# A non linear approximation method for solving high dimensional partial differential equations: Application in Finance.

José Infante Acevedo and Tony Lelièvre
Université Paris-Est, CERMICS,
Ecole des Ponts, 6-8 avenue Blaise Pascal, 77455 Marne-la-vallée, France
jose-infante.acevedo@cermics.enpc.fr, lelievre@cermics.enpc.fr

February 18, 2016

**Abstract:** We present the problem of approximating a square-integrable function by a sum of tensor products using a greedy algorithm. As an example of this approximation by a sum of tensor products, we study the pricing of European basket options. In numerical experiments, we obtain results for up to 10 underlyings; that means the dimension $d$ is equal to 10. This greedy algorithm thus enables to treat problems in higher dimension than the sparse tensor product like Pommier in [6].
**Acknowledgements:** José Infante Acevedo is grateful to AXA Research Fund for his doctoral fellowship.

# Premia 18

## Introduction

Many problems of interest for various applications (for example material sciences and finance) involve high-dimensional partial differential equations (PDEs). The typical example in finance is the pricing of a basket option, which can be obtained by solving the Black-Scholes PDE with as dimension the number of underlying assets.

We propose to investigate an algorithm which has been recently proposed by Chinesta *et al.* [1] for solving high-dimensional Fokker-Planck equations in the context of kinetic models for polymers, and by Nouy *et al.* [5] in uncertainty quantification framework based on previous works by Ladevèze [3]. This approach is also studied in [4] to try to circumvent the curse of dimensionality in problems such as the Poisson problem. This approach is a nonlinear approximation method that we will called below the *greedy algorithm* because it is related to the so-called greedy algorithms introduced in nonlinear approximation theory by Temlyakov in [8]. The principal idea is to represent the solution as a sum of tensor products:

$$\begin{aligned} u(x_1,\dots,x_d) &= \sum_{k\geq 1} r_k^1(x_1) r_k^2(x_2)\dots r_k^d(x_d) \\ &= \sum_{k\geq 1} \left(r_k^1 \otimes r_k^2 \otimes \dots \otimes r_k^d\right)(x_1,\dots,x_d) \end{aligned} \tag{1}$$

and to compute iteratively each term of this sum using a greedy algorithm. This greedy algorithm can be applied to any PDE which admits a variational interpretation as a minimization problem. The practical interest of the greedy algorithm in various contexts has been demonstrated (see for example [1] for applications in fluid mechanics).

In this paper, as a simple illustration, we explain how the greedy algorithm can be used to approximate a square-integrable function by a sum of tensor products. This yields a simple technique to price a vanilla basket option of European type for example. This algorithm also applies to approximating the solution to high-dimensional partial differential equations, and in particular the Black-Scholes partial differential equation. We refer to [2] for more details.

We will study also the practical implementation of the greedy algorithm. We will not solve the minimization problem associated, but the first-order optimality conditions of this minimization problem, namely the Euler equation. This leads to a system of equations where the number of degrees of freedom does not grow exponentially with respect to the dimension, and this fact will be very important in order to attain high-dimensional frameworks in practical applications. However, the algorithmic complexity of the approach is problematic, given that a system of $d$ nonlinear equations has to be solved, where $d$ is the dimension considered.

Other deterministic techniques have been applied to price European options in a high-dimensional framework. Classical methods such as finite differences and finite elements are limited in their application when the dimension increases (typically $d = 4$), because the number of degrees of freedom increases exponentially with respect to the dimension, as the memory of space for storage is limited. Financial applications of the sparse tensor product methods have been studied by Pommier in [6]. These sparse methods also use the representation of the solution as a sum of tensor products, and assume that the solution is regular enough to obviate fine discretizations in each direction. In practice, this method may be difficult to apply for reasons such as the regularity of the solution and the mesh adaptation.

In numerical experiences, we obtain results for up to 10 underlyings; that means the dimension $d$ is equal to 10. This is higher than a result obtained using, for example, the sparse tensor product method like Pommier in [6] where $d = 5$.

The plan of this document is the following. In Section 1, we introduce the general setting for the definition of the greedy algorithms and we give some theoretical results that have been proved in the literature and that assure the convergence of the approach. The practical implementation of the greedy algorithm in the case of the approximation of a function by a sum of tensor products is discussed in Section 2.1. Results and applications on the approximation of a basket put option are presented in Sections 2.2 and 2.3.

# 1 Greedy algorithms for high dimensional problems

In this section, we define a general framework for the greedy algorithm that we will use to solve the high-dimensional problems studied in this paper.

The bottom line of deterministic approaches for high-dimensional problems is to represent the solutions as linear combinations of tensor products of one-dimensional functions as in (1). If the number of terms in the expansion remains small, this enables us to represent the solution, avoiding the curse of dimensionality.

The greedy algorithm proposed in [1, 4, 5] is based on two important points. The first one is to recast the original problem (in our case, this is the option pricing problem) as a minimization problem:

$$u = \underset{v \in V}{\operatorname{argmin}}\, \mathcal{E}(v), \tag{2}$$

where $\mathcal{E} : V \mapsto \mathbb{R}$ is functional with a unique global minimizer $u \in V$ with $V$ a Hilbert space. For example, $V = L^2(\mathcal{X}^d)$ with $\mathcal{X}$ a bounded one-dimensional domain and $d$ large.

The second point is to look iteratively for the best tensor product in the expansion of the solution as a sum of tensor products of lower-dimensional functions

$$u_n(x_1, x_2, \ldots, x_d) = \sum_{k=1}^{n} r_k^1 \otimes r_k^2 \ldots \otimes r_k^d(x_1, \ldots, x_d) \tag{3}$$

where for all $i = 1, \ldots, d$ and $k = 1, \ldots, n$, the functions $r_k^i \in V_{x_i}$, with $V_{x_i}$ Hilbert spaces depending on the low-dimensional variable $x_i$. This sequential search for the terms in the sum (3) is related to the so-called

greedy algorithms introduced in the nonlinear approximation theory by Temlyakov in [8] and by De Vore and Temlyakov in [7].

In what follows, we assume that $V, V_{x_1}, V_{x_2}, \ldots, V_{x_d}$ are Hilbert spaces such that

(H1) $\text{Vect}\{r^1 \otimes r^2 \otimes, \ldots \otimes r^d, r^1 \in V_{x_1}, r^2 \in V_{x_2}, \ldots, r^d \in V_{x_d}\} \subset V$ is dense.

To compute $u_n$ in the separated form (3), $u_n$ being the approximation of $u$ that is the solution of the problem (2), we define the greedy algorithm as follows:

Iterating for all $n \geq 1$:

$$(r_n^1, r_n^2, \ldots, r_n^d) \in \underset{r^1 \in V_{x_1}, r^2 \in V_{x_2}, \ldots, r^d \in V_{x_d}}{\operatorname{argmin}} \mathcal{E}\left(\sum_{k=1}^{n-1} r_k^1 \otimes r_k^2 \otimes \ldots \otimes r_k^d + r^1 \otimes r^2 \otimes \ldots \otimes r^d\right). \tag{4}$$

The following result given in [4] ensures the convergence of the greedy algorithm defined in (4) for the case where the functional $\mathcal{E}$ has the following form:

$$\mathcal{E}(v) = \|u - v\|_V^2. \tag{5}$$

**Theorem 1.1.** *Let us assume that the assumption (H1) holds and that the functional $\mathcal{E}$ has the form (5). Then,*

$$\|u_n - u\|_V \underset{n \to \infty}{\longrightarrow} 0. \tag{6}$$

An estimate of error is also proposed in [4]. To obtain this result, we need to introduce the functional space adapted to the convergence analysis

$$\mathcal{A}^1 = \left\{ u = \sum_{k=1}^{+\infty} r_k^1 \otimes r_k^2 \otimes \ldots \otimes r_k^d, \text{ where } r_k^i \in V_{x_i}, i = 1, \ldots, d, \sum_{k=1}^{+\infty} \|r_k^1 \otimes r_k^2 \ldots \otimes r_k^d\|_V < +\infty \right\} \tag{7}$$

and the associated norm is

$$\|u\|_{\mathcal{A}^1} = \inf \left\{ \sum_{k=1}^{+\infty} \|r_k^1 \otimes r_k^2 \ldots \otimes r_k^d\|_V \mid u = \sum_{k=1}^{+\infty} r_k^1 \otimes r_k^2 \otimes \ldots \otimes r_k^d \right\} \tag{8}$$

**Theorem 1.2.** *Let us assume that the assumption (H1) is verified and that the functional $\mathcal{E}$ has the form (5). Then, for a function $u \in \mathcal{A}^1$, there exists a constant $C > 0$ such that*

$$\|u_n - u\|_V \leq C n^{-1/6}, \tag{9}$$

*for all $n \in \mathbb{N}^*$.*

We note that the convergence rate factor of $\frac{1}{6}$ can be improved to $\frac{11}{62}$ and that the constant $C$ depends on the norm $\|u\|_{\mathcal{A}^1}$. See [4, 7].

Our work relies on these theoretical results because in the setting that we plan to analyze in this paper, we will consider a functional $\mathcal{E}$ such that $\mathcal{E}(v) = \|u - v\|_V^2$.

## 2 Implementation of the algorithm in the case of the approximation of a square-integrable function

In this section, we will discuss the implementation of the algorithm defined by (4) in the case of the approximation of a function $f$ by a sum of tensor products. We will then provide numerical examples of the application of this approach. This particular case has the advantage of being an easy example to help understanding the implementation of the greedy algorithm.

## 2.1 Greedy algorithm for the approximation of a square-integrable function

In order to show the implementation that we use for the algorithm (4), let us present the simple problem of approximating a square-integrable function $f$ by a sum of tensor products.

Mathematically, we consider the spaces $V = L^2(\Omega_1 \times \Omega_2 \times \ldots \times \Omega_d)$, $V_{x_i} = L^2(\Omega_i)$ for $i = 1, \ldots, d$, where $\Omega_i \subset \mathbb{R}$ is a bounded domain for $i$ such that $1 \leq i \leq d$. We recall that we are looking for a separate representation $f = \sum_{k \geq 1} r_k^1 \otimes r_k^2 \otimes \ldots \otimes r_k^d$. So, let us consider the following minimization problem:

$$\text{Find } u \in L^2(\Omega_1 \times \Omega_2 \times \ldots \times \Omega_d) \text{ such that } u = \arg\min_{u \in L^2} \left( \frac{1}{2} \int_{\Omega_1 \times \Omega_2 \times \ldots \times \Omega_d} u^2 - \int_{\Omega_1 \times \Omega_2 \times \ldots \times \Omega_d} uf \right) \tag{10}$$

where the solution is $u = f$. In this context, the greedy algorithm (4) can be rewritten as follows:

Iterate for all $n \geq 1$: Find $(r_n^1, r_n^2, \ldots, r_n^d) \in V_{x_1} \times V_{x_2} \times \ldots \times V_{x_d}$ such that $(r_n^1, r_n^2, \ldots, r_n^d)$ belongs to

$$\begin{aligned} \underset{r^1 \in L^2(\Omega_1), \ldots, r^d \in L^2(\Omega_d)}{\operatorname{argmin}} \; & \frac{1}{2} \int_{\Omega_1 \times \Omega_2 \times \ldots \times \Omega_d} \left| \sum_{k=1}^{n-1} r_k^1 \otimes r_k^2 \otimes \ldots \otimes r_k^d + r^1 \otimes r^2 \otimes \ldots \otimes r^d \right|^2 \\ & - \int_{\Omega_1 \times \Omega_2 \times \ldots \times \Omega_d} \left( \sum_{k=1}^{n-1} r_k^1 \otimes r_k^2 \otimes \ldots r_k^d + r^1 \otimes r^2 \otimes \ldots \otimes r^d \right) f, \end{aligned} \tag{11}$$

As proposed in [4] instead of solving the problem (11), we will determine the solutions of the Euler equation for (11). It has to be remarked that, in general, the solutions of the Euler equation are not necessarily the solutions of the minimization problem, given the nonlinearity of the tensor product space $L^2(\Omega_1) \otimes L^2(\Omega_2) \otimes \ldots \otimes L^2(\Omega_d)$.

The Euler equation for (11) has the following form:

Find $(r_n^1, r_n^2, \ldots, r_n^d) \in L^2(\Omega_1) \times L^2(\Omega_2) \times \ldots \times L^2(\Omega_d)$ such that for any functions $(r^1, r^2, \ldots, r^d) \in L^2(\Omega_1) \times L^2(\Omega_2) \times L^2(\Omega_d)$

$$\begin{aligned} & \int_{\Omega_1 \times \Omega_2 \times \ldots \times \Omega_d} (r_n^1 \otimes r_n^2 \otimes \ldots \otimes r_n^d) \left( r^1 \otimes r_n^2 \otimes \ldots \otimes r_n^d + r_n^1 \otimes r^2 \otimes \ldots \otimes r_n^d + \ldots + r_n^1 \otimes r_n^2 \otimes \ldots \otimes r^d \right) \\ & = \int_{\Omega_1 \times \Omega_2 \times \ldots \times \Omega_d} f_{n-1} \left( r^1 \otimes r_n^2 \otimes \ldots \otimes r_n^d + r_n^1 \otimes r^2 \otimes \ldots \otimes r_n^d + \ldots + r_n^1 \otimes r_n^2 \otimes \ldots \otimes r^d \right) \end{aligned} \tag{12}$$

where $f_{n-1} = f - \sum_{k=1}^{n-1} r_k^1 \otimes r_k^2 \otimes \ldots \otimes r_k^d$.

Equation (12) can be written equivalently as

$$\begin{cases} \left( \int_{\Omega_2} |r_n^2|^2 \right) \left( \int_{\Omega_3} |r_n^3|^2 \right) \ldots \left( \int_{\Omega_d} |r_n^d|^2 \right) \; r_n^1 = \int_{\Omega_2 \times \Omega_3 \times \ldots \times \Omega_d} \left( r_n^2 \otimes \ldots \otimes r_n^d \right) \; f_{n-1}, \\ \left( \int_{\Omega_1} |r_n^1|^2 \right) \left( \int_{\Omega_3} |r_n^3|^2 \right) \left( \int_{\Omega_4} |r_n^4|^2 \right) \ldots \left( \int_{\Omega_d} |r_n^d|^2 \right) \; r_n^2 = \int_{\Omega_1 \times \Omega_3 \times \Omega_4 \ldots \times \Omega_d} \left( r_n^1 \otimes r_n^3 \otimes r_n^4 \otimes \ldots \otimes r_n^d \right) \; f_{n-1}, \\ \quad \vdots \\ \left( \int_{\Omega_1} |r_n^1|^2 \right) \left( \int_{\Omega_2} |r_n^2|^2 \right) \left( \int_{\Omega_3} |r_n^3|^2 \right) \ldots \left( \int_{\Omega_{d-1}} |r_n^{d-1}|^2 \right) \; r_n^d = \int_{\Omega_1 \times \Omega_2 \times \Omega_3 \ldots \times \Omega_{d-1}} \left( r_n^1 \otimes r_n^2 \otimes r_n^3 \otimes \ldots \otimes r_n^{d-1} \right) \; f_{n-1}. \end{cases} \tag{13}$$

The system (13) is a non linear coupled system of equations which can be solved by a fixed point procedure as proposed in [1].

Choose $(r_n^{1,(0)}, r_n^{2,(0)}, \ldots, r^{d,(0)}) \in L^2(\Omega_1) \times L^2(\Omega_2) \times \ldots \times L^2(\Omega_d)$, and at iteration $k \geq 0$, compute $(r_n^{1,(k)}, r_n^{2,(k)}, \ldots, r^{d,(k)}) \in L^2(\Omega_1) \times L^2(\Omega_2) \times \ldots \times L^2(\Omega_d)$ which is the solution to

$$\begin{cases} \|r_n^{2,(k)}\|^2_{L^2(\Omega_2)}\|r_n^{3,(k)}\|^2_{L^2(\Omega_3)}\cdots\|r_n^{d,(k)}\|^2_{L^2(\Omega_d)}r_n^{1,(k+1)} = \int_{\Omega_2\times\Omega_3\times\ldots\times\Omega_d}\left(r_n^{2,(k)}\otimes\ldots\otimes r_n^{d,(k)}\right) f_{n-1}, \\ \|r_n^{1,(k+1)}\|^2_{L^2(\Omega_1)}\|r_n^{3,(k)}\|^2_{L^2(\Omega_3)}\|r_n^{4,(k)}\|^2_{L^2(\Omega_4)}\cdots\|r_n^{d,(k)}\|^2_{L^2(\Omega_d)}r_n^{2,(k+1)} = \int_{\Omega_1\times\Omega_3\times\Omega_4\ldots\times\Omega_d}\left(r_n^{1,(k+1)}\otimes r_n^{3,(k)}\otimes r_n^{4,(k)}\otimes\ldots\otimes r_n^{d,(k)}\right) f_{n-1}, \\ \\ \|r_n^{1,(k+1)}\|^2_{L^2(\Omega_1)}\|r_n^{2,(k+1)}\|^2_{L^2(\Omega_2)}\|r_n^{3,(k+1)}\|^2_{L^2(\Omega_3)}\cdots\|r_n^{d-1,(k+1)}\|^2_{L^2(\Omega_{d-1})}r_n^{d,(k+1)} = \int_{\hat{\Omega}_1}\left(r_n^{1,(k+1)}\otimes r_n^{2,(k+1)}\otimes r_n^{3,(k+1)}\otimes\ldots\otimes r_n^{d-1,(k+1)}\right) \end{cases} \tag{14}$$

until convergence is reached when discretized by standard full tensor product techniques.

An important point to note is that we start with a linear problem (10) with exponential complexity with respect to the dimension, and at the end, we obtain a nonlinear problem (13) with at each iteration linear complexity with respect to the dimension.

In the two-dimensional case, the algorithm given by (11) is related to the Singular Value Decomposition (or rank one decomposition), as it is explained in [4]. In this case, the solutions of the variational problem (11) are exactly the solutions to the Euler equation (12) that verify the second order optimality conditions. This property does not hold for a $d$-dimensional framework with $d \geq 3$.

## 2.2 Example of a separated representation of a put payoff

In this section we will apply the algorithm (11) to obtain an approximation of the payoff of a basket put option. For the practical implementation of the greedy algorithm, we need to introduce the space discretization. In practice, the spaces $V_{x_i}^{\Delta x}$ for $i = 1,\ldots,d$ that are used to discretize $L^2(\Omega_i)$ for $i = 1,\ldots d$ are the P1 finite elements on a uniform mesh with space step $\Delta x$. The number $\Delta x = \frac{1}{N}$ is the discretization parameter and $N$ is the number of discretization points. For each $k$, we discretize the functions $r_k^i$ for $i = 1,\ldots,d$ that appear in the approximation of the solution given by the expression (12) as follows:

$$r_k^{i,\Delta x}(x_i) = \sum_{j=0}^{N} r_k^{i,j}\phi_j(x_i), \quad r_k^{i,j} \in \mathbb{R},\ \forall j,k, \tag{15}$$

where $\phi_i(x) = \phi\left(\frac{x-x_i}{\Delta x}\right)$ with $\phi(x) = \begin{cases} 1-|x| & \text{if } |x| \leq 1, \\ 0 & \text{if } |x| > 0. \end{cases}$

This type of discretization and its generalization to the $d$-dimensional case will be used throughout this paper whenever simulations are concerned.

Let us consider now the problem (10) with $f(x_1,\ldots,x_d) = \left(K - \frac{1}{d}\sum_{i=1}^d x_i\right)_+$. Figure 1 shows how the algorithm approximates the basket put payoff in a two-dimensional framework ($d = 2$). We observe that as the number of iterations of the greedy algorithm increases, the approximation of the function $f(x_1,x_2)$ improves.

Figure 2 shows the convergence curves that we obtain for this problem with respect to the dimension. We observe that, as the dimension increases, the number of iterations needed to obtain the convergence increases as well.

We also provide in Table 1 the number of iterations needed in order to obtain a relative error of $10^{-5}$ when we consider 11 points of discretization per dimension ($N = 10$). The relative error calculated is the discrete $L^2$ error

$$e_n = \frac{\sqrt{\frac{1}{N}\sum_{i_1=1}^N\sum_{i_2=1}^N\cdots\sum_{i_d=1}^N\left(f(x_{i_1},x_{i_2},\ldots,x_{i_d}) - u^n(x_{i_1},x_{i_2},\ldots,x_{i_d})\right)^2}}{\sqrt{\frac{1}{N}\sum_{i_1=1}^N\sum_{i_2=1}^N\cdots\sum_{i_d=1}^N f(x_1,\ldots,x_d)^2}} \tag{16}$$

where $u^n(x_1,x_2,\ldots,x_d) = \sum_{k=1}^n r_k^1\otimes r_k^2\otimes\ldots\otimes r_k^d$ is the solution obtained with the greedy algorithm at the iteration $n$.

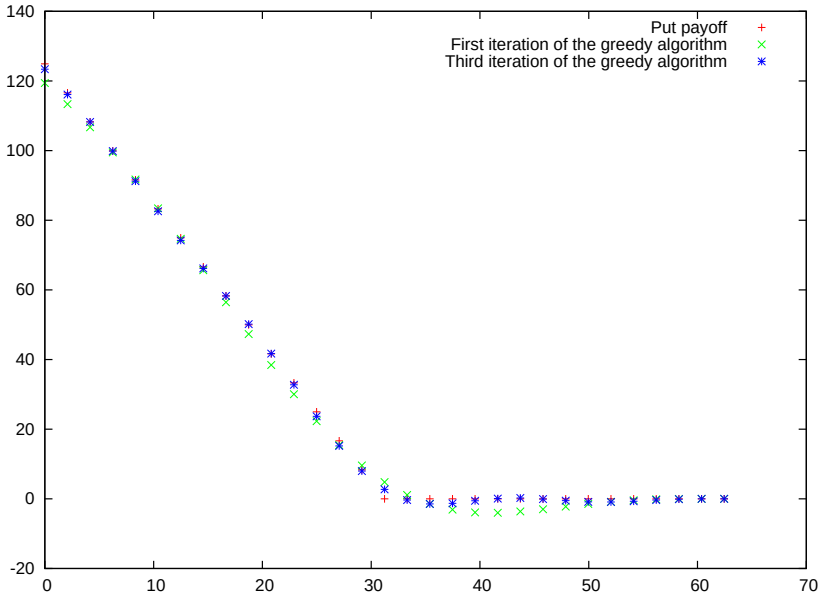


Figure 1: Basket put option with two assets. We have considered here the intersection between the surface of prices and the plane $S_1 = S_2$. To obtain this approximation we take 31 points of discretization per dimension. In this figure, we show the approximation given after the first and third iteration of the algorithm.

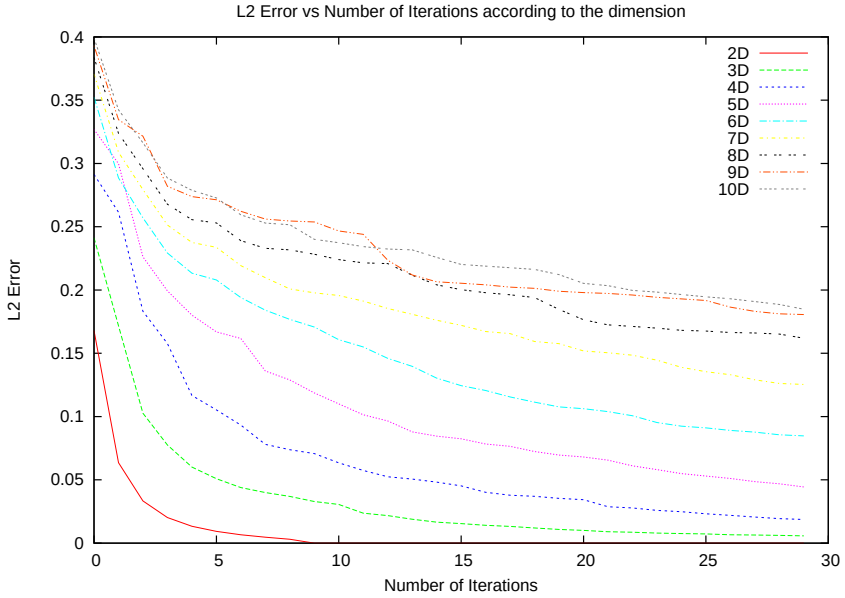


Figure 2: Convergence curves for the approximation of a basket put payoff by a sum of tensor products. We observe that if the dimension increases, then the number of iterations needed to obtain the convergence increases as well. The error calculated is given by (16).

| Dimension | Number of iterations |
|---|---|
| 1 | 1 |
| 2 | 2 |
| 3 | 10 |
| 4 | 22 |
| 5 | 101 |
| 6 | 228 |
| 7 | 1077 |
| 8 | 3974 |

Table 1: Number of iterations needed to obtain a relative error of $10^{-5}$ when we take 11 discretization points per dimension

Notice that the full tensor product approximation would require $11^8 \simeq 2.10^8$ degrees of freedom in an 8-dimensional case to be compared with the $3974 \times 8 \times 11 \simeq 350000$ degrees of freedom that we obtained. For the evaluation of this number, we used the fact that at each iteration of the algorithm we get 8 functions that are determined by 11 discretization points.

In order to reduce the computational time of this calculation, we use the specific form of the payoff function to deduce in a preliminary step the points that belong to the support of this function. Therefore, when we calculate numerically the integral term

$$\int_{\Omega_2 \times \Omega_3 \times \ldots \times \Omega_d} \left(r_n^2 \otimes \ldots \otimes r_n^d\right) \; f_{n-1} \; dx_2 dx_3 \ldots dx_d, \tag{17}$$

in (13), we do not need to pass through the points where the function vanishes. In practice, we have used a backtracking algorithm to describe the support of this payoff. This type of algorithm consists in constructing candidates sequentially and neglecting them when they do not verify the conditions required as a solution, in this case to belong to the support of the payoff. For instance, in a 5-dimensional case, the computational time is reduced by a factor of $\frac{4}{5}$ by taking into account the support of the payoff in the computations of the integral term.

In our numerical experiences, the initial conditions needed to begin the iterations in the fixed point procedure are taken randomly because we get better results in terms of convergence than in the case where we use constant initial conditions.

Concerning the computational time, we note that if the dimension increases, one iteration of the algorithm takes more time to be computed because the number of the equations in the system generated by the Euler equation increases linearly with respect to the dimension. The integral terms of type (17) also demand more time of execution because the domain has a new variable.

## 2.3 Pricing of a basket put using the separated approximation of the payoff

As an example to show that the approximation by a sum of tensor products makes sense, we can use the approximation as a method to obtain prices of options. In order to do this, we must obtain an approximation of $h(x_1(T) \ldots x_d(T)) f_{S_1(T) \ldots S_d(T)}(x_1(T) \ldots x_d(T); t)$ where $h(x_1(T) \ldots x_d(T))$ is the payoff of the option and $f_{S_1(T) \ldots S_d(T)}(x_1(T) \ldots x_d(T); t)$ the joint density of the variables $S_1(T) \ldots S_d(T)$.

So, given the price of a European option

$$\begin{aligned} P_t &= \mathbb{E}\left[e^{-r(T-t)} h(S_1(T) \ldots S_d(T)) | \mathcal{F}_t\right] \\ &= \int_{\Omega_1 \times \ldots \times \Omega_d} e^{-r(T-t)} h(x_1(T) \ldots x_d(T)) f_{S_1(T) \ldots S_d(T)}(x_1(T) \ldots x_d(T); t) dx_1(T) \ldots dx_d(T), \end{aligned}$$

we obtain a separable approximation of $h(x_1(T)\dots x_d(T))f_{S_1(T)\dots S_d(T)}(x_1(T)\dots x_d(T);t)$ using the greedy algorithm and then we can calculate this integral very efficiently using Fubini's rule. In Figure 3, we apply this idea for the case of a basket put option on 7 assets.

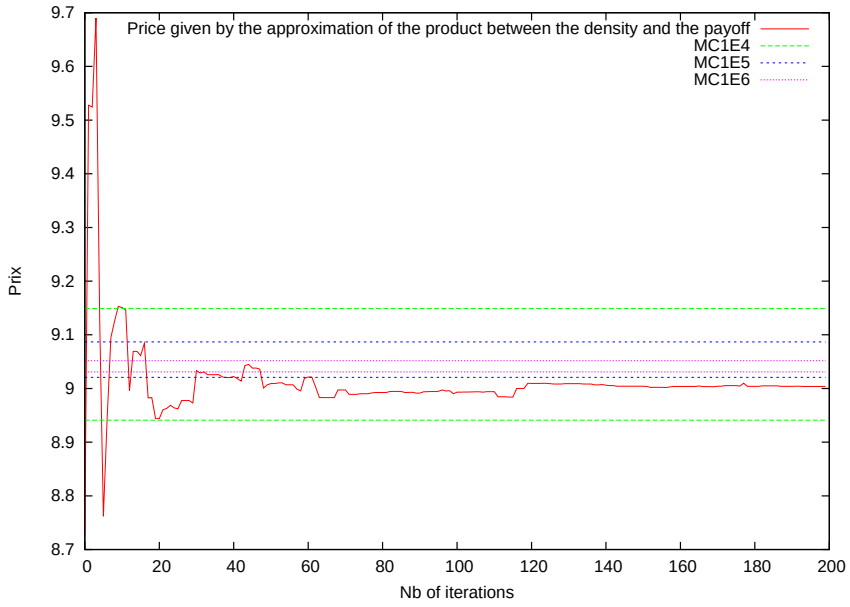


Figure 3: Price of put basket option with 7 assets. The red curve gives us the price of this financial product with respect to the number of iterations of the algorithm. The horizontal lines represent the confidence interval obtained with a Monte Carlo method using respectively $10^4$, $10^5$ and $10^6$ iterations.

## References


[1] A. Ammar, B. Mokdad, F. Chinesta, and R. Keunings. A new family of solvers for some classes of multidimensional partial differential equations encountered in kinetic theory modeling of complex fluids. *J. Non-Newtonian Fluid Mech.*, 139:153–176, 2006. 1, 2, 4

[2] J. Infante Acevedo. *Méthodes et modèles numériques appliqués aux risques du marché et à l'évaluation financière.* PhD thesis, Université de Paris-Est. 2

[3] P. Ladevèze. Nonlinear computational structural mechanics: new approaches and non-incremental methods of calculations. 1999. 1

[4] C. Le Bris, T. Lelièvre, and Y. Maday. Results and questions on a nonlinear approximation approach for solving high-dimensional partial differential equations. *Constructive Approximation*, 30(3):621–651, 2009. 1, 2, 3, 4, 5

[5] A. Nouy. A priori tensor approximations for the numerical solution of high-dimensional problems: alternative definitions. *Preprint*, 2007. 1, 2

[6] D. Pommier. *Méthodes numériques sur des grilles sparse appliquées à l'évaluation d'options en finance.* PhD thesis, Université Pierre et Marie Curie, 2008. 1, 2

[7] DeVore R.A. and Temlyakov V.N. Some remarks on greedy algorithms. *Adv. Comput. Math.*, 5:173–187, 1996. 3

[8] V. N. Temlyakov. Greedy approximation. *j-ACTA-NUMERICA*, 17:235–409, 2008. 1, 3