68	28.47
504	2000	6.342071	6.342072	7.91	7.06	0.593922	0.593922	6.94	7.46
504	4000	6.342072	6.342073	30.65	21.73	0.593922	0.593922	26.77	27.29
1008	2000	6.317620	6.317621	11.63	11.24	0.569846	0.569846	13.99	8.47
1008	4000	6.317621	6.317623	45.04	26.05	0.569846	0.569846	54.64	27.97
10,000	2000	6.275649	6.275652	72.95	122.85	0.530602	0.530603	85.20	49.35
10,000	4000	6.275651	6.275652	282.11	154.93	0.530602	0.530604	331.26	91.33

set $N = 1$ and the BP algorithm is not at all convenient because the factorization is too costly with respect to a single matrix–vector multiplication. However the algorithm has the same accuracy as the analytical formula: our price estimates agree with those of the first row of Table 3.

Numerical results for barrier options with $\beta = -0.5$ are given in Table 5. Prices for different values of β and for up-and-out options are reported in the electronic supplementary material. We notice that:

- prices computed with the BP algorithm agree with the ones from the pure recursion up to five decimal digits;
- as expected, the BP algorithm performs better as we increase the number of monitoring dates N, since the factorization has to be performed only once, and, at the same time, the bandwidth decreases (see Table 1 – we recall that our algorithm performs well if the bandwidth size is not too large). In fact, from Table 5 we notice the benefits of the BP factorization for $N = 252$ or greater;
- the algorithm works better as we increase the number of quadrature nodes m, since the cluster size r_ϵ does not increase varying m, while the computational cost of the matrix–vector multiplication increases;
- for double barrier options we observe that, increasing N, we have a trade-off between the cluster size (it improves) and the bandwidth size (it becomes larger). On this point we can make two remarks:
  1. In general, our algorithm improves the standard recursion for N larger than 252. Additional numerical tests have shown that the algorithm applied to the double barrier case can achieve a speed-up up to 3.6 when $N = 10,000$.
  2. Numerical results in Table 2 show that the BP algorithm performs better for single barrier respect to double barrier options if N is lower than 1008.
- concerning the convergence of the discrete monitoring price to the continuous monitoring case, we notice a slow convergence from above of prices in Table 5 to the ones in Table 3. For example, when $\beta = -0.5$ and $N = 10,000$, a single barrier option with discrete monitoring is worth 6.2756, whilst the continuous version is 6.2554. Moreover, comparing the two tables, it is clear that pricing a discrete monitoring contracts with a continuous monitoring algorithm could result in a substantial different, and thus wrong, price. As an example, considering $\beta = -0.5$ and assuming to price a twice-a-day monitoring double barrier option, i.e., $N = 52$, the real price, i.e., the one computed considering the real monitoring is 0.7710, while the one computed assuming the continuous monitoring is 0.5126. Thus the importance of considering numerical algorithms with discrete monitoring, when contracts with this feature have to be priced.

Table 7
Monte Carlo values for fixed-strike lookback put options with 1,000,000 iterations, parameters as in Table 6.

N	Confidence interval	CPU times (second)
52	14.5242–14.5576	2523
104	14.8738–14.9073	5014
252	15.1781–15.2116	12,104
504	15.3483–15.3818	24,180
1008	15.4511–15.4846	48,331

Table 6
Fixed-strike lookback put: $\beta = -0.5$ is the parameter in (8), m is the matrix dimension and N is the number of monitoring dates. Parameters as in Table 2. ω is the parameter in Remark 1 in the appendix.

N	m	Prices				CPU times (second)			
		Rec. $\omega = 1$	Rec.+BP $\omega = 1$	Rec.+BP $\omega = 2$	Rec.+BP $\omega = 4$	Rec. $\omega = 1$	Rec.+BP $\omega = 1$	Rec.+BP $\omega = 2$	Rec.+BP $\omega = 4$
52	2000	14.545701	14.545701	14.544358	14.553307	592	323	140	64
52	4000	14.542978	14.542978	14.542853	14.546845	4566	1422	703	332
104	2000	14.887939	14.887940	14.886958	14.893453	1069	629	270	124
104	4000	14.886366	14.886366	14.886282	14.889097	8084	2897	1381	618
252	2000	15.191305	15.191306	15.190640	15.194984	2193	1361	598	254
252	4000	15.190981	15.190982	15.190932	15.192716	16,567	7015	3212	1376
504	2000	15.353628	15.353629	15.353129	15.356335	3953	2684	1194	534
504	4000	15.354248	15.354250	15.354219	15.355452	29,570	13,634	5969	2573
1008	2000	15.469194	15.469194	15.468811	15.471220	6886	5117	2043	986
1008	4000	15.470853	15.470857	15.470839	15.471678	51,971	26,563	11,362	5025

Table 8
Fixed-strike Asian call: $\beta = -0.5$ is the parameter in (8), m is the matrix dimension and N is the number of monitoring dates. Parameters as in Table 2. ω is the parameter in Remark 1 in the appendix.

N	m	Prices				CPU times (second)			
		Rec. $\omega = 1$	Rec.+BP $\omega = 1$	Rec.+BP $\omega = 2$	Rec.+BP $\omega = 4$	Rec. $\omega = 1$	Rec.+BP $\omega = 1$	Rec.+BP $\omega = 2$	Rec.+BP $\omega = 4$
52	2000	2.919996	2.919996	2.920009	2.919707	843	573	284	157
52	4000	2.920010	2.920010	2.920007	2.920017	5615	2377	1186	673
104	2000	2.928446	2.928446	2.928341	2.926760	1642	1128	543	307
104	4000	2.928465	2.928465	2.928459	2.928347	10,072	4814	2379	1304
252	2000	2.933353	2.933353	2.932724	2.929454	3592	2591	1370	725
252	4000	2.933498	2.933499	2.933402	2.932702	21,221	11,376	5755	3140
504	2000	2.934878	2.934878	2.933790	2.929121	6448	5137	2796	1400
504	4000	2.935238	2.935238	2.934972	2.933777	38,909	22,662	11,140	5495
1008	2000	2.935444	2.935444	2.933788	2.927539	12,413	9834	5488	2992
1008	4000	2.936030	2.936031	2.935583	2.933823	71,120	45,563	22,044	10,207

Experiments not reported here show similar performances for Bermudan options. The results are not affected by the presence or absence of dividends.

4.3.2. Lookback options

Pricing lookback options is in general more expensive than pricing barrier options, because we have to keep trace of an additional state variable, the running minimum J. Thus, we expect an increase in the CPU time with respect to barrier and Bermudan option. However, since the matrix factorization is independent on the J-grid nodes, we still expect an improvement with respect to the standard recursion. Results reported in Table 6 confirm this. In addition the two recursive algorithms show comparable accuracy. Table 7 provides, as benchmark, confidence intervals computed by the Monte Carlo algorithm. An additional speed-up can be obtained considering less nodes on the J-grid, up to ten times if we reduce by a factor of four the nodes on the J-grid ($\omega = 4$ in the mentioned table), maintaining a two decimal digits accuracy.

Table 9
Monte Carlo values for fixed-strike Asian call options with 1,000,000 iterations, parameters as in Table 8.

N	Confidence interval	CPU times (second)
52	2.9151–2.9356	2523
104	2.9196–2.9401	5014
252	2.9272–2.9477	12,104
504	2.9283–2.9487	24,180
1008	2.9264–2.9468	48,331

4.3.3. Asian options

Asian options (see Section 2.4) share with lookback options the presence of an additional state variable. Given that the iteration matrix is the same for the two contracts, we expect a similar performance of the algorithm in the Asian as in the lookback case. Results are given in Table 8, and comments given in Section 4.3.2 still

Table 10
Fixed-strike Asian call: A comparison between the Rec.+BP algorithm and (Černý & Kyriakou, 2011, Table 7): Gaussian case ($\beta = 0$) and strike price $E = 90$. Benchmark price: 11.58113 ($\sigma = 0.1$), 13.66981 ($\sigma = 0.3$) and 17.19239 ($\sigma = 0.5$).

ω	m	σ					
		Tol 10^{-8}			Tol 10^{-10}		
		0.1	0.3	0.5	0.1	0.3	0.5
1	1000	11.58113	13.66975	17.19193	11.58113	13.66991	17.19255
1	2000	11.58113	13.66977	17.19196	11.58113	13.66980	17.19236
1	4000	11.58113	13.66977	17.19192	11.58113	13.66982	17.19240
2	1000	11.58118	13.67068	17.19441	11.58120	13.67232	17.19750
2	2000	11.58113	13.66974	17.19191	11.58113	13.66982	17.19272
2	4000	11.58113	13.66977	17.19191	11.58113	13.66982	17.19239
4	1000	11.58175	13.67434	17.20552	11.58184	13.68018	17.20388
4	2000	11.58118	13.67081	17.19409	11.58121	13.67196	17.19835
4	4000	11.58113	13.66973	17.19192	11.58113	13.66981	17.19270

Table 11
Step call: m is the matrix dimension, N is the number of monitoring dates. Parameters as in Table 2.

N	m	Prices		CPU times (second)		Monte Carlo values	
		Rec.	Rec.+BP	Rec.	Rec.+BP	Confidence interval	CPU times (second)
52	2000	5.580878	5.580878	13.08	7.46	5.5772–5.6133	2446.38
52	4000	5.581670	5.581670	50.64	23.71		
104	2000	5.564554	5.564554	35.21	13.98	5.5473–5.5832	4881.28
104	4000	5.565345	5.565345	134.63	35.45		
252	2000	5.554981	5.554981	154.47	60.41	5.5414–5.5772	11,811.86
252	4000	5.555772	5.555773	589.53	135.57		
504	2000	5.551620	5.551620	533.47	241.04	5.5367–5.5725	23,616.72
504	4000	5.552411	5.552412	2003.21	554.70		
1008	2000	5.549940	5.549940	1823.83	1044.87	5.5276–5.5634	47,231.25
1008	4000	5.550731	5.550732	6931.57	2169.42		

apply. For example, if we set $\omega = 4$, we reduce the CPU time by a factor of seven, still maintaining a two decimal digits accuracy. Reported option prices always fall into the Monte Carlo confidence intervals (see Table 9).

Finally, in Table 10 we consider the log-normal process ($\beta = 0$). In this case, indeed, it is more efficient to use a FFT approach, as in Černý and Kyriakou (2011), or a randomization technique, as in Fusai, Marazzina, and Marena (2011). We use the results in Černý and Kyriakou (2011, Table 7) as benchmark. In Table 10 we also analyze the effect on the price accuracy of the truncation of the integration interval (see Section 2.1). The Rec.+BP algorithm always achieves a three to five decimal digits accuracy depending on the tolerance level when we truncate the domain. Significant reduction in the CPU time, without loss of accuracy, can be achieved with a sparser grid ($\omega = 2$ and $\omega = 4$) on the running average.

4.3.4. Step options

Numerical results given in Table 11 show that our algorithm applied to step options achieves the same accuracy as the direct recursive procedure: they agree up to the sixth digit, but with a strong reduction in the computational time. Thus, also for this kind of contracts the Rec.+BP algorithm is more efficient than the plain recursion and Monte Carlo simulation.

5. Conclusion

In this paper we have shown how to price exotic options when discrete monitoring is considered exploiting the structure of the matrix arising from the numerical quadrature of the pricing backward formulas. This has been accomplished through a convenient factorization of the iteration matrix that helps greatly in the speed-up of the recursion. The proposed BP algorithm is general and is examined in detail with reference to the CEV process, for which, according to our knowledge, no efficient and general enough procedure is available in literature, and to discretely monitored option.

Our numerical experiments show that the BP algorithm performs well respect to the standard recursive quadrature for all considered exotic options (barrier, lookback, Asian, Bermudan and step options), mainly when the number of monitoring dates is large: they have a similar accuracy, but our scheme is considerably faster. In addition, we show how to accelerate the scheme for both lookback and Asian options, losing a little bit of accuracy.

We would like to stress that, according to our knowledge, this article is the first one investigating a fast and accurate algorithm for pricing exotic options when the discrete monitoring is assumed and a CEV process (or, more generally, processes for which the log-price is not i.i.d.) is considered. Finally the proposed factorization has potential and immediate applications to the study of properties of discrete time Markov Chains as well, a very important topic in operational research for modeling queuing sequences and many other practical systems, see (Norris, 1997). Extension to other processes, as normal mixture distribution, see (Bhat & Kumar, 2012), and options with multiple risk factors, such as stochastic volatility models or multi-assets contracts, as in Breton and de Frutos (2010), Jin, Li, Tan, and Wu (2013), Rambeerich, Tangman, Lollchund, and Bhuruth (2013), are also amenable to the presented technique, and will be treated in a follow-up paper.

Appendix A. Quadrature for exotic derivatives

In the following we detail how the quadrature applies to different contractual settings.

A.1. Barrier and Bermudan options

For barrier options, i.e., recursions (1) and (2), we have $W(x,N) = \phi(x), H(x,y;\Delta) = e^{-r\Delta}p(x,y;\Delta)$, and a, b depend on the domain Ω. For example, for down-and-out barrier options, we set $a = L$ and $b = \mathcal{U}$. From the discretization of (1), we obtain a recursion of the form (7) with $\mathbf{W}_N = [\phi(s_i)]_{i=0}^{m-1}$.

If Bermudan options are considered, we have $W(x,N), H(x,y;\Delta)$, and a, b as above. From the discretization of (2) we obtain

$$\begin{cases} \mathbf{W}_n^{CV} = \mathbf{H}_m \mathbf{D}_m \mathbf{W}_{n+1} \\ \mathbf{W}_n = \max\{\mathbf{W}_n^{CV}, \mathbf{\Phi}\} \end{cases}$$

where $\mathbf{W}^{CV}$ is the continuation value, and $\mathbf{\Phi} = [\phi(s_i)]_{i=0}^{m-1} = \mathbf{W}_N$.

A.2. Lookback options

If fixed-strike lookback put options are considered, we set $H(x,y;\Delta) = e^{-r\Delta}p(x,y;\Delta), a = 0$ and $b = +\infty$ (truncated to $\mathcal{L}$ and $\mathcal{U}$ for numerical valuation). Thus the semi-discrete formulation of (3) is

$$W(s_i,J,n) = \sum_{l=0}^{m-1} w_l H(s_i,s_l;\Delta) W(s_l, \min\{J,s_l\}, n+1),$$

$i = 0,\dots,m-1$, with $W(s_i,J,N) = (E - \min\{s_i,J\})^+$. Since J is the minimum value of the underlying asset, we discretize J on the same grid $\{s_i\}_{i=0}^{m-1}$ used for the underlying asset. More precisely, we can implement recursion (3) as follows: considering m quadrature nodes s_j and weights w_j, $j = 0,\dots,m-1$, we define for $n = N-1,\dots,0$ the vectors

$$\mathbf{W}_n^j := [W(s_i,s_j,n)]_{i=0}^{m-1} \quad \text{and} \quad \widehat{\mathbf{W}}_n^j := [W(s_i,\min\{s_i,s_j\},n)]_{i=0}^{m-1},$$

with $\widehat{\mathbf{W}}_N^j = [(E - \min\{s_i,s_j\})^+]_{i=0}^{m-1}$.

The fully-discretized lookback recursion (3) is: for $n = N-1,\dots,0$

$$\mathbf{W}_n^j = \mathbf{H}_m \mathbf{D}_m \widehat{\mathbf{W}}_{n+1}^j,$$

with $\mathbf{D}_m$ and $\mathbf{H}_m$ defined as above. Notice that $(\widehat{\mathbf{W}}_n^j)_i$ corresponds to $W(s_i,s_j,n)$ (and thus to $(\mathbf{W}_n^j)_i$) only if $s_i \geqslant s_j$, and thus $i \geqslant j$. Thus, moving from $\mathbf{W}_n^j$ to $\widehat{\mathbf{W}}_n^j$, an update of the minimum value is necessary for the indices i such that $s_j > s_i$. This implies that the pricing recursion can be written as: for $n = N-1,\dots,0$ and $j = 0,\dots,m-1$,

$$\begin{cases} \mathbf{W}_n^j = \mathbf{H}_m \mathbf{D}_m \widehat{\mathbf{W}}_{n+1}^j & \text{Recursion Step,} \\ (\widehat{\mathbf{W}}_n^j)_i = \begin{cases} (\mathbf{W}_n^j)_i & \text{if } i \geqslant j, \\ (\mathbf{W}_n^i)_i & \text{if } i < j, \end{cases} & \text{Updating Step.} \end{cases} \tag{A.1}$$

Remark 1. The iteration (A.1) can be accelerated using a subset $\widetilde{s}$ of m/ω quadrature nodes, i.e., for $n = N-1,\dots,0$, for $j_\omega = 0,\dots,m/\omega - 1$, being $(\mathbf{W}_n^{j_\omega})_i = W(s_i,\widetilde{s}_{j_\omega},n)$, it holds

$$\begin{cases} \mathbf{W}_n^{j_\omega} = \mathbf{H}_m \mathbf{D}_m \widehat{\mathbf{W}}_{n+1}^{j_\omega}, \\ (\widehat{\mathbf{W}}_n^{j_\omega})_i = \begin{cases} (\mathbf{W}_n^{j_\omega})_i & \text{if } i \geqslant j_\omega, \\ (\mathbf{W}_n^i)_i & \text{if } i < j_\omega, \end{cases} \end{cases}$$

where the element $(\mathbf{W}_n^i)_i$ is computed by cubic interpolation if the node s_i do not belong to the subgrid $\widetilde{s}$.

A.3. Asian options

For Asian options, at each step the possible new values of the state variable A do not fall on the A-grid at the previous step.

Therefore an interpolation is also required at each iteration. More precisely, the semi-discrete formulation of (4) is: for $i = 0, \dots, m-1$

$$W(s_i, A, n) = \sum_{l=0}^{m-1} w_l H(s_i, s_l; \Delta) W\left(s_l,\ \frac{n+1}{n+2}A + \frac{1}{n+2}s_l,\ n+1\right).$$

To obtain the fully-discrete formulation, we define for $j = 0, \dots, m-1$

$$\mathbf{W}_n^j = (W(s_i, s_j, n))_{i=0}^{m-1} \quad \text{and} \quad \hat{\mathbf{W}}_n^j = \left(W\left(s_i, \frac{n}{n+1}s_j + \frac{1}{n+1}s_i,\ n\right)\right)_{i=0}^{m-1}.$$

Thus the discretization of (4) can be written as: for $n = N-1, \dots, 0$, given $\mathbf{W}_{n+1}^j$, compute $\hat{\mathbf{W}}_n^j$ exploiting cubic interpolation, and then set

$$\mathbf{W}_n^j = \mathbf{H}_m \mathbf{D}_m \hat{\mathbf{W}}_{n+1}^j.$$

Again, as stated in Remark 1, we can assume that the average falls on a subset $\tilde{s}$ of m/ω quadrature nodes, i.e., $\mathbf{W}_n^{j_\omega} = (W(s_i, s_{j_\omega}, n))_{i=0}^{m-1}$, $j_\omega = 0, \dots, \frac{m}{\omega} - 1$.

A.4. Step options

Finally, for step derivatives, I_n can only assume the $n+1$ values $0, 1, \dots, n$. The payoff is discretized according to

$$\widehat{\mathbf{W}}_N^j = \left(\left(1 - \frac{\rho}{N}\left(j + \mathbf{1}_{\{s_i \in \mathcal{I}\}}\right)\right)(\varphi(s_i - E))^+\right)_{i=0}^{m-1}, \quad j = 0, \dots, N-1.$$

Then, for $n = N-1, \dots, 0$, we compute

$$\mathbf{W}_n^j = \mathbf{H}_m \mathbf{D}_m \widehat{\mathbf{W}}_{n+1}^j, \quad j = 0, \dots, n,$$

and then, if $n > 0$, we construct $\widehat{\mathbf{W}}_n^j$ as follows: for $i = 0, \dots, m-1$, for $j = 0, \dots, n-1$, if $s_i \notin \mathcal{I}$, then $\left(\widehat{\mathbf{W}}_n^j\right)_i = \left(\mathbf{W}_n^j\right)_i$, otherwise $\left(\widehat{\mathbf{W}}_n^j\right)_i = \left(\mathbf{W}_n^{j+1}\right)_i$.

Appendix B. Matrix factorization

In this section, we analyze the spectral properties of the matrix **HD**. These properties allow us to factorize **HD** by using the bidiagonalization algorithm of Golub and Kahan (1965). For this purpose, at first we define the cluster point of eigenvalues of a sequence of matrices of increasing dimension (that can be obtained, for example, by increasing the number of nodes in the quadrature formula (7)). This means to analyze the asymptotic behavior of eigenvalues of a sequence of matrices, and to show that, in our case, they cluster around zero. In particular, the number r_ϵ of eigenvalues greater than a fixed tolerance ϵ remains constant as we increase the number of quadrature nodes. Therefore the factorization of the matrix **HD** into the product $\mathbf{PJQ}^H$ returns a bidiagonal matrix **J** having as non-zero elements only the first k_ϵ elements, $k_\epsilon \approx r_\epsilon$, on the main and on the upper diagonals. This means that it is not necessary to perform the full factorization $\mathbf{PJQ}^H$, but we can stop at the k_ϵth step. If k_ϵ is much smaller than m, where m is the dimension of the matrix, this implies a significant reduction in the computational cost. Furthermore, the particular "composition" of matrices **P** and **Q** allows us to compute the matrix–vector multiplication in $O(k_\epsilon m)$ operations instead of the $O(m^2)$ characterizing the standard recursive approach.

B.1. Spectral properties of a sequence of matrices

In this section we give the main definitions and theorems related to the spectral properties of sequences of matrices that satisfy certain conditions.

In the following, we denote with $\{\mathbf{A}_m\} = \{\mathbf{A}_m\}_{m=1}^{\infty}$ a sequence of matrices in $\mathbb{C}^{m\times m}$ joined by a structural content that remains unchanged when the size varies. For brevity, we will denote the sequence simply by $\{\mathbf{A}_m\}$ and we use the symbols $\lambda_j^{(m)}$ and $\sigma_j^{(m)}$, $j = 1, \dots, m$, to denote, respectively, the eigenvalues and the singular values of $\mathbf{A}_m$.

A property of the spectrum of a sequence of matrices is its cluster point.

Definition 1 Tyrtyshnikov (1996). A matrix sequence $\{\mathbf{A}_m\}$ is *strongly clustered at* $s \in \mathbb{C}$ (in the eigenvalue sense), if for any $\varepsilon > 0$ the number of the eigenvalues of $\mathbf{A}_m$ off the disc $D(s, \varepsilon) := \{z : |z - s| < \varepsilon\}$ can be bounded by a pure constant q_ε possibly depending on ε, but not on m. In other words

$$q_\varepsilon(m, s) := \#\{j : \lambda_j^{(m)} \notin D(s, \varepsilon)\} = O(1), \quad m \to \infty.$$

If every $\mathbf{A}_m$ has only real eigenvalues (at least for large m) then we may assume that s is real and that the disc $D(s, \varepsilon)$ is the interval $(s - \varepsilon, s + \varepsilon)$. We replace the term "strongly" by "weakly", if $q_\varepsilon(m, s) = o(m)$, when $m \to \infty$. Similar definitions hold if we replace the term eigenvalues with singular values.

A sufficient condition under which a sequence of matrices is strongly clustered is given in the following theorem.

Theorem 1 Serra Capizzano, Bertaccini, and Golub (2005, Theorem 1.2). *Let $\{\mathbf{A}_m\}$ be a sequence of matrices of strictly increasing dimension ($\mathbf{A}_m \in \mathbb{C}^{m\times m}$) with eigenvalues $|\lambda_1^{(m)}| \geqslant |\lambda_2^{(m)}| \geqslant \cdots \geqslant |\lambda_m^{(m)}|$ and singular values $\sigma_1^{(m)} \geqslant \sigma_2^{(m)} \geqslant \cdots \geqslant \sigma_m^{(m)}$. If*

- *there exist a number $N > 0$, independent of m, such that $\sigma_1^{(m)} = \|\mathbf{A}_m\|_2 \leqslant N$, that is $\{\mathbf{A}_m\}$ is a sequence uniformly bounded;*
- *the sequence $\{\mathbf{A}_m\}$ is strongly clustered at 0 in the singular value sense, that is, following Definition 1, $\forall \epsilon > 0 \exists C = C_\epsilon$ independent of m such that $\#\{j : \sigma_j^{(m)} > \epsilon\} \leqslant C_\epsilon$, uniformly $\forall m$;*

then $\{\mathbf{A}_m\}$ is strongly clustered at 0 in the eigenvalue sense, that is $\forall \epsilon > 0\ \exists \widehat{C} = \widehat{C}_\epsilon$ independent of m such that $\#\{j : |\lambda_j^{(m)}| > \epsilon\} \leqslant \widehat{C}_\epsilon$, uniformly $\forall m$.

The following lemma gives us a sufficient condition under which a sequence of matrices is strongly clustered at zero.

Lemma 2. *Let $\{\mathbf{A}_m\}$ be a sequence of matrices ($\mathbf{A}_m \in \mathbb{C}^{m\times m}$), if $\exists N > 0$, independent of m, such that $\|\mathbf{A}_m\|_F \leqslant N$, where $\|\cdot\|_F$ is the Frobenius norm, then the sequence $\{\mathbf{A}_m\}$ is strongly clustered at zero in the singular value and eigenvalue sense.*

Proof 1. For the singular value cluster see (Serra Capizzano, 2001, Section 4, Corollary 4.1, point 2, with B_n=0). For the cluster of the eigenvalues, if $\sigma_1^{(m)} \geqslant \sigma_2^{(m)} \geqslant \cdots \geqslant \sigma_m^{(m)}$ are the singular values of $\mathbf{A}_m$ and $\sigma^m = [\sigma_1^{(m)}, \sigma_2^{(m)}, \dots, \sigma_m^{(m)}]$, we observe that $N \geqslant \|\mathbf{A}_m\|_F = \|\sigma^{(m)}\|_2 \geqslant \|\sigma^{(m)}\|_\infty = \sigma_1^{(m)} = \|\mathbf{A}_m\|_2$, then we are under the hypotheses of Theorem 1 and we can conclude that the sequence $\{\mathbf{A}_m\}$ is strongly clustered at zero in the sense of the eigenvalues. □

We conclude this section with our main result on the clustering of sequences of matrices that arise from discretization of integrals of functions in two variables.

Theorem 3. *Let us define $\mathbf{A}_m = \mathbf{K}_m \mathbf{D}_m$, where $\mathbf{K}_m$ is the sampling matrix of a continuous function $k, k(\cdot,\cdot) : \Omega \times \Omega \to \mathbb{R}$, $\Omega \subset \mathbb{R}^d$, $d \geqslant 1$, Ω closed and bounded, and $\mathbf{D}_m = \mathrm{diag}(w_0, \dots, w_{m-1})$ is the diagonal matrix with the weights of the quadrature formula w_i. Then the sequence $\{\mathbf{A}_m\}$ is strongly clustered at zero in the singular value and eigenvalue sense.*

Proof 2. We consider the Frobenius norm of the matrix $\mathbf{A}_m$:

$$\|\mathbf{A}_m\|_F^2 = \sum_{i,j=0}^{m} (\mathbf{A}_m)_{i,j}^2 = \sum_{i,j=0}^{m} k^2(ih, jh) w_j^2 = \int_{\Omega^2} k^2(x,y)dxdy + \epsilon_m \leqslant C,$$

where C is a constant which depends on Ω and the smoothness of the function k, and ϵ_m is the error of the quadrature formula which approaches to 0 as m increases. The application of Lemma 2 concludes the proof. □

In conclusion, we observe that the above theorem can be applied to the sampling matrix $\mathbf{H}_m$ in (7). Thus we have proved that our iteration matrix is strongly clustered at zero.

B.2. Bidiagonalization

Formula (7) implies that the option price $\mathbf{W}_0$ can be obtained by performing N times the matrix–vector multiplication $\mathbf{HDv}$, starting from the payoff vector $\mathbf{W}_N$. In order to speed-up the matrix–vector product, we use the spectral properties of the matrix $\mathbf{HD}$ as discussed above. These properties allow us to factorize the iteration matrix into the product of "simpler" matrices. To this end, we consider the following theorem.

Theorem 4 Golub and Kahan (1965, Theorem 1). *Let* $\mathbf{A}$ *be any* $m \times m$ *matrix with complex elements. Then* $\mathbf{A}$ *can be decomposed as* $\mathbf{A} = \mathbf{PJQ}^H$ *where* $\mathbf{P}$ *and* $\mathbf{Q}$ *are unitary matrices and* $\mathbf{J}$ *is an* $m \times m$ *bidiagonal matrix of the form*

$$\mathbf{J} = \begin{pmatrix} \alpha_1 & \beta_1 & 0 & \cdots & 0 \\ 0 & \alpha_2 & \beta_2 & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & 0 \\ \vdots & \ddots & \ddots & \ddots & \beta_{m-1} \\ 0 & \cdots & \cdots & 0 & \alpha_m \end{pmatrix}.$$

Matrices $\mathbf{P}$ and $\mathbf{Q}$ are obtained as products of Householder's elementary (rank 1) matrices, i.e., matrices of the form $\mathbf{I} - 2\mathbf{xx}^H$, $\mathbf{x} \in \mathbb{C}^m$, $\mathbf{x}^H\mathbf{x} = 1$. Moreover, the matrix–vector product with Householder matrix requires only $2m$ multiplicative and $2m-1$ additive operations. This means that it is not necessary to calculate explicitly the matrices $\mathbf{P}$ and $\mathbf{Q}$, but we can simply store the vectors of the Householder's matrices that generate them. If, for example, $\mathbf{P} = (\mathbf{I} - 2\mathbf{x}_1\mathbf{x}_1^H)(\mathbf{I} - 2\mathbf{x}_2\mathbf{x}_2^H)\cdots(\mathbf{I} - 2\mathbf{x}_m\mathbf{x}_m^H)$, $\mathbf{x}_i \in \mathbb{C}^m$ it is sufficient to store the vectors $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_m$ to get all the informations necessary to calculate the matrix–vector product $\mathbf{Pv}$.

Using Theorem 4 we can factorize the matrix $\mathbf{HD}$ as $\mathbf{PJQ}^H$. In general, this algorithm is very expensive requiring $O(m^3)$ operations (see Golub & VanLoan, 1996). However, using the cluster property of the matrix $\mathbf{HD}$, it is possible to see experimentally that the elements $(\alpha_1, \ldots, \alpha_m)$ on the main diagonal and those $(\beta_1, \ldots, \beta_{m-1})$ on the upper diagonal of the matrix $\mathbf{J}$, exhibit, in modulus, the behavior shown in Fig. B.4. Therefore, if we consider to be non-zero only the k_ϵ elements above a certain threshold ϵ, the computation of the matrices $\mathbf{P}, \mathbf{J}$ and $\mathbf{Q}$ can stop at the k_ϵth iteration with $O(k_\epsilon m^2)$ operations. The value of k_ϵ is closely related to the fixed tolerance ϵ and to the cluster of the matrix. As a rule of thumb, if the number of eigenvalues greater than a tolerance ϵ is r_ϵ, then the number of steps k_ϵ of the algorithm is only slightly larger than r_ϵ.

Given the matrix $\mathbf{HD}$ and using the algorithm of Golub and Kahan (1965), we compute the vectors $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_{k_\epsilon}$, $\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_{k_\epsilon}$ and the elements $\alpha_1, \ldots, \alpha_{k_\epsilon}$, and $\beta_1, \ldots, \beta_{k_\epsilon - 1}$, so that

$$\mathbf{P}_{k_\epsilon} = (\mathbf{I} - 2\mathbf{x}_1\mathbf{x}_1^H)(\mathbf{I} - 2\mathbf{x}_2\mathbf{x}_2^H)\cdots(\mathbf{I} - 2\mathbf{x}_{k_\epsilon}\mathbf{x}_{k_\epsilon}^H),$$
$$\mathbf{Q}_{k_\epsilon} = (\mathbf{I} - 2\mathbf{y}_1\mathbf{y}_1^H)(\mathbf{I} - 2\mathbf{y}_2\mathbf{y}_2^H)\cdots(\mathbf{I} - 2\mathbf{y}_{k_\epsilon}\mathbf{y}_{k_\epsilon}^H),$$

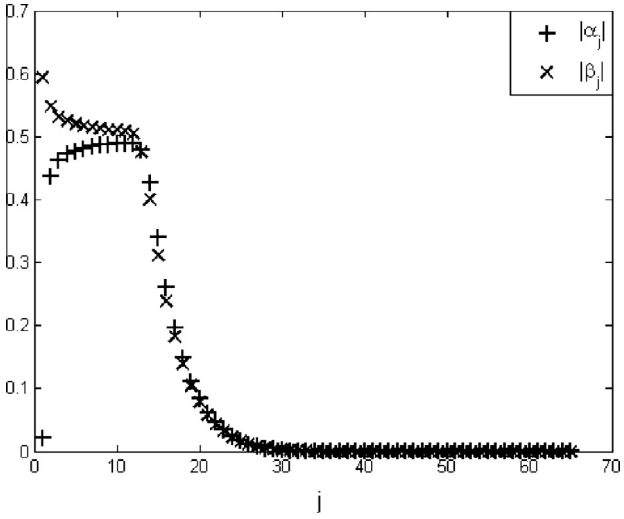

Fig. B.4. Trend of the absolute values of α_j on the main diagonal and β_j on the upper diagonal of the matrix $\mathbf{J}$ obtained from the discretization of the iteration matrix $\mathbf{HD}$ for a double barrier option in the lognormal model with $N = 252, m = 1000$. Parameters setting as in Section 4. In this case $\epsilon = 10^{-8}$, $r_\epsilon = 64$ and $j \leqslant k_\epsilon = 65$.

and

$$\mathbf{P}_{k_\epsilon} \begin{pmatrix} \alpha_1 & \beta_1 & 0 & \cdots & 0 \\ 0 & \ddots & \ddots & & \vdots \\ \vdots & & \ddots & \beta_{k_\epsilon - 1} & \vdots \\ \vdots & & & \alpha_{k_\epsilon} & 0 \\ 0 & \cdots & \cdots & \cdots & 0 \end{pmatrix} \mathbf{Q}_{k_\epsilon}^H = \mathbf{P}_{k_\epsilon}\mathbf{J}_{k_\epsilon}\mathbf{Q}_{k_\epsilon}^H \cong \mathbf{HD}.$$

Using this factorization, we compute $\mathbf{P}_{k_\epsilon}\mathbf{J}_{k_\epsilon}\mathbf{Q}_{k_\epsilon}^H\mathbf{v}$, instead of $\mathbf{HDv}$, $\mathbf{v} \in \mathbb{C}^m$. In addition, exploiting the Householder structure of matrices $\mathbf{P}_{k_\epsilon}$ and $\mathbf{Q}_{k_\epsilon}$ we can reduce the number of operations in the matrix–vector product from $O(m^2)$ to $O(k_\epsilon m)$.

Appendix C. Supplementary material

Supplementary data associated with this article can be found, in the online version, at http://dx.doi.org/10.1016/j.ejor.2013.12.009.

References

Alexander, D. R., Mengija, M., & Stent, A. F. (2012). Arithmetic Brownian motion and real options. *European Journal of Operational Research, 219*, 114–122.
Andricopoulos, A. D., Widdicks, M., Duck, P. W., & Newton, D. P. (2003). Universal option valuation using quadrature methods. *Journal of Financial Economics, 67*, 447–471.
Atkinson, K. E. (2009). *The numerical solution of integral equations of the second kind*. Cambridge University Press.
Becker, C., & Wystup, U. (2009). On the cost of delayed currency fixing announcements. *Annals of Finance, 5*(2), 161–174.
Bhat, H. S., & Kumar, N. (2012). Option pricing under a normal mixture distribution derived from the Markov tree model. *European Journal of Operational Research, 223*, 744–762.
Bouzoubaa, B., & Osseiran, A. (2010). *Exotic options and hybrids: A guide to structuring, pricing and trading*. Wiley Finance.
Boyle, P. P., & Tian, Y. S. (1999). Pricing lookback and barrier options under the CEV process. *The Journal of Financial and Quantitative Analysis, 34*(2), 241–264.
Boyle, P. P., Tian, Y. S., & Imai, J. (1999). Lookback options under the CEV process: A correction. *The Journal of Financial and Quantitative Analysis*. Unpublished Appendixes, Notes, Comments, and Corrections.
Breton, M., & de Frutos, J. (2010). Option pricing under GARCH processes using PDE methods. *Operations Research, 58*, 1148–1157.
Broadie, M., Glasserman, P., & Kou, S. G. (1997). A continuity correction for the discrete barrier options. *Mathematical Finance, 7*, 325–349.

Cai, N., Chen, N., & Wan, X. (2010). Occupation times of jump-diffusion processes with double exponential jumps and the pricing of options. *Mathematics of Operations Research, 35*, 412–437.

Castagna, A. (2009). The hedging costs of discrete monitoring of FX barrier options. SSRN: http://ssrn.com/abstract=1335302.

Černý, A., & Kyriakou, I. (2011). An improved convolution algorithm for discretely sampled Asian options. *Quantitative Finance*, 381–389.

Costabile, M. (2006). On pricing lookback options under the CEV process. *Decisions in Economics and Finance, 29*, 139–153.

Cox, J. (1996). The constant elasticity of variance option pricing model. *Journal of Portfolio Management, 23*, 15–17.

Cox, J., & Ross, S. (1976). The valuation of options for alternative stochastic processes. *Journal of Financial Economics, 3*, 145–166.

Davydov, D., & Linetsky, V. (2001). Pricing and hedging path-dependent options under the CEV process. *Management Science, 47*(7), 949–965.

Fusai, G., Marazzina, D., Marena, M., & Ng, M. (2012). Z-transform and preconditioning techniques for option pricing. *Quantitative Finance, 12*(9), 1381–1394.

Fusai, G., Marazzina, D., & Marena, M. (2011). Pricing discretely monitored Asian options by maturity randomization. *SIAM Journal on Financial Mathematics, 2*, 383–403.

Fusai, G., & Recchioni, M. C. (2007). Analysis of quadrature methods for pricing discrete barrier options. *Journal of Economic Dynamics and Control, 31*, 826–860.

Fusai, G., Abrahams, D., & Sgarra, C. (2006). An exact analytical solution of discrete barrier options. *Finance and Stochastics, 10*, 1–26.

Golub, G. H., & Kahan, W. (1965). Calculating the singular values and pseudo-inverse of a matrix. *SIAM Journal on Numerical Analysis, 2*, 205–224.

Golub, G. H., & VanLoan, C. F. (1996). *Matrix computations* (3rd ed.). Baltimore, Maryland: The Johns Hopkins University Press.

Jin, X., Li, X., Tan, H. H., & Wu, Z. (2013). A computationally efficient state-space partitioning approach to pricing high-dimensional American options via dimension reduction. *European Journal of Operational Research, 231*(2), 362–370.

Lin, J., & Palmer, K. (2013). Convergence of barrier option prices in the binomial model. *Mathematical Finance, 23*(2), 318–338.

Linetsky, V. (1999). Step options. *Mathematical Finance, 9*, 55–96.

Lord, R., Fang, F., Bervoets, F., & Oosterlee, C. W. (2008). A fast and accurate FFT-based method for pricing early-exercise options under Lévy processes. *SIAM Journal on Scientific Computing, 30*, 1678–1705.

Moon, K.-S., & Kim, H. (2013). An adaptive averaging binomial method for option valuation. *Operations Research Letters*, 511–515.

Norris, J. R. (1997). *Markov chains*. Cambridge Series in Statistical and Probabilistic Mathematics, Cambridge University Press.

Nunes, J. P. V. (2009). Pricing American options under the constant elasticity of variance model and subject to bankruptcy. *Journal of Financial and Quantitative Analysis, 44*, 1231–1263.

Peng, B. (2006). Pricing geometric Asian option under the CEV process. *International Economic Journal, 20*(4), 515–522.

Quarteroni, A., Sacco, R., & Saleri, F. (2000). *Numerical mathematics*. New York: Springer-Verlag.

Rambeerich, N., Tangman, D. Y., Lollchund, M. R., & Bhuruth, M. (2013). High-order computational methods for option valuation under multifactor models. *European Journal of Operational Research, 224*, 219–226.

Serra Capizzano, S., Bertaccini, D., & Golub, G. H. (2005). How to deduce a proper eigenvalue cluster from a proper singular value cluster in the nonnormal case. *SIAM Journal on Matrix Analysis and Applications, 27*(1), 82–86.

Serra Capizzano, S. (2001). Spectral behavior of matrix sequences and discretized boundary value problems. *Linear Algebra and Its Applications, 337*, 37–78.

Schroder, M. (1989). Computing the constant elasticity of variance option pricing formula. *The Journal of Finance, 44*(1), 211–219.

Tyrtyshnikov, E. E. (1996). A unifying approach to some old and new theorems on distribution and clustering. *Linear Algebra and Its Applications, 232*, 1–43.