COMMUN. MATH. SCI.
Vol. 18, No. 1, pp. 109–121

OPTIMAL STOPPING VIA REINFORCED REGRESSION*

DENIS BELOMESTNY†, JOHN SCHOENMAKERS‡, VLADIMIR SPOKOINY§, AND BAKHYT ZHARKYNBAY¶

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

Keywords. Monte Carlo; optimal stopping; regression; reinforcement.

AMS subject classifications. 65C05; 60H35; 62P05.

1. Introduction

A discrete-time optimal stopping problem can be efficiently solved in low dimensions, for instance by tree methods or by using deterministic numerical methods for the corresponding partial differential equation. However, many optimal stopping problems arising in applications (see e.g. [10]) involve high dimensional underlying processes and this made it necessary to develop Monte Carlo methods for solving such problems. 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 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 [7], the regression-based approaches of [5, 8, 9, 11, 12] and [4]. 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 [10] and [6]). 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 modelling highly non-linear behaviour of the exercise boundary. Higher-degree polynomials or local polynomials (splines) 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. As an alternative to the polynomial bases, nonlinear approximation structures can be used instead (see, e.g. [3]).

In this note a Monte Carlo based *reinforced regression* approach is developed for

*Received: April 1, 2019; Accepted (in revised form): September 3, 2019. Communicated by Ronnie Sircar.
This work was supported by the Russian Science Foundation (RSF) grant 19-71-30020 and by the Excellence Cluster Math$^+$ Berlin, project AA4-2.

†Faculty of Mathematics, Duisburg-Essen University, Essen, Germany; Faculty of Computer Science, National University Higher School of Economics (HSE), Moscow, Russia and Institute for Information Transmission Problems (IITP), Russian Academy of Sciences (RAS), Moscow, Russia (denis.belomestny@uni-due.de).

‡Weierstrass Institute for Applied Analysis and Stochastics (WIAS), Mohrenstr. 39, 10117 Berlin, Germany (schoenma@wias-berlin.de).

§WIAS, Berlin, Germany; Departments of Mathematics and Economics, Humboldt-Universität zu Berlin, Berlin, Germany; Faculty of Computer Science, HSE, Moscow, Russia and IITP RAS, Moscow, Russia (spokoiny@wias-berlin.de).

¶Faculty of Computer Science, HSE, Moscow, Russia (bjarkinbayev@nu.edu.kz).

building sparse regression models at each backward step of the dynamic programming algorithm. This enables estimating the value function with virtually the same cost as the standard regression algorithms based on lower degree polynomials but with higher precision. The additional basis functions are constructed 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. It should be noted that, as we will see, the construction of additional basis functions requires extra storage capacity which is proportional to the square of the number of stopping opportunities. This means that the computational efficiency gain of our new approach may be limited when the number of stopping dates is too large. On the other hand, for a very large number of stopping dates regression-based approaches are unsuitable anyway, see for example the results in [13].

Our algorithm may be viewed as a method of constructing sparse nonlinear approximations (in terms of their dependence on Monte Carlo paths) of the underlying value function and in this sense it extends the literature on nonlinear learning-type algorithms for optimal stopping problems, see, for example, the recent paper [3] 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 reinforced procedure is presented in Section 3. The numerical performance is studied in Section 5.

2. Main setup

A general class of 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 $\mathcal{F}_s \subseteq \mathcal{F}_t \subseteq \mathcal{F}_T =: \mathcal{F}$ for $s \le t$. Henceforth we restrict ourselves to the case where only a finite number of stopping opportunities $0 < t_1 < t_2 < \ldots < t_{\mathcal{J}} = T$ are allowed. We now consider the pre-specified reward process $g_j(Z_j)$ 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 $[0, \infty)$. Let $\mathcal{T}_j$ denote the set of stopping times taking values in $\{j, j+1, \ldots, \mathcal{J}\}$ and consider the optimal stopping problems of the form

$$V_j(x) = \sup_{\tau \in \mathcal{T}_j} \mathsf{E}[g_\tau(Z_\tau) | Z_j = x], \quad x \in \mathbb{R}^d, \tag{2.1}$$

In (2.1) we have to read $\mathcal{T}_0 := \mathcal{T}_1$ for $j = 0$. A common feature of many approximation algorithms for optimal stopping problems is that they deliver estimates $C_{N,1}(x), \ldots, C_{N,\mathcal{J}-1}(x)$ for the so-called continuation functions:

$$C_j(x) := \mathsf{E}[V_{j+1}(Z_{j+1}) | Z_j = x], \quad j = 1, \ldots, \mathcal{J} - 1. \tag{2.2}$$

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

$$(Z_1^{(i)}, \ldots, Z_{\mathcal{J}}^{(i)}), \quad i = 1, \ldots, N, \tag{2.3}$$

all starting from one point. In the case of the so-called regression methods, the estimates for (2.1) and (2.2) are obtained via the application of *Dynamic Programming Principle*:

$$C_j(x)=\mathsf{E}[V_{j+1}(Z_{j+1})|Z_j=x],\quad V_j(x)=\max(g_j(x),C_j(x)),\quad 1\le j\le \mathcal{J}-1,$$

with $V_{\mathcal{J}}(x)=g_{\mathcal{J}}(x)$, $C_{\mathcal{J}}(x)=0$, combined with nonparametric regression.

In the setting of Tsitsiklis-van Roy [12], this regression algorithm can be described as follows. First initialize $C_{N,\mathcal{J}}(x)\equiv 0$. 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 j-th step one needs to estimate the conditional expectation

$$\mathsf{E}[V_{N,j+1}(Z_{j+1})|Z_j=x], \tag{2.4}$$

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

$$(Z_j^{(i)},V_{N,j+1}(Z_{j+1}^{(i)})),\quad i=1,\dots,N, \tag{2.5}$$

due to a family of basis functions resulting in the estimate $C_{N,j}(x)$.

In the method of Longstaff-Schwartz [11], one constructs the estimates, $C_{N,j}^{\mathrm{LS}}$ say, by regression using an interleaving set of "dummy cash-flows" $\widehat{V}_j^{(i)}$ in the following way. First initialize, besides $C_{\mathcal{J}}^{\mathrm{LS}}\equiv 0$, $\widehat{V}_{\mathcal{J}}^{(i)}:=g_{\mathcal{J}}(Z_{\mathcal{J}}^{(i)})$, $i=1,\dots,N$. Once $C_{N,j+1}^{\mathrm{LS}}$ and $\widehat{V}_{j+1}^{(i)}$ are constructed for $j+1\le\mathcal{J}$, compute the regression estimate $C_{N,j}^{\mathrm{LS}}$ with respect to some set of basis functions via (2.5) with $V_{N,j+1}(Z_{j+1}^{(i)})$ replaced by $\widehat{V}_{j+1}^{(i)}$. Next update

$$\widehat{V}_j^{(i)}=\begin{cases} g_j(Z_j^{(i)}), & g_j(Z_j^{(i)})\ge C_{N,j}^{\mathrm{LS}}(Z_j^{(i)});\\ \widehat{V}_{N,j+1}^{(i)}, & g_j(Z_j^{(i)})<C_{N,j}^{\mathrm{LS}}(Z_j^{(i)}),\end{cases}$$

for $i=1,\dots,N$ (see also [10]).

Given the estimates $C_{N,1}(x),\dots,C_{N,\mathcal{J}-1}(x)$ (Tsitsiklis-van Roy or Longstaff-Schwartz), next we may construct a lower bound (low biased estimate) for V_0 using the (generally suboptimal) stopping rule:

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

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 $V_0=V_0(x_0)$ at a given point x_0 can be then constructed as

$$V_0^{N_{\mathrm{test}},N}=\frac{1}{N_{\mathrm{test}}}\sum_{r=1}^{N_{\mathrm{test}}} g_{\tau_N^{(r)}}\big(Z_{\tau_k^{(r)}}^{(r)}\big) \tag{2.6}$$

with

$$\tau_N^{(r)}=\min\Big\{1\le j\le\mathcal{J}:g_j(Z_j^{(r)})\ge C_{N,j}(Z_j^{(r)})\Big\}. \tag{2.7}$$

3. Reinforced regression algorithms

In this section we outline our methodology for estimating the solution to (2.1) at time $t=0$, based on a set of training trajectories (2.3). In this respect, as a novel ingredient, we will reinforce the standard regression procedures by learning and incorporating

new basis functions on the backward fly. As a canonical example one may consider 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 reinforcing basis functions at each step.

3.1. Backward reinforcement of regression basis. Let us suppose that we have at hand some fixed and a computationally cheap system of basis functions $(\psi_1(x),\dots,\psi_K(x))$. We now extend this basis at each backward regression step $j-1$ with an additional and sparse set of new functions $v_1^{N,j-1},\dots,v_b^{N,j-1}$ that are constructed in the preceding backward step j, on the given training paths. The main idea is that the so constructed 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 reinforced regression algorithm. Based on the training sample (2.3), we propose a reinforced 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 reinforced regression basis via

$$\Psi^{N,j-1}(x):=\Big(\psi_1(x),\dots,\psi_K(x),\nu_1^{N,j-1}(x),\dots,\nu_b^{N,j-1}(x)\Big) \tag{3.1}$$

(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 $b=1$ and consider the function

$$\nu_1^{N,j-1}(x)=\max(g_j(x),C_{N,j}(x)). \tag{3.2}$$

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

$$\mathcal{M}^{j-1}_{mk}:=\Psi_k^{N,j-1}(Z_{j-1}^{(m)}),\quad m=1,\dots,N,\ k=1,\dots,K+b, \tag{3.3}$$

and the (column) vector

$$\begin{aligned}\mathcal{V}_j&=\Big(V_{N,j}(Z_j^{(1)}),\dots,V_{N,j}(Z_j^{(N)})\Big)^\top\\&=\Big(\max(g_j(Z_j^{(1)}),C_{N,j}(Z_j^{(1)})),\dots,\max(g_j(Z_j^{(N)}),C_{N,j}(Z_j^{(N)}))\Big)^\top.\end{aligned} \tag{3.4}$$

Next compute and store

$$\gamma^{N,j-1}:=\Big((\mathcal{M}^{j-1})^\top\mathcal{M}^{j-1}\Big)^{-1}(\mathcal{M}^{j-1})^\top\mathcal{V}_j, \tag{3.5}$$

and then set

$$C_{N,j-1}(x)=\Psi^{N,j-1}(x)\gamma^{N,j-1}$$

$$= \sum_{k=1}^{K} \gamma_k^{N,j-1} \psi_k(x) + \sum_{k=1}^{b} \gamma_{k+K}^{N,j-1} \nu_k^{N,j-1}(x). \tag{3.6}$$

REMARK 3.1. For definiteness the regression steps (3.4)-(3.5) are chosen due to the Tsitsiklis-van Roy (TV) approach [12]. With a few minor and obvious changes our reinforced regression approach may be applied to the Longstaff-Schwartz (LS) method [11] as well. Since the details and the complexity analysis are very similar, we restrict ourselves to the TV approach in this paper.

3.3. Spelling out the algorithm

Let us spell out the above pseudo-algorithm under the choice (3.2) of reinforcing functions in more details (general case can be studied in a similar way). In a pre-computation step we first generate and save for $m = 1,\dots,N$, the values

$$\psi_k(Z_j^{(m)}), \quad g_i(Z_j^{(m)}), \quad 1 \le j \le i \le \mathcal{J}, \quad 1 \le k \le K. \tag{3.7}$$

Backward procedure

At the initial time $j = \mathcal{J}$, we set $C_{N,\mathcal{J}} := 0$. For a generic backward step $j < \mathcal{J}$ we assume that the quantities

$$C_{N,j}(Z_l^{(m)}), \quad 0 \le l \le j, \quad m = 1,...,N, \tag{3.8}$$

as well as the coefficients $\gamma^{N,j} \in \mathbb{R}^{K+1}$ are already computed and stored, where formally $C_{N,j}(x)$ satisfies

$$C_{N,j}(x) = \sum_{k=1}^{K} \gamma_k^{N,j} \psi_k(x) + \gamma_{K+1}^{N,j} \nu_1^{N,j}(x) \tag{3.9}$$

with $\nu_1^{N,j} = \max(g_{j+1}, C_{N,j+1})$. Let us now assume that $0 < j \le \mathcal{J}$, and proceed to time $j-1$. We first compute (3.3) and (3.4). The latter one, $\mathcal{V}_j$, is directly obtained by (3.8) for $l = j$ and the pre-computed values (3.7). To compute (3.3), we need $\Psi_{K+1}^{N,j-1}(Z_{j-1}^{(m)}) = \nu_1^{N,j-1}(Z_{j-1}^{(m)})$, $m = 1,\dots,N$. Hence, we set

$$\nu_1^{N,j-1}(Z_{j-1}^{(m)}) = \max(g_j(Z_{j-1}^{(m)}), C_{N,j}(Z_{j-1}^{(m)}))$$

for $m = 1,\dots,N$, using (3.8) for $l = j-1$. Next we may compute (and store) the coefficients vector (3.5), i.e., $\gamma^{N,j-1}$, using (3.3) and (3.4), and formally establish (3.9). In order to complete the generic backward step, we now need to evaluate

$$C_{N,j-1}(Z_l^{(m)}) = \sum_{k=1}^{K} \gamma_k^{N,j-1} \psi_k(Z_l^{(m)}) \tag{3.10}$$

$$+ \gamma_{K+1}^{N,j-1} \nu_1^{N,j-1}(Z_l^{(m)}), \tag{3.11}$$

for $m = 1,\dots,N$, $0 \le l \le j-1$. The first part (3.10) is directly obtained from the pre-computation (3.7) and the coefficients (3.5) computed in this step. For the second part (3.11), we have that

$$\nu_1^{N,j-1}(Z_l^{(m)}) = \max(g_j(Z_l^{(m)}), C_{N,j}(Z_l^{(m)}))$$

for $m=1,\dots,N$, and $0\le l\le j-1$. Thus the terms (3.11) are directly obtained from (3.7), the coefficients (3.5), and (3.8).

REMARK 3.2.

(i) Keeping track of the whole set (3.8) (rather than some subset, for example $j-1 \le l \le j$) in the above procedure is subtle and necessary due to the nested structure of the additional basis functions backwardly generated. From a more formal programming point of view this pops up as a natural ingredient for the logical recursion invariant when going from j to $j-1$.

(ii) As can be seen, each approximation $C_{N,j-1}$ nonlinearly depends on all previously estimated continuation functions $C_{N,j},\dots,C_{N,\mathcal{J}-1}$ and hence on all "features" $(g_l(Z_l^{(m)}),\psi_k(Z_l^{(m)}), k=1,\dots,K, m=1,\dots,N, l=j,j+1,\dots,\mathcal{J})$. In this sense our procedure finds a sparse nonlinear-type approximation for the continuation functions based on simulated "features". Compared to other nonlinear learning-type algorithms (see, e.g., [3]), our procedure doesn't require any nonlinear optimization over high-dimensional parameter spaces.

Cost estimation. The total cost needed to perform the pre-computation (3.7) 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, $j=0,\dots,\mathcal{J}$, and ψ_k, $k=1,\dots,K$, at a given point. The cost of one backward step from j to $j-1$ can be then estimated from above by

$$\begin{aligned} &NK^2c_* && \text{due to computation of (3.5)}\\ &NKjc_* && \text{due to the construction of (3.10)+(3.11)}, \end{aligned}$$

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

$$\frac{1}{2}N\mathcal{J}^2c_f+N\mathcal{J}Kc_f+N\mathcal{J}K^2c_*+\frac{1}{2}N\mathcal{J}^2Kc_* \tag{3.12}$$

including the pre-computation.

3.4. Lower estimate based on a new realization. The backward algorithm of Section 3.2 gives an approximation to the stopping problem that may be biased upwards or downwards. In order to compare different methods properly, it is better to construct lower bounds due to the estimated continuation functions of the different methods, which may be done via an independent re-simulation of the underlying process. Specifically suppose that the backward algorithm of Section 3.2 has been carried out, and that we now have an independent set of paths $(\widetilde{Z}_j^{(m)}, j=0,\dots,\mathcal{J})$ with $\widetilde{Z}_0^{(m)}=X_0$, $m=1,\dots,N_{\text{test}}$. In view of (2.6) and (2.7), let us introduce the stopping rule

$$\tau_N=\min\bigl\{j:1\le j\le\mathcal{J},\quad g_j(Z_j)\ge C_{N,j}(Z_j)\bigr\}. \tag{3.13}$$

A lower estimate for the value $V_0=V_0(x_0)$ at a given point x_0 is then obtained via

$$\underline{V_0}:=\frac{1}{N_{\text{test}}}\sum_{m=1}^{N_{\text{test}}} g_{\tau_N^{(m)}}\Bigl(\widetilde{Z}^{(m)}_{\tau_N^{(m)}}\Bigr). \tag{3.14}$$

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, (3.13) is a suboptimal stopping time and (3.14) is a lower biased estimate. Let us consider the computation of (3.13). 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 $Z\in\{\widetilde{Z}_j^{(m)}, m=1,\dots,N_{\text{test}}\}$ 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:

(a) Initialize $C_{N,\mathcal{J}}(Z):=0$. Once $C_{N,l}(Z)$ with $j<l\le\mathcal{J}$, is computed and saved, evaluate $\nu_1^{N,l-1}(Z)$ using (3.2).

(b) Compute

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

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 (3.14), based on the worst case costs of computing (3.13), will be of order

$$N_{\text{test}}\mathcal{J}Kc_f+\frac{1}{2}\mathcal{J}^2N_{\text{test}}c_f+\frac{1}{2}N_{\text{test}}K\mathcal{J}^2c_*. \tag{3.15}$$

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

3.5. Cost comparison standard vs reinforced regression. From the cost analysis of the reinforced regression algorithm it is obviously inferable that the standard regression procedure, that is, the regression procedure due to a fixed basis $\psi_1,\dots,\psi_K$ without reinforcement, would require a computational cost of order

$$N\mathcal{J}Kc_f+N\mathcal{J}K^2c_* \tag{3.16}$$

for computing the regression coefficients. As an ultimate goal of the reinforcement method we will try to achieve an accuracy comparable with standard regression, while the cardinality of the fixed basis is vastly reduced. If we denote the cardinality of the fixed basis in the reinforced regression by K_r, the cost ratio with respect to standard regression is then given by (3.12)/(3.16), that is

$$\frac{\text{Cost for coefficients of the reinforced regression}}{\text{Cost for coefficients of the standard regression}}=\frac{K_r+\mathcal{J}/2}{K}\frac{1+K_rc_*/c_f}{1+Kc_*/c_f}.$$

On the other hand, a subsequent lower estimate based on a new realization in the standard case would require about $N_{\text{test}}\mathcal{J}Kc_f$, yielding a cost ratio (see (3.15)),

$$\frac{\text{Cost new simulation reinforced regression}}{\text{Cost new simulation standard regression}}=\frac{K_r+\mathcal{J}/2}{K}+\frac{1}{2}\frac{\mathcal{J}K_r}{K}c_*/c_f.$$

From this we conclude that the cost reduction due to the reinforced regression algorithm is "large" when $(K_r+\mathcal{J}/2)/K$ is "small", while there is also a "large" reduction in the lower bound construction when in addition $\mathcal{J}c_*\lesssim c_f$ (for example).

4. Some theoretical results

Let us consider for a random vector $(X,Y)\in\mathbb{R}^d\times\mathbb{R}$ on some probability space $(\Omega,\mathcal{F},\mathbb{P})$, a problem of estimating the conditional expectation

$$u(x)=\mathrm{E}[Y\,|\,X=x], \tag{4.1}$$

based on a sample $(X^{(n)},Y^{(n)})$, $n=1,\dots,N$, from the joint distribution of (X,Y). Suppose that the regression basis consists of a fixed set of standard basis functions $\psi_k:\mathbb{R}^d\to\mathbb{R}$, $k=1,...,K$, (for example, polynomials) and a set of auxiliary basis functions $\nu_1,\dots,\nu_b$, where typically b is much smaller than K. The idea is that the function u can be well approximated by functions from $\mathcal{V}_b:=\mathsf{span}\{\nu_1,\dots,\nu_b\}$. In this case one can consider the least squares problem,

$$\widetilde{\beta}:=\operatorname*{arg\,inf}_{\beta\in\mathbb{R}^{K+b}}\sum_{n=1}^{N}\left(Y^{(n)}-\sum_{k=1}^{K}\widetilde{\beta}_k\psi_k(X^{(n)})-\sum_{k=1}^{b}\widetilde{\beta}_{K+k}\nu_k(X^{(n)})\right)^2 \tag{4.2}$$

and set

$$\widetilde{u}(x)=\sum_{k=1}^{K}\widetilde{\beta}_k\psi_k(x)+\sum_{k=1}^{b}\widetilde{\beta}_{K+k}\nu_k(x). \tag{4.3}$$

The following theorem provides error bounds for $\widetilde{u}$, see [2].

THEOREM 4.1 (Accuracy standard global regression). *Fix some $\varepsilon\in(0,1)$. Suppose that*

$$\sup_{x\in\mathbb{R}^d}|u(x)|\le L \quad \textit{and} \quad \sup_{x\in\mathbb{R}^d}\mathrm{Var}[Y\,|\,X=x]\le\sigma^2,$$

then it holds with probability at least $1-\varepsilon$

$$\begin{aligned}\int|\widetilde{u}(x)-u(x)|^2\mu(dx)\lesssim&\max\left(\sigma^2,L^2\right)\frac{(1+\ln N)K+\log(\varepsilon^{-1})}{N}\\&+\inf_{w\in\Psi_K+\mathcal{V}_b}\int_{\mathbb{R}^d}|w(x)-u(x)|^2\mu(dx)\end{aligned} \tag{4.4}$$

where $\Psi_K:=\mathsf{span}\{\psi_1,\dots,\psi_K\}$, μ denotes the distribution of X in (4.1) and $\lesssim$ stands for inequality up to some absolute constant.

In view of (4.2) one trivially has for any arbitrary but fixed $w(x)\in\Psi_K+\mathcal{V}_b$,

$$\begin{aligned}&\inf_{\widetilde{\beta}\in\mathbb{R}^{K+b}}\sum_{n=1}^{N}\left(Y^{(n)}-\sum_{k=1}^{K}\widetilde{\beta}_k\psi_k\left(X^{(n)}\right)-\sum_{k=1}^{b}\widetilde{\beta}_{K+k}\nu_k\left(X^{(n)}\right)\right)^2\\&=\inf_{\widehat{\beta}\in\mathbb{R}^{K+b}}\sum_{n=1}^{N}\left(Y^{(n)}-w(X^{(n)})-\sum_{k=1}^{K}\widehat{\beta}_k\psi_k\left(X^{(n)}\right)-\sum_{k=1}^{b}\widehat{\beta}_{K+k}\nu_k\left(X^{(n)}\right)\right)^2\end{aligned}$$

with the corresponding estimator

$$\widehat{u}(x)=\sum_{k=1}^{K}\widehat{\beta}_k\psi_k(x)-\sum_{k=1}^{b}\widehat{\beta}_{K+k}\nu_k(x) \tag{4.5}$$

of the function $u(x)-w(x)$. Due to (4.4) we thus have for (4.5),

$$\int |\widehat{u}(x)-u(x)+w(x)|^2\mu(dx)\lesssim \max\left(\sigma^2,L_w^2\right)\frac{(1+\ln N)K+\log(\varepsilon^{-1})}{N}+\delta_K$$

with

$$L_w:=\sup_{x\in\mathbb{R}^d}|u(x)-w(x)|,\quad \delta_K:=\inf_{w\in\Psi_K+\mathcal{V}_b}\int_{\mathbb{R}^d}|w(x)-u(x)|^2\mu(dx).$$

Since the choice of w was arbitrary, we derive with probability at least $1-\varepsilon$

$$\int |\widetilde{u}(x)-u(x)|^2\mu(dx)\lesssim \max\left(\sigma^2,L_\star^2\right)\frac{(1+\ln N)K+\log(\varepsilon^{-1})}{N}+\delta_K,$$

where

$$L_\star:=\inf_{w\in\Psi_K+\mathcal{V}_b}\sup_{x\in\mathbb{R}^d}|u(x)-w(x)|.$$

The reduction of the bound L in (4.4) to $L_\star$ is of prime importance in the backward algorithm developed in Section 3.3. In particular, for a diffusion process X, the conditional variance of the underlying process $Z_j=X_{t_j}$ at t_j, given its state $Z_{j-1}=X_{t_{j-1}}$, is of order $O(t_j-t_{j-1})$. It is not difficult to show that, under some conditions,

$$\mathsf{Var}[V_j(Z_j)|Z_{j-1}=z]\le \mathsf{E}\left[(V_j(Z_j)-V_j(Z_{j-1}))^2|Z_{j-1}=z\right]=O(t_j-t_{j-1}),$$

uniformly in z, implying $\sigma^2\lesssim\max_j(t_j-t_{j-1})$ in (4.4). As a result,

$$\begin{aligned}L_\star&\le\max_j\sup_z\mathsf{E}[|V_j(Z_j)-V_j(z)||Z_{j-1}=z]|\\&\le\max_j\sup_z\sqrt{\mathsf{E}\left[|V_j(Z_j)-V_j(z)|^2|Z_{j-1}=z\right]}\lesssim\max_j\sqrt{t_j-t_{j-1}}.\end{aligned}$$

So in this case $\sigma^2\ll L$ in (4.4) and the decrease of L to $L_\star\asymp\sigma$ will result in a substantial computational gain.

5. Numerical examples

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

5.1. Bermudan max-call on d assets. This is a benchmark example studied in [7] among others. Specifically, the model with d identically distributed assets is considered, where each underlying asset has dividend yield δ. The risk-neutral dynamic of assets is given by

$$\frac{dX_t^k}{X_t^k}=(r-\delta)dt+\sigma dW_t^k,\quad k=1,...,d,$$

where $W_t^1,\dots,W_t^d$ are independent one-dimensional Brownian motions and r,δ,σ are constants. At any time $t\in\{t_0,\dots,t_{\mathcal{J}}\}$ the holder of the option may exercise it and receive the payoff

$$g(X_t)=(\max(X_t^1,\dots,X_t^d)-K)^+.$$

We take $t_i = iT/\mathcal{J}, i = 0,...,\mathcal{J}$, with $T = 3, \mathcal{J} = 9$ and $X_0 = (X_0^1,\dots,X_0^d)^T$ with $X_0^1 = \dots = X_0^d = x_0$. The lower bounds for the standard least-squares approach and the reinforced regression algorithm are presented in Table 5.1 depending on dimension d and the choice of basis functions. In both cases we generate $N = 1{,}000{,}000$ paths to estimate regression coefficients and another $N_{\text{test}} = 1{,}000{,}000$ to construct lower bounds (see (3.14)) presented in Table 5.1. The dual upper bounds in the last column of Table 5.1 are obtained based on the reinforced regression using 1000 inner paths and 1,000,000 outer paths. Table 5.1 shows that the RLS algorithm is more efficient than the LS algorithm. As one can see, there is a clear improvement in bounds when using the same basis functions across all dimensions. This improvement is especially pronounced in small dimensions. Indeed, as can be seen from Table 5.1, the lower bounds for the RLS algorithm achieved when using linear polynomials, can be obtained for the LS algorithm only on quadratic ones resulting in a cost reduction of order $\frac{2d+\mathcal{J}}{d(d+1)}$ (see Section 3.5), which increases when d decreases. The basis $1,(X_i),g(X)$ is skipped for the reinforced regression, since g is already included (at least at time $\mathcal{J}-1$).

Dimension	Basis functions	Lower bounds		Upper bounds
		Regression	Reinf. Reggression	
2	$1, X_i$	12.91(0.018)	13.77(0.015)	14.12(0.042)
	$1, X_i, X_iX_j$	13.75(0.014)	13.86(0.016)	13.97(0.026)
	$1, X_i, g(X)$	13.66(0.023)	-	14.09(0.071)
5	$1, X_i$	25.25(0.013)	25.99(0.017)	26.34(0.080)
	$1, X_i, X_iX_j$	25.93(0.020)	26.12(0.017)	26.22(0.026)
	$1, X_i, g(X)$	25.82(0.026)	-	26.35(0.081)
10	$1, X_i$	37.95(0.025)	38.22(0.020)	38.48(0.073)
	$1, X_i, X_iX_j$	38.27(0.014)	38.31(0.021)	38.41(0.028)
	$1, X_i, g(X)$	38.03(0.016)	-	38.59(0.066)
20	$1, X_i$	51.48(0.019)	51.61(0.024)	51.88(0.091)
	$1, X_i, X_iX_j$	51.72(0.023)	51.73(0.023)	51.79(0.035)
	$1, X_i, g(X)$	51.50(0.020)	-	51.87(0.122)

TABLE 5.1. *Bounds (with 95% confidence intervals) for the Bermudan max-call with parameters* $K = 100, r = 0.05$, $\sigma = 0.2$, $\delta = 0.1$, $x_0 = 100$ *and different values of* d.

5.2. Bermudan cancelable swap. We test our algorithm in the case of the so-called complex structured asset-based cancelable swap. 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 $\rho_{lm} = t^{-1}\mathsf{E}[W_l(t)W_m(t)]$, $1 \le l,m \le d$. 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

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

ρ	Basis functions	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	Reinf. regression	
		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)
	$1,\mathcal{C},X_{(i)},X_{(i)}X_{(j)}$	183.93(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 5.2. *Comparison of the standard linear regression method and the reinforced regression algorithm for the problem of pricing cancelable swaps.*

$\mathcal{J}$	Basis	Regression		Reinforced Regression	
		CPU Time	Memory	CPU time	Memory
10	$1,\mathcal{C},X_{(i)}$	7m 40s	7055 MiB	9m 10s	7062 MiB
	$1,\mathcal{C},X_{(i)},X_{(i)}X_{(j)}$	27m	9666 MiB	30m 35s	11500 MiB
20	$1,\mathcal{C},X_{(i)}$	21m 30s	13510 MiB	28m 50s	13520 MiB
	$1,\mathcal{C},X_{(i)},X_{(i)}X_{(j)}$	1h 2m 50s	15100 MiB	1h 19m 20s	15200 MiB
30	$1,\mathcal{C},X_{(i)}$	41m 50s	20000 MiB	59m 50s	20000 MiB
	$1,\mathcal{C},X_{(i)},X_{(i)}X_{(j)}$	1h 45m 3s	21600 MiB	2h 26m 45s	21600 MiB

TABLE 5.3. *Pricing cancelable swaps: comparison of run times and memory requirements for the standard linear regression method and the reinforced regression algorithm.*

that is, $N(i)$ is the number of assets which at time t_i are below $1-\alpha$ percent 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,

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

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,

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

and

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

As to the basis functions, we used a constant, the discounted net coupon process $\mathcal{C}(i)$ and the order statistics $X_{(1)} \le X_{(2)} \le \ldots \le X_{(n)}$. Table 5.2 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 5.2) and by the reinforced regression approach described in Section 3.3 with one additional basis function ($\nu_1^{N,j}$). The main conclusion is that the reinforced 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. This can also be seen from Table 5.3 where CPU times and memory usage statistics are reported.

REFERENCES

[1] L.B.G. Andersen, *A simple approach to the pricing of Bermudan swaptions in the multi-factor LIBOR market model*, J. Comput. Finance, 3:5–32, 1999. 1

[2] J.-Y. Audibert and O. Catoni, *Robust linear least squares regression*, Ann. Statist., 39(5):2766–2794, 2011. 4

[3] S. Becker, P. Cheridito, and A. Jentzen, *Deep optimal stopping*, J. Mach. Learn. Res., 20:1–25, 2019. 1, 3.2

[4] D. Belomestny, *Pricing Bermudan options by nonparametric regression: optimal rates of convergence for lower estimates*, Finance Stoch., 15(4):655–683, 2011. 1

[5] D. Belomestny, A. Kolodko, and J. Schoenmakers, *Regression methods for stochastic control problems and their convergence analysis*, SIAM J. Control Optim., 48(5):3562–3588, 2009/10. 1

[6] D. Belomestny and J. Schoenmakers, *Advanced Simulation-Based Methods for Optimal Stopping and Control: With Applications in Finance*, Palgrave Macmillan, London, 2018. 1

[7] M. Broadie and P. Glasserman, *Pricing American-style securities using simulation*, J. Econ. Dyn. Control, 21(8):1323–1352, 1997. 1, 5.1

[8] J.F. Carriere, *Valuation of the early-exercise price for options using simulations and nonparametric regression*, Insur. Math. Econ., 19(1):19–30, 1996. 1

[9] D. Egloff, *Monte Carlo algorithms for optimal stopping and statistical learning*, Ann. Appl. Probab., 15(2):1396–1432, 2005. 1

[10] P. Glasserman, *Monte Carlo Methods in Financial Engineering*, Springer Science & Business Media, 53, 2003. 1, 2

[11] F.A. Longstaff and E.S. Schwartz, *Valuing American options by simulation: a simple least-squares approach*, Rev. Financ. Stud., 14(1):113–147, 2001. 1, 2, 3.1

[12] J. Tsitsiklis and B. Van Roy, *Regression methods for pricing complex American style options*, IEEE Trans. Neural. Net., 12(14):694–703, 2001. 1, 2, 3.1

[13] D.Z. Zanger, *Quantitative error estimates for a least-squares Monte Carlo algorithm for American option pricing*, Finance Stoch., 17(3):503–534, 2013. 1