Weierstraß-Institut
für Angewandte Analysis und Stochastik
Leibniz-Institut im Forschungsverbund Berlin e. V.

Preprint ISSN 2198-5855

Optimal stopping via deeply boosted backward regression

Denis Belomestny[1], John G. M. Schoenmakers[2], Vladimir Spokoiny[2], Yuri Tavyrikov[3]

submitted: August 9, 2018

[1] Duisburg-Essen University
Thea-Leymann-Str. 9
45127 Essen
Germany
E-Mail: denis.belomestny@uni-due.de

[2] Weierstrass Institute
Mohrenstr. 39
10117 Berlin
Germany
E-Mail: john.schoenmakers@wias-berlin.de
vladimir.spokoiny@wias-berlin.de

[3] Higher School of Economics
Moscow
Russia
E-Mail: tavyrikov@gmail.com

No. 2530
Berlin 2018

2010 *Mathematics Subject Classification.* 60G40, 65C05, 62J02.

Key words and phrases. Optimal stopping, nonlinear regression, deep learning.

J. S. and V. S. acknowledge support by Research Center MATHEON "Mathematics for Key Technologies" in Berlin.

Edited by
Weierstraß-Institut für Angewandte Analysis und Stochastik (WIAS)
Leibniz-Institut im Forschungsverbund Berlin e. V.
Mohrenstraße 39
10117 Berlin
Germany

Fax: +49 30 20372-303
E-Mail: preprint@wias-berlin.de
World Wide Web: http://www.wias-berlin.de/

Optimal stopping via deeply boosted backward regression

Denis Belomestny, John G. M. Schoenmakers, Vladimir Spokoiny, Yuri Tavyrikov

Abstract

In this note we propose a new approach towards solving numerically optimal stopping problems via boosted regression based Monte Carlo algorithms. The main idea of the method is to boost standard linear regression algorithms in each backward induction step by adding new basis functions based on previously estimated continuation values. The proposed methodology is illustrated by several numerical examples from finance.

1 Introduction

An optimal stopping problem, in finance virtually synonym with the pricing problem of an American style derivative, can be efficiently solved in low dimensions, for instance by tree methods or using deterministic numerical methods for the corresponding partial differential equation. However, many American options in practice (see e.g. [7]) involve high dimensional underlying processes and this made it necessary to develop Monte Carlo methods for pricing such options. Pricing American derivatives, hence solving optimal stopping problems via Monte Carlo is a challenging task, because this typically requires backward dynamic programming that for long time was thought to be incompatible with forward structure of the Monte Carlo methods. In recent years much research was focused on the development of efficient methods to compute approximations to the value functions or optimal exercise policy. Eminent examples include the functional optimization approach of [1], the mesh method of [4], the regression-based approaches of [5], [8], [9], [6] and [3]. The most popular type of algorithms are with no doubt the regression ones. In fact, in many practical pricing problems, the low-degree polynomials are typically used for regression (see [7]). The resulting least squares problem has a relatively small number of unknown parameters. However, this approach has an important disadvantage - it may exhibit too little flexibility for modeling highly non-linear behaviour of the exercise boundary. Higher-degree polynomials can be used, but they may contain too many parameters and, therefore, either over-fit the Monte Carlo sample or prohibit parameter estimation because the number of parameters is too large. One possible approach for controlling the complexity of a regression model is subset selection. The goal of subset selection is to find a subset, from a fixed full predetermined dictionary of basis functions, that corresponds to a model of the best predictive performance. Before performing the actual subset selection, one must first predefine the dictionary that will provide the basis functions for model generation. This is usually done by setting the maximum degree of a full polynomial and taking the set of its basis functions. By using subset selection, one implicitly assumes that the predefined fixed finite dictionary of basis functions contains a subset that is sufficient for a model to describe the target relation sufficiently well. The problem is that generally the required maximum degree is unknown beforehand and, since it may differ from one backward step to another, it needs to be either guessed or found by additional meta-search over the whole subset selection process.

In this paper a regression based Monte Carlo approach is developed for building sparse regression models at each backward step of the dynamic programming algorithm. This enables estimating the

DOI 10.20347/WIAS.PREPRINT.2530 Berlin 2018

value function with virtually the same cost as the standard regression algorithms based on low degree polinomials but with higher precision. The additional basis functions are constructed specifically for the optimal stopping problem at hand without using a fixed predefined finite dictionary. Specifically, the new basis functions are learned during the backward induction via incorporating information from the preceding backward induction step. Our algorithm may be viewed as a method of constructing sparse nonlinear approximations of the underlying value function and in this sense it extends the literature on deep learning type algorithms for optimal stopping problems, see, for example, the recent paper [2] and references therein.

The structure of the paper is as follows. After recalling basic facts on American options and settling the main setup in Section 2, the boosting procedure is presented in Section 3. The numerical performance is studied in Section 4.

2 Main setup

An American option grants its holder the right to select the time at which she exercises the option, i.e calls a pre-specified reward or cash-flow. This is in contrast to a European option that may be exercised only at a fixed date. A general class of American option pricing problems, i.e. optimal stopping problems respectively, can be formulated with respect to an underlying $\mathbb{R}^d$-valued Markov process $\{X_t,\, 0 \le t \le T\}$ defined on a filtered probability space $(\Omega, \mathcal{F}, (\mathcal{F}_t)_{0\le t\le T}, \mathrm{P})$. The process (X_t) is assumed to be adapted to a filtration $(\mathcal{F}_t)_{0\le t\le T}$ in the sense that each X_t is $\mathcal{F}_t$ measurable. Recall that each $\mathcal{F}_t$ is a σ -algebra of subsets of Ω such that $\mathcal{F}_s \subseteq \mathcal{F}_t \subseteq: \mathcal{F}$ for $s \le t$. Henceforth we restrict our selves to the case where only a finite number of exercise opportunities $0 < t_1 < t_2 < \ldots < t_{\mathcal{J}} = T$ are allowed (the Bermudan case in financial terms, where for notational convenience exercise at $t_0 := 0$ is excluded). (In this respect it should be noted that a continuous exercise (American) option can be approximated by such a Bermudan option with arbitrary accuracy, and so this is not a huge restriction). We now consider the pre-specified reward $g_j(Z_j)$ in terms of the Markov chain

$$Z_j := X_{t_j}, \quad j = 1, \ldots, \mathcal{J},$$

for some given functions $g_1, \ldots, g_{\mathcal{J}}$ mapping $\mathbb{R}^d$ into $[0, \infty)$. In a financial context we may and will assume that the reward $g_j(Z_j)$ is expressed in units of some (tradable) pricing numéraire that has initial value 1 Euro, say. That is, if exercised at time t_j, $j = 1, \ldots, \mathcal{J}$, the option pays cash equivalent with $g_j(Z_j)$ units of the numéraire. Let $\mathcal{T}_j$ denote the set of stopping times taking values in $\{j, j+1, \ldots, \mathcal{J}\}$. A standard result in the theory of contingent claims states that a fair price $V_j(x)$ of the Bermudan option at time t_j in state x, given that the option was not exercised prior to t_j, is its value under the optimal exercise policy,

$$V_j(x) = \sup_{\tau\in\mathcal{T}_j} \mathrm{E}[g_\tau(Z_\tau)|Z_j = x], \quad x \in \mathbb{R}^d, \tag{1}$$

due to a corresponding martingale measure, hence the solution to an optimal stopping problem. In (1) we have to read $\mathcal{T}_0 := \mathcal{T}_1$ for $j = 0$. Note that any tradable expressed in units of the numéraire is a martingale under this measure. A common feature of many approximation algorithms is that they deliver estimates $C_{N,1}(x), \ldots, C_{N,\mathcal{J}-1}(x)$ for the so-called continuation values:

$$C_j(x) := \mathrm{E}[V_{j+1}(Z_{j+1})|Z_j = x], \quad j = 1, \ldots, \mathcal{J} - 1. \tag{2}$$

Here the index N indicates that the above estimates are based on a set of N independent "training" trajectories

$$(Z_1^{(i)}, \ldots, Z_{\mathcal{J}}^{(i)}), \quad i = 1, \ldots, N, \tag{3}$$

all starting from one point. In the case of the so-called regression methods, the estimates for (1) and (2) are obtained via the *Dynamic Programming Principle*:

$$\begin{aligned} V_{\mathcal{J}}(x) &= g_{\mathcal{J}}(x), \quad C_{\mathcal{J}}(x) = 0, \\ C_j(x) &= \mathrm{E}[V_{j+1}(Z_{j+1})|Z_j = x], \quad 1 \le j < \mathcal{J}, \\ V_j(x) &= \max\left(g_j(x), C_j(x)\right), \quad 1 \le j \le \mathcal{J}, \end{aligned} \tag{4}$$

combined with Monte Carlo. These regression algorithms can be described as follows. Suppose that for some $1 \le j < \mathcal{J}$, an estimate $C_{N,j+1}(x)$ for $C_{j+1}(x)$ is already constructed. Then in the jth step one needs to estimate the conditional expectation

$$\mathrm{E}[V_{N,j+1}(Z_{j+1}))|Z_j = x], \tag{5}$$

where $V_{N,j+1}(x) = \max\left(g_{j+1}(x), C_{N,j+1}(x)\right)$. This can be done by performing regression (linear or nonlinear) on the set of paths

$$(Z_j^{(i)}, V_{N,j+1}(Z_{j+1}^{(i)})), \quad i = 1, \ldots, N.$$

The whole backward procedure is trivially initialized by setting $C_{N,\mathcal{J}}(x) = 0$. Given the estimates $C_{N,1}(x), \ldots, C_{N,\mathcal{J}-1}(x)$, we next may construct a lower bound (low biased estimate) for V_0 using the (generally suboptimal) stopping rule:

$$\tau_N = \min\big\{1 \le j \le \mathcal{J} : g_j(Z_j) \ge C_{N,j}(Z_j)\big\},$$

with $C_{N,\mathcal{J}} \equiv 0$ by definition. Indeed, fix a natural number N_{test} and simulate N_{test} new independent trajectories of the process Z. A low-biased estimate for V_0 can be then defined as

$$V_0^{N_{\text{test}},N} = \frac{1}{N_{\text{test}}} \sum_{r=1}^{N_{\text{test}}} g_{\tau_N^{(r)}}\big(Z^{(r)}_{\tau_k^{(r)}}\big) \tag{6}$$

with

$$\tau_N^{(r)} = \inf\Big\{1 \le j \le \mathcal{J} : g_j(Z_j^{(r)}) \ge C_{N,j}(Z_j^{(r)})\Big\}. \tag{7}$$

3 Adaptive regression algorithms

In this section we outline our methodology for estimating the solution to (1) at time $t = 0$, based on a set of training trajectories (3). In this respect, as a novel ingredient, we will boost the standard regression procedures by learning and incorporating new basis functions on the backward fly. As a canonical example one may consider the incorporation of $V_{N,j}$ as a basis function in the regression step of estimating C_{j-1}. Other possibilities are, for example, certain (spatial) derivatives of V_j, or functions directly related to the underlying exercise boundary at time j, for example $1_{\{g_j - C_{N,j}\}}$. In general one may choose a (typically small) number of suitable boosting basis functions at each step.

3.1 Enhancing basis on the fly

Let us suppose that we have at hand some fixed and a computationally cheep system of basis functions $(\psi_1(x), \ldots, \psi_K(x))$. We now extend this basis at each backward regression step $j-1$ with an additional and sparse set of new functions $v_1^{N,j-1}, \ldots, v_b^{N,j-1}$ that are constructed in the preceding backward step j, on the given training paths. The main idea is that the so boosted basis delivers more accurate regression estimate $C_{N,j-1}$ of the continuation function C_{j-1}, compared to the original basis, and at the same time remains cheap.

3.2 Backward boosted regression algorithm

Based on the training sample (3), we propose a boosted backward algorithm that in pseudo-algorithmic terms works as follows.

At time $\mathcal{J}$ we initialize as $C_{N,\mathcal{J}}(x) = 0$. Suppose that for $j < \mathcal{J}$, $C_{N,j}$ is already constructed in the form

$$C_{N,j}(x) = \sum_{k=1}^{K} \gamma_k^{N,j} \psi_k(x) + \sum_{k=1}^{b} \gamma_{k+K}^{N,j} \nu_k^{N,j}(x) \quad \text{for some } \gamma^{N,j} \in \mathbb{R}^{K+b}.$$

For going from $j > 0$ down to $j-1$, define the new boosted regression basis via

$$\Psi^{N,j-1}(x) := \Big(\psi_1(x), \dots, \psi_K(x), \nu_1^{N,j-1}(x), \dots, \nu_b^{N,j-1}(x)\Big) \tag{8}$$

(as a row vector) due to a choice of the set of functions $(\nu_1^{N,j-1}, \dots, \nu_b^{N,j-1})$ based on the previously estimated continuation value $C_{N,j}$. For example, we might take $b = 2$ and consider functions of the form

$$\nu_1^{N,j-1}(x) = \max(g_j(x), C_{N,j}(x)), \quad \nu_2^{N,j-1}(x) = 1_{\{g_j(x) - C_{N,j}(x) > 0\}}. \tag{9}$$

Then consider the $N \times (K+b)$ design matrix $\mathcal{M}^{j-1}$ with entries.

$$\mathcal{M}_{mk}^{j-1} := \Psi_k^{N,j-1}(Z_{j-1}^{(m)}), \quad m = 1, \dots, N,\ k = 1, \dots, K+b, \tag{10}$$

and the (column) vector

$$\begin{aligned} \mathcal{V}_j &= \Big(V_{N,j}(Z_j^{(1)}), \dots, V_{N,j}(Z_j^{(N)})\Big)^\top \\ &= \Big(\max(g_j(Z_j^{(1)}), C_{N,j}(Z_j^{(1)})), \dots, \max(g_j(Z_j^{(N)}), C_{N,j}(Z_j^{(N)}))\Big)^\top. \end{aligned} \tag{11}$$

Next compute and store

$$\gamma^{N,j-1} := \Big(\big(\mathcal{M}^{j-1}\big)^\top \mathcal{M}^{j-1}\Big)^{-1} \big(\mathcal{M}^{j-1}\big)^\top \mathcal{V}_j, \tag{12}$$

and then set

$$\begin{aligned} C_{N,j-1}(x) &= \Psi^{N,j-1}(x)\gamma^{N,j-1} \\ &= \sum_{k=1}^{K} \gamma_k^{N,j-1} \psi_k(x) + \sum_{k=1}^{b} \gamma_{k+K}^{N,j-1} \nu_k^{N,j-1}(x). \end{aligned} \tag{13}$$

3.3 Spelling out the algorithm

Let us spell out the above pseudo-algorithm under the choice (9) of boosting functions in more details. In a pre-computation step we first generate and save for $m = 1, \dots, N$, the values

$$\psi_k(Z_j^{(m)}), \quad g_i(Z_j^{(m)}), \quad 1 \le j \le i \le \mathcal{J}, \quad 1 \le k \le K. \tag{14}$$

Backward procedure For a generic backward step j we assume that the quantities

$$C_{N,j}(Z_l^{(m)}), \quad 0 \leq l \leq j, \quad m = 1, ..., N, \tag{15}$$

are already constructed and stored by using the functional approximations

$$C_{N,j}(x) = \sum_{k=1}^{K} \gamma_k^{N,j} \psi_k(x) + \gamma_{K+1}^{N,j} \nu_1^{N,j}(x) + \gamma_{K+2}^{N,j} \nu_2^{N,j}(x) \tag{16}$$

with

$$\nu_1^{N,j} = \max(g_{j+1}, C_{N,j+1}), \quad \nu_2^{N,j} = 1_{\{g_{j+1} - C_{N,j+1} > 0\}},$$

where for $j < \mathcal{J}$, $\gamma^{N,j} \in \mathbb{R}^{K+2}$ are constructed and stored.

At the initial time $j = \mathcal{J}$, we set $C_{N,\mathcal{J}} := 0$. Let us now assume that $0 < j \leq \mathcal{J}$, and proceed to time $j-1$. We first compute (10) and (11). The latter one, $\mathcal{V}_j$, is directly obtained by (15) for $l = j$ and the pre-computed values (14). To compute (10), we need $\Psi_{K+k}^{N,j-1}(Z_{j-1}^{(m)}) = \nu_k^{N,j-1}(Z_{j-1}^{(m)})$, $k = 1, 2$, $m = 1, \ldots, N$. Hence, we set

$$\begin{aligned} \nu_1^{N,j-1}(Z_{j-1}^{(m)}) &= \max(g_j(Z_{j-1}^{(m)}), C_{N,j}(Z_{j-1}^{(m)})), \\ \nu_2^{N,j-1}(Z_{j-1}^{(m)}) &= 1_{\left\{g_j(Z_{j-1}^{(m)}) - C_{N,j}(Z_{j-1}^{(m)}) \geq 0\right\}} \end{aligned}$$

for $m = 1, \ldots, N$, using (15) for $l = j - 1$. Next we may compute (and store) the coefficients vector (12), i.e., $\gamma^{N,j-1}$, using (10) and (11), and formally establish (16). In order to complete the generic backward step, we now need to evaluate

$$C_{N,j-1}(Z_l^{(m)}) = \sum_{k=1}^{K} \gamma_k^{N,j-1} \psi_k(Z_l^{(m)}) \tag{17}$$

$$+ \gamma_{K+1}^{N,j-1} \nu_1^{N,j-1}(Z_l^{(m)}) + \gamma_{K+2}^{N,j-1} \nu_2^{N,j-1}(Z_l^{(m)}), \tag{18}$$

for $m = 1, ..., N$, $0 \leq l \leq j-1$. The first part (17) is directly obtained from the pre-computation (14) and the coefficients (12) computed in this step. For the second part (18) we have that

$$\begin{aligned} \nu_1^{N,j-1}(Z_l^{(m)}) &= \max(g_j(Z_l^{(m)}), C_{N,j}(Z_l^{(m)})), \\ \nu_2^{N,j-1}(Z_l^{(m)}) &= 1_{\left\{g_j(Z_l^{(m)}) - C_{N,j}(Z_l^{(m)}) \geq 0\right\}}, \end{aligned}$$

for $m = 1, \ldots, N$, and $0 \leq l \leq j - 1$. Thus the terms (18) are directly obtained from (14), the coefficients (12), and (15).

Remark 1. *As can be seen, each approximation $C_{N,j-1}$ nonlinearly depends on all previously estimated continuation functions $C_{N,j}, \ldots, C_{N,\mathcal{J}-1}$ and hence on all "features" $(g_l(Z_l^{(m)}), \psi_k(Z_l^{(m)}), k = 1, \ldots, K, m = 1, \ldots, N, l = j, j+1, \ldots, \mathcal{J})$. In this sense our procedure tries to find a sparse deep network type approximation (with indicator or maximum as activation functions) for the continuation functions based on simulated "features". Compared to other deep learning type algorithms (see, e.g., [2]), our procedure doesn't require any type of time-consuming nonlinear optimisation over high-dimensional parameter spaces.*

Cost estimation The total cost needed to perform the pre-computation (14) is about $\frac{1}{2}N\mathcal{J}^2c_f + N\mathcal{J}Kc_f$, where c_f denotes the maximal cost of evaluating each function g_j, $j = 0, \ldots, \mathcal{J}$ and ψ_k, $k = 1, \ldots, K$, at a given point. The cost of one backward step from j to $j-1$ can be then estimated from above by

$$\begin{aligned} &NK^2c_* \quad \text{due to computation of (12)} \\ &NKjc_* \quad \text{due to the construction of (17)+(18),} \end{aligned}$$

where c_* denotes the sum of costs due to the addition and multiplication of two reals. Hence the total cost of the above algorithm can be upper bounded by

$$\frac{1}{2}N\mathcal{J}^2c_f + N\mathcal{J}Kc_f + N\mathcal{J}K^2c_* + \frac{1}{2}N\mathcal{J}^2Kc_* \tag{19}$$

including the pre-computation.

Remark 2. *In the above cost estimation the cost of determining the maximum of two numbers is neglected.*

3.4 Lower estimate based on a new realization

Suppose that the backward algorithm of Section 3.2 has been carried out, and that we now have an independent set of realizations $(\widetilde{Z}_j^{(m)}, j = 0, ..., \mathcal{J})$ with $\widetilde{Z}_0^{(m)} = X_0$, $m = 1, ..., N_{\text{test}}$. In view of (6) and (7), let us introduce the stopping rule

$$\tau_N = \inf\left\{j : 1 \le j \le \mathcal{J}, \quad g_j(Z_j) \ge C_{N,j}(Z_j)\right\}. \tag{20}$$

A lower estimate of V_0 is then obtained via

$$\underline{V_0} := \frac{1}{N_{\text{test}}} \sum_{m=1}^{N_{\text{test}}} g_{\tau_N^{(m)}}(\widetilde{Z}_{\tau_N^{(m)}}^{(m)}). \tag{21}$$

Here the index N in the $C_{N,j}$ indicates that these objects are constructed using the simulation sample used in (3.2). As a result, (20) is a suboptimal stopping time and (21) is a lower biased estimate. Let us consider the computation of (20). The coefficient vectors $\gamma^{N,j}$, $1 \le j \le \mathcal{J}$, were already computed in the backward algorithm above. We now have to consider the computation of $C_{N,j}(Z)$ for an arbitrary point $Z \in \{\widetilde{Z}_j^{(m)}, m = 1, \ldots, N_{\text{test}}\}$ at a particular time j, for $1 \le j \le \mathcal{J}$. For this we propose the following backward procedure.

Procedure for computing $C_{N,j}(Z)$ for arbitrary state Z.

1 We first (pre-)compute $\psi_k(Z)$ for $1 \le k \le K$, and $g_l(Z)$ for $j < l \le \mathcal{J}$, leading to the cost of order $(K + (\mathcal{J} - j))\, c_f$.

2 Next compute $C_{N,j}(Z)$ recursively as follows:

2.1 Initialize $C_{N,\mathcal{J}}(Z) := 0$. Once $C_{N,l}(Z)$ with $j < l \le \mathcal{J}$, is computed and saved, evaluate $\nu_1^{N,l-1}(Z)$ and $\nu_2^{N,l-1}(Z)$ using (9).

2.2 Compute

$$C_{N,l-1}(Z) = \sum_{k=1}^{K} \gamma_k^{N,l-1} \psi_k(Z) + \gamma_{K+1}^{N,l-1} \nu_1^{N,l-1}(Z) + \gamma_{K+2}^{N,l-1} \nu_2^{N,l-1}(Z)$$

at a cost of order Kc_*. In this way we proceed all the way down to $C_{N,j}(Z)$, at a total cost of $(K + (\mathcal{J} - j))\, c_f + K\,(\mathcal{J} - j)\, c_*$ including the pre-computation step.

Due to the procedure described above, the costs of evaluating (21), based on the worst case costs of computing (20), will be of order

$$N_{\text{test}} \mathcal{J} K c_f + \frac{1}{2} \mathcal{J}^2 N_{\text{test}} c_f + \frac{1}{2} N_{\text{test}} K \mathcal{J}^2 c_*.$$

Obviously, (for $N_{\text{test}} = N$) this is the same order as for the regression base backward induction procedure described in Section 3.2.

Remark 3. *From the cost analysis of the boosted regression algorithm it is obviously inferable that the standard regression procedure, i.e. the regression procedure due to a fixed basis $\psi_1, \ldots, \psi_K$ without boosting, would require a computational cost of order*

$$N \mathcal{J} K c_f + N \mathcal{J} K^2 c_*$$

for computing the regression coefficients. Hence the cost ratio due to the boosting procedure is approximately,

$$\frac{\textit{Cost for coefficients of the boosted regression}}{\textit{Cost for coefficients of the standard regression}} = 1 + \frac{\mathcal{J}}{2K}$$

A subsequent lower estimate based on a new realization in the standard case would require about $N_{test} \mathcal{J} K c_f$, yielding a cost ratio

$$1 + \frac{\mathcal{J}}{2} \frac{1 + K c_*/c_f}{K} \approx 1 + \frac{\mathcal{J}}{2} c_*/c_f$$

accordingly (assuming K is large). From this we conclude that the boosted regression is not much more expensive than the standard one as long as $\frac{\mathcal{J}}{2K}$ is small (i.e. K large), while the lower bound construction due to the boosted basis is not substantially more expensive as long as $\mathcal{J} c_* < c_f$.

4 Numerical examples

In this section we illustrate the performance of boosted regression based Monte Carlo algorithms by considering two option pricing problems in finance.

4.1 Bermudan cancelable swap

We first test our algorithm in the case of the so-called complex structured asset based cancelable swap. In particular, we demonstrate how to achieve a trade-off between accuracy and computational complexity by choosing the number of basis functions.

We consider a multi-dimensional Black-Scholes model, that is, we define the dynamic of d assets X_l, $l = 1, \dots, d$, under the risk-neutral measure via a system of SDEs

$$dX_l(t) = (\rho - \delta)X_l(t)dt + \sigma_l X_l(t)dW_l(t), \quad 0 \le t \le T, \quad l = 1, \dots, d.$$

Here $W_1(t), \dots, W_d(t)$ are correlated d-dimensional Brownian motions with time independent correlations $\rho_{lm} = t^{-1}\mathrm{E}[W_l(t)W_m(t)]$, $1 \le l, m \le d$. The continuously compounded interest rate r and a dividend rate δ are assumed to be constant.

Define the asset based cancelable coupon swap. Let $t_1, \dots, t_{\mathcal{J}}$ be a sequence of exercise dates. Fix a quantile α, $0 < \alpha < 1$, numbers $1 \le n_1 < n_2 \le d$ (we assume $d \ge 2$), and three rates s_1, s_2, s_3. Let

$$N(i) = \#\{l : 1 \le l \le d,\ X_l(t_i) \le (1-\alpha)X_l(0)\},$$

i.e., $N(i)$ is the number of assets which at time t_i are below $1 - \alpha$ percents of the initial value. We then introduce the random rate

$$a(i) = s_1 1_{\{N(i) \le n_1\}} + s_2 1_{\{n_1 < N(i) \le n_2\}} + s_3 1_{\{n_2 < N(i)\}}$$

and specify the t_i-coupon to be

$$C(i) = a(i)(t_i - t_{i-1}).$$

For pricing this structured product, we need to compare the coupons $C(i)$ with risk free coupons over the period $[t_{i-1}, t_i]$ and thus to consider the discounted net coupon process

$$\mathcal{C}(i) = e^{-rt_i}(e^{r(t_i - t_{i-1})} - 1 - C(i)), \quad i = 1, \dots, \mathcal{J}.$$

The product value at time zero may then be represented as the solution of an optimal stopping problem with respect to the adapted discounted cash-flow, obtained as the aggregated net coupon process,

$$V_0 = \sup_{\tau \in \{1, \dots, \mathcal{J}\}} \mathrm{E}[\mathcal{Z}_\tau], \quad \mathcal{Z}_j := \sum_{i=1}^{j} \mathcal{C}(i).$$

For our experiments, we choose a five-year option with semiannual exercise possibility, that is, we have

$$\mathcal{J} = 10, \quad t_i - t_{i-1} = 0.5, \quad 1 \le i \le 10,$$

on a basket of $d = 20$ assets. In detail, we take the following values for the parameters,

$$d = 20, \quad r = 0.05, \quad \delta = 0, \quad \sigma_l = 0.2, \quad X_l(0) = 100, \quad 1 \le l, m \le 20,$$
$$d_1 = 5, \quad d_2 = 10, \quad \alpha = 0.05, \quad s_1 = 0.09, \quad s_2 = 0.03, \quad s_3 = 0,$$

and

$$\rho_{lm} = \begin{cases} \rho, & l \ne m, \\ 1, & l = m. \end{cases}$$

As to the basis functions, we used a constant, the discounted net coupon process $\mathcal{C}(i)$ and the order statistics $X_{(1)} \le X_{(2)} \le \dots \le X_{(n)}$. Table 4.1 shows the results of the numerical experiment comparing the lower and the corresponding dual upper bounds by the standard linear regression method with fixed basis (the second column of Table 4.1) and by the boosted approach described in Section 3.3 with one additional basis function $(\nu_1^{N,j})$. The main conclusion is that the boosted regression algorithm delivers estimates of the same quality as the standard least squares approach by using much less basis functions (sparse basis). As a result the new algorithm turns out to be computationally cheaper.

ρ	Basis functions	Linear regression	
		Low Estimation	High Estimation
0	$1, \mathcal{C}, X_{(i)}$	171.59(0.037)	177.24(0.061)
	$1, \mathcal{C}, X_{(i)}, X_{(i)}X_{(j)}$	173.62(0.044)	177.33(0.062)
0.2	$1, \mathcal{C}, X_{(i)}$	180.0(0.060)	199.62(0.125)
	$1, \mathcal{C}, X_{(i)}, X_{(i)}X_{(j)}$	188.01(0.055)	197.02(0.143)
0.5	$1, \mathcal{C}, X_{(i)}$	176.43(0.073)	201.21(0.189)
	$1, \mathcal{C}, X_{(i)}, X_{(i)}X_{(j)}$	183.41(0.033)	196.58(0.147)
0.8	$1, \mathcal{C}, X_{(i)}$	133.29(0.065)	158.12(0.197)
	$1, \mathcal{C}, X_{(i)}, X_{(i)}X_{(j)}$	140.17(0.061)	153.49(0.106)

ρ	Basis functions	Linear regression & $\nu_1^{N,l}$	
		Low Estimation	High Estimation
0	$1, \mathcal{C}, X_{(i)}$	173.28(0.031)	177.32(0.091)
	$1, \mathcal{C}, X_{(i)}, X_{(i)}X_{(j)}$	174.33(0.036)	176.58(0.057)
0.2	$1, \mathcal{C}, X_{(i)}$	187.57(0.057)	195.09(0.121)
	$1, \mathcal{C}, X_{(i)}, X_{(i)}X_{(j)}$	188.07(0.046)	195.95(0.108)
0.5	$1, \mathcal{C}, X_{(i)}$	181.98(0.047)	194.04(0.088)
	$1, \mathcal{C}, X_{(i)}, X_{(i)}X_{(j)}$	182.93(0.057)	194.97(0.127)
0.8	$1, \mathcal{C}, X_{(i)}$	138.41(0.087)	153.08(0.106)
	$1, \mathcal{C}, X_{(i)}, X_{(i)}X_{(j)}$	139.62(0.035)	152.57(0.096)

Table 1: Comparison of the standard linear regression method and the boosted regression algorithm for the problem of pricing cancelable swaps

4.2 Bermudan MaxCal option

To illustrate the impact of including the additional basis functions, such as the indicator $\nu_2(x)$, we consider Bermudan option on the maximum of d underlying assets, each modeled by the geometric Brownian motion,

$$Z_j^k = z_0^k \exp\left\{(r - \delta_k - \sigma_k^2/2)t_j + \sigma_k W_{t_j}^k\right\}, \quad k = 1, \ldots, d,$$

with equal initial values $z_0^1 = z_0^2 = \ldots = z_0^d = z_0$, interest rate $r = 5\%$, dividend yields $\delta_1 = \delta_2 = \ldots = \delta_d = 10\%$, volatilities $\sigma_k = 0.1 + k/(2d)$, $k = 1, \ldots, d$, and d-dimensional Brownian motion W with independent components. The discounted payoff upon exercise at time t_j is the defined as

$$g_j(x) = \big[\max_{1 \le j \le d} x_j - K\big]^+ \exp(-rt_j),$$

where we take $K = 100$.

Table 2 shows a significant overall increase of the lower bound (and corresponding decrease of upper bound), for $d \le 30$, when the indicator functions $(\nu_2^{N,j})$ are added to the set of basis functions. On the other hand, it turned out that the addition of this single basis function did not lead to an appreciable increase of computation times (see also cost analysis in Remark 3).

d	z_0	With indicator		Without indicator	
		Lower	Upper	Lower	Upper
10	90	85.219	86.636	85.001	86.895
10	100	103.881	105.790	103.617	106.022
10	110	122.793	125.141	122.612	125.176
20	90	125.433	127.280	125.250	127.542
20	100	149.047	151.062	148.877	151.099
20	110	172.652	174.889	172.455	175.121
30	90	154.070	156.109	153.968	156.007
30	100	180.954	182.957	180.686	183.102
30	110	207.656	210.214	207.339	210.266
50	90	195.838	197.683	195.623	197.962
50	100	227.052	229.356	227.042	229.535
50	110	258.449	261.392	258.446	261.152
100	90	263.192	265.571	263.157	265.542
100	100	301.979	304.500	302.014	304.148
100	110	340.745	343.917	340.831	343.470

Table 2: Comparison of boosted regression algorithms with and without indicator basis functions for the Bermudan maxcall option.

References

[1] Leif B.G. Andersen. A simple approach to the pricing of bermudan swaptions in the multi-factor libor market model. *Journal of Computational Finance*, 3:5–32, 1999.

[2] Sebastian Becker, Patrick Cheridito, and Arnulf Jentzen. Deep optimal stopping. *arXiv preprint arXiv:1804.05394*, 2018.

[3] Denis Belomestny. Pricing bermudan options by nonparametric regression: optimal rates of convergence for lower estimates. *Finance and Stochastics*, 15(4):655–683, 2011.

[4] Mark Broadie and Paul Glasserman. Pricing american-style securities using simulation. *Journal of Economic Dynamics and Control*, 21(8):1323–1352, 1997.

[5] Jacques F. Carriere. Valuation of the early-exercise price for options using simulations and nonparametric regression. *Insurance: mathematics and Economics*, 19(1):19–30, 1996.

[6] Daniel Egloff et al. Monte carlo algorithms for optimal stopping and statistical learning. *The Annals of Applied Probability*, 15(2):1396–1432, 2005.

[7] Paul Glasserman. *Monte Carlo methods in financial engineering*, volume 53. Springer Science & Business Media, 2003.

[8] Francis A. Longstaff and Eduardo S. Schwartz. Valuing american options by simulation: a simple least-squares approach. *Review of Financial Studies*, 14(1):113–147, 2001.

[9] John N. Tsitsiklis and Benjamin Van Roy. Regression methods for pricing complex american style options. *IEEE Trans. Neural. Net.*, 12(14):694–703, 2001.