'Regression Anytime' with Brute-Force SVD Truncation

Christian Bender[1] and Nikolaus Schweizer[2]

August 21, 2019

Abstract

We propose a new least-squares Monte Carlo algorithm for the approximation of conditional expectations in the presence of stochastic derivative weights. The algorithm can serve as a building block for solving dynamic programming equations, which arise, e.g., in non-linear option pricing problems or in probabilistic discretization schemes for fully non-linear parabolic partial differential equations. Our algorithm can be generically applied when the underlying dynamics stem from an Euler approximation to a stochastic differential equation. A built-in variance reduction ensures that the convergence in the number of samples to the true regression function takes place at an arbitrarily fast polynomial rate, if the problem under consideration is smooth enough.

Keywords: Monte Carlo simulation, Least-Squares Monte Carlo, Regression Later, Dynamic Programming, BSDEs, Quantitative Finance, Statistical Learning

AMS 2000 Classification: 65C05, 60H35, 62G08, 90C39.

1 Introduction

Approximating conditional expectations functions numerically is one of the central difficulties when solving dynamic programming problems in financial and economic applications [15, 35, 36], or in implementations of probabilistic time-discretization schemes for parabolic partial differential equations [10, 42, 14, 38]. For instance, when solving an optimal stopping problem numerically, there are trade-offs between stopping now for an immediate reward or waiting, receiving the continuation value – which is a conditional expectation of future rewards. In a highly influential paper, Longstaff and Schwartz [30] proposed to compute conditional expectations functions within Monte Carlo simulations in exactly the same way such functions are estimated from real world data. In their least-squares Monte Carlo (LSMC) approach, conditional expectations are approximated by regressing future realizations of some quantity of interest on basis functions (e.g. polynomials) that depend on current values of the state variables. This approach of mimicking the empiricist's regression method with real data replaced by simulated data has been the starting point of a vast and successful LSMC literature, see [39, 37, 2, 29, 11, 32, 14, 20, 13, 26, 18, 33] for a broad selection of contributions from various fields ranging from economics and finance to numerical analysis. Yet early on, Glasserman and Yu [16] pointed out that, in principle, exploiting properties of the Monte Carlo setting that are not available to the empiricist

[1]Saarland University, Department of Mathematics, Postfach 151150, D-66041 Saarbrücken, Germany, bender@math.uni-sb.de

[2]Tilburg University, Department of Econometrics and OR, PO box 90153, NL-5000 LE Tilburg, The Netherlands, n.f.f.schweizer@uvt.nl

can lead to even more powerful algorithms. Specifically, they proposed to consider LSMC algorithms where the basis functions depend on future values of the state variables. They called this type of algorithm 'regression later' to contrast it against traditional LSMC algorithms which rely on a 'regression now'.

When the underlying simulation model has the property that conditional expectations of the basis functions can be computed in closed form, 'regression later' provides an alternative method for approximating conditional expectations. For instance, when the basis functions are polynomials, regression later requires that conditional moments of the simulated state process are available in closed form. To appreciate the promise of 'regression later', consider a setting where the quantity within the time-t conditional expectation happens to be a quadratic polynomial of some univariate state variable at time $t+1$. Suppose that such time-t conditional expectations of quadratic polynomials are again quadratic polynomials in the state variable at time t. In that case, under very mild conditions, three(!) random observations of the state variable at time $t+1$ will suffice to determine the time-t conditional expectations function without error using 'regression later'. In contrast, even when the model is correctly specified, fitting three coefficients by 'regression now' will need more than three observations due to the classical (and slow) square-root convergence of Monte Carlo methods.

Despite this tremendous promise, 'regression later' has not replaced 'regression now' as the default algorithm for numerical approximation of conditional expectations in its first 15 years. In our view, there are at least three reasons for this. (i) The method is harder to implement than 'regression now' as it requires implementing not only the basis functions but also their conditionals expectations. On top of this, in 'regression later', basis choice is restricted to functions with known closed-form conditional expectations while it is almost unrestricted in 'regression now'. (ii) It took time to become clear that the most successful applications of 'regression later' seem to be outside the original optimal stopping setting of [16], see e.g. [8, 9, 34, 7, 4]. Crucially, [8] pointed out that, under mild assumptions, Malliavin derivatives of the quantity of interest can be computed without additional error in 'regression later'. This observation is (largely) irrelevant for optimal stopping. Yet it implies that one of the greatest difficulties of 'regression now' (see [1] for discussion and control variates) does not exist in the 'regression later' setting. (iii) Finally, while 'regression later' promises to converge much faster than Monte Carlo and thus 'regression now', there is, despite partial results in [9], a lack of theoretical support for this claim. The main obstacle in the theoretical analysis is that the usual derivations of convergence rates for regression problems rely on truncations to stabilize the approximation. Yet, basically, this truncation would have to be applied after the regression but before the closed-form calculation – thus destroying the scope for making the calculation in closed-form.

To address these shortcomings and combine the advantages of 'regression later' and 'regression now', this paper introduces and rigorously analyzes 'Regression Anytime with Brute-Force SVD Truncation' (RAWBFST, pronounced "raw-beef-st"). Under sufficient smoothness, our algorithm can be calibrated to achieve any polynomial convergence rate for the mean squared error, thus holding some of the (bold) promise of convergence with a finite number samples in a fairly generic setting. These rates are achieved not only for the conditional expectations functions but also for their Malliavin derivatives that can be computed simultaneously. Compared to the results in [9] for 'regression later', our convergence rates for RAWBFST are better, matching theoretically optimal rates from the statistical learning literature [27] under certain conditions (and up to logarithmic factors). Moreover, while the results in [9] are based on having access to an orthonormal basis for the distribution of interest, RAWBFST is based on an explicit basis of localized Legendre polynomials that does not need to be tailored to the problem at hand.

The first ingredient of RAWBFST is what we call 'regression anytime'. The idea is to let the basis functions depend on the state variables both 'now' and 'later'. In particular, we consider basis functions that are products of a function that depends on 'now' and a function that depends on 'later'. When taking the conditional expectation given 'now' of such a function, the first factor depending only on 'now' can be pulled out of the expectation. Thus, to compute the conditional expectation of the basis function, it suffices to consider the second factor. 'Regression anytime' was previously applied within the stochastic grid bundling method of [25, 12] and in the LSMC algorithm of [6]. While the idea of 'regression anytime' is simple, its additional flexibility in basis choice is key for developing practically effective implementations of 'regression later', both in their settings and in ours. In particular, 'regression later' is a special case of 'regression anytime' and, after a suitable redefinition of the state process, 'regression anytime' can be rephrased as 'regression later'. In this sense, the step from 'regression later' to 'regression anytime' is a change of perspective and a reevaluation of possibilities rather than the invention of a new algorithm.

The second ingredient of RAWBFST is a new type of truncation – along with a machinery for controlling the truncation error. The idea of 'Brute-Force SVD Truncation' is again simple. Before the regression, we compute the singular values of the empirical regression matrix. If all singular values are above a previously specified threshold, we perform the regression in the usual way, otherwise we set all coefficients to zero. The key ingredient of our error analysis for RAWBFST is a new bound on the approximation error of noiseless regression with this type of brute-force SVD truncation, our Theorem 4.3. Here, the term 'noiseless' refers to settings in which the observations and the explanatory variables are driven by the same randomness so that the regression problem is ultimately an interpolation problem. We show that the statistical error of this type of regression vanishes exponentially quickly in the number of Monte Carlo samples for a fixed set of basis functions. This exponential decay is a key reason why our Monte Carlo algorithm can do better than the usual Monte Carlo convergence rate. The exponential decay implies that, up to log-factors, we can achieve convergence by letting the number of basis functions and the number of Monte Carlo samples grow at the same rate.

The bound of Theorem 4.3 can be applied to any regression later or regression anytime algorithm that applies our brute-force SVD truncation. This way, we can, e.g., achieve a better convergence rate for 'regression later' than the one that [9] show without SVD truncation. The exponential decay rate in Theorem 4.3 depends explicitly on the singular values of the expected regression matrix, i.e. the matrix containing the $\mathcal{L}^2$-inner products of the basis functions with respect to the law of the state variable. This dependence stems from an application of a matrix Bernstein inequality that links these singular values to the probability that the brute-force SVD truncation is performed. Thus, to fully exploit Theorem 4.3 some control of the singular values of the expected regression matrix is necessary. As our second main contribution, we propose an explicit construction of a 'regression anytime' algorithm that provides a guaranteed control of the singular values in generic settings where the state process is discretized via (one step of) a Euler scheme. This is the RAWBFST algorithm, which is detailed in Section 3.

The paper is organized as follows: In Section 2 we describe the problem of computing conditional expectations with Malliavin Monte Carlo weights for higher-order partial derivatives. We, then, explain how 'regression now'- and 'regression later'-algorithms deal with this problem, and discuss advantages and drawbacks of these approaches. Section 3 is devoted to our new algorithm, RAWBFST. We first state the algorithm and provide the resulting error analysis in Theorem 3.5. It shows that the algorithm converges to a partial derivative of order α as $L^{-(Q+1)/(D(1+\alpha/2))}$ (up to log-factors) in the total number

L of simulated samples, where D is the dimension of the problem and the function is of class $\mathcal{C}_b^{Q+1}(\mathbb{R}^D)$. In particular, it can beat the Monte Carlo rate of $1/2$, if the smoothness-to-dimension ratio is sufficiently large. The theoretical convergence results are tested in two numerical examples: The first example is the computation of the second derivative of a smooth function, in which we empirically verify our theoretical findings. In the second example, we call the RAWBFST algorithm iteratively within a dynamic programming framework for option pricing under uncertain volatility (corresponding to solving a fully non-linear second-order parabolic Cauchy problem). Again, the built-in variance reduction ensures that our algorithm converges faster than standard Monte Carlo pricing of a European option in the Black-Scholes model. In Section 4, we analyze the convergence behavior of noiseless regression with brute-force SVD truncation, thus providing the main building of the RAWBFST algorithm. Finally, the technical details of the error analysis for RAWBFST are provided in Section 5.

Notation

For vectors $x = (x_1, \dots, x_D) \in \mathbb{R}^D$, we write $|x|_p = (\sum_d |x_d|^p)^{1/p}$ for the p-norm ($p \geq 1$) and $|x|_\infty = \max_d |x_d|$ for the maximum norm. Given a symmetric matrix $A \in \mathbb{R}^{D\times D}$, $\lambda_{\min}(A)$ and $\lambda_{\max}(A)$ denote the smallest and largest eigenvalue of A respectively. For general matrices $A \in \mathbb{R}^{D_1\times D_2}$, we apply the spectral norm $\|A\|_2 = \sqrt{\lambda_{\max}(AA^\top)}$, where $(\cdot)^\top$ stands for matrix transposition. By $O(D)$ we denote the set of orthogonal matrices in $\mathbb{R}^{D\times D}$. We write $\mathcal{C}_b^Q(\mathbb{R}^D)$ for the space of bounded real valued function on $\mathbb{R}^D$, which are Q-times continuously differentiable with bounded derivatives ($Q \in \mathbb{N}_0$), and $\|f\|_\infty = \sup_{x\in\mathbb{R}^D} |f(x)|_2$ for the sup-norm of a function $f : \mathbb{R}^D \to \mathbb{R}^M$. $\chi^2_D(\alpha)$ denotes the $(1-\alpha)$-quantile of the χ^2-distribution with D degrees of freedom while Φ and φ stand for the distribution function and density of the standard normal distribution, respectively. Given a random vector X in $\mathbb{R}^D$, we write $\operatorname{supp} X$ for the support of X, i.e. the set of $x \in \mathbb{R}^D$ such that X hits every ϵ-ball around x with positive probability. For a vector $x \in \mathbb{R}^D$ and a constant $r > 0$ we denote by $[x]_r$ the componentwise truncation at level $\pm r$, i.e.

$$[x]_r = ([x_1]_r, \dots [x_D]_r)^\top, \quad [x_d]_r = \max(\min(x_d, r), -r).$$

2 State of the art

2.1 Setting of the problem

Our main motivation is the problem of approximating conditional expectations of the form

$$E[\mathcal{H}_{\iota,\Delta}(\xi) y(X_2)|X_1] \tag{1}$$

via empirical least-squares regression, where X_2 is one step of an Euler scheme with step size Δ starting at X_1 and $\mathcal{H}_{\iota,\Delta}(\xi)$ is a Malliavin Monte Carlo weight for the approximation of a (higher order) partial derivative of y. These type of conditional expectations appear in discretization schemes for backward stochastic differential equations (BSDEs), see e.g. [42, 10], and, more generally, in stochastic time discretization schemes of fully nonlinear parabolic partial differential equations (PDEs), see [14, 38], including Hamilton-Jacobi-Bellman equations arising from stochastic control problems.

More precisely, let X_1 be an $\mathbb{R}^D$-valued random variable with law μ_1, and denote by ξ a D-dimensional vector of independent standard normal random variables, which is also assumed to be independent of X_1. For measurable coefficient functions $b : \mathbb{R}^D \to \mathbb{R}^D$ and $\sigma : \mathbb{R}^D \to \mathbb{R}^{D\times D}$, we consider

$$X_2 = X_1 + b(X_1)\Delta + \sigma(X_1)\sqrt{\Delta}\xi. \tag{2}$$

Conditions on the law of X_1 and on the coefficient functions will be specified later on. On the function $y : \mathbb{R}^D \to \mathbb{R}$ we assume that it is $Q+1$-times continuously differentiable and bounded with bounded derivatives, for some $Q \in \mathbb{N}$. The boundedness assumptions can, of course, be relaxed, but we impose them for sake of simplicity. In order to specify the stochastic weights, we denote by

$$\mathcal{H}_q(x) = (-1)^q e^{x^2/2} \frac{\mathrm{d}^q}{\mathrm{d}x^q} e^{-x^2/2}, \quad x \in \mathbb{R},$$

the Hermite polynomial with parameter 1 of degree $q \in \mathbb{N}_0$. For a multi-index $\iota \in \mathbb{N}_0^D$, we denote its absolute value by $|\iota|_1 = \sum_{d=1}^D \iota_d$. Then, the stochastic weight is defined as a scaled multivariate Hermite polynomial of degree $|\iota|_1$, namely,

$$\mathcal{H}_{\iota,\Delta}(x) = \Delta^{-|\iota|_1/2} \prod_{d=1}^D \mathcal{H}_{\iota_d}(x_d), \quad x = (x_1, \dots, x_D).$$

Moreover, we write $\bar{\iota}$ for the vector in $\{1, \dots D\}^{|\iota|_1}$ which has, for each $d = 1, \dots, D$, the entry d ι_d-times, and whose entries are increasingly ordered. Note that, under the assumptions stated above, integration by parts yields, for $1 \le |\iota|_1 \le Q-1$,

$$\begin{aligned} & E[\mathcal{H}_{\iota,\Delta}(\xi) y(X_2) | X_1] \\ = & \sum_{j_1,\dots,j_{|\iota|_1}=1}^D \sigma_{j_1,\bar{\iota}_1}(X_1) \cdots \sigma_{j_{|\iota|_1},\bar{\iota}_{|\iota|_1}}(X_1) E\left[\frac{\partial^{|\iota|_1}}{\partial(x_{j_1}, \dots x_{j_{|\iota|_1}})} y(X_2) \middle| X_1 \right]. \end{aligned} \tag{3}$$

Hence, the conditional expectation (1) approximates the weighted sum of partial derivatives of y

$$\sum_{j_1,\dots,j_{|\iota|}=1}^D \sigma_{j_1,\bar{\iota}_1}(X_1) \cdots \sigma_{j_{|\iota|_1},\bar{\iota}_{|\iota|_1}}(X_1) \frac{\partial^{|\iota|_1}}{\partial(x_{j_1}, \dots x_{j_{|\iota|_1}})} y(X_1),$$

as Δ tends to zero. By a first-order Taylor expansion of

$$\frac{\partial^{|\iota|_1}}{\partial(x_{j_1}, \dots x_{j_{|\iota|_1}})} y(X_2)$$

around X_1, this convergence will be of order Δ in $\mathcal{L}^2(\Omega, \mathcal{F}, P)$, if b and σ are bounded.

2.2 'Regression now', 'regression later', and 'regression anytime'

Our overall goal is to design an efficient simulation-based approximation $\hat{E}[\mathcal{H}_{\iota,\Delta}(\xi) y(X_2)|X_1]$ to $E[\mathcal{H}_{\iota,\Delta}(\xi) y(X_2)|X_1]$, which also converges in $\mathcal{L}^2(\Omega, \mathcal{F}, P)$ at the order Δ. Before we introduce and explain our new algorithm in Section 3 below, we first briefly discuss the benefits and limitations of two existing least-squares regression approaches, which are sometimes called 'regression now' and 'regression later'. To this end, we consider the problem of approximating the regression function

$$z(x) = E\left[\frac{\xi^2 - 1}{\Delta} y(X_1 + \sqrt{\Delta}\xi) \middle| X_1 = x \right], \quad x \in \mathbb{R}, \tag{4}$$

where X_1 and ξ are independent and standard normal. This example corresponds to (1) with $D = 1$, $\iota = 2$, $b = 0$, and $\sigma = 1$. Thus, by (3), $z(x)$ approximates the second derivative $y''(x)$ of y, which we assume to be of class $\mathcal{C}_b^{Q+1}(\mathbb{R})$. As a main emphasis is on variance issues, we only discuss the notationally simpler 1D case in this subsection.

2.2.1 'Regression now'

The terminology 'regression now' can be traced to the paper [16] on the Monte Carlo approximation of optimal stopping problems and just describes the standard situation of linear least-squares estimates in statistical learning, but with simulated data. This least-squares Monte Carlo approach was popularized in financial engineering in [30] and thoroughly analyzed in the framework of backward stochastic differential equations in [29].

In the 'regression now' approach one chooses a set of basis functions

$$\eta(x) = (\eta_1(x), \dots, \eta_K(x)),$$

which we think of as a row vector. One then generates independent samples $(X_{1,l}, \xi_l)$, $l = 1, \dots, L$, and solves the linear least-squares problem

$$\hat{\alpha}^{L,\,\mathrm{now}} = \operatorname*{arginf}_{\alpha \in \mathbb{R}^K} \frac{1}{L} \sum_{l=1}^{L} \left| \frac{\xi_l^2 - 1}{\Delta} y(X_{1,l} + \sqrt{\Delta}\xi_l) - \eta(X_{1,l})\alpha \right|^2 \tag{5}$$

(In the case of multiple minimizers, one can choose e.g. the one with the minimal Euclidean norm). One then approximates the regression function z by

$$\hat{z}^{L,\,\mathrm{now}}(x) = [\eta(x)\hat{\alpha}^{L,\,\mathrm{now}}]_B,$$

where truncation takes place at a level B, which is any upper bound for the supremum norm of the regression function m. For instance, in our case, one can choose, B to be any upper bound of $\|y''\|_\infty$ by (3).

The resulting estimate is, thus, a linear combination of the basis functions, truncated at level B. This truncation is the standard way to come up with a 'stable' estimate in situations where (say, due to an unfavorable realization of the sample) the least-squares regression problem in (5) is ill-conditioned.

If one thinks of X_1 and $X_2 = X_1 + \sqrt{\Delta}\xi$ as modeling a system at two time points 1 ('now') and 2 ('later'), the phrase 'regression now' simply emphasizes that the basis functions only depend on X_1.

According to Theorem 11.3 in [21], the $\mathcal{L}^2$-error for this 'regression now'-estimate decomposes into the sum of a 'projection error' and a 'statistical error', which are of the form

$$\text{projection error} \;=\; \sqrt{\inf_{\alpha \in \mathbb{R}^K} E[|z(X_1) - \eta(X_1)\alpha|^2]} \tag{6}$$

$$\text{statistical error} \;=\; \sqrt{(\sup_{x \in \mathbb{R}^D} \mathrm{Var}(Z|X_1 = x) + B^2)\, \frac{(\log(L)+1)K}{L}}, \tag{7}$$

where

$$Z = \frac{\xi^2 - 1}{\Delta} y(X_1 + \sqrt{\Delta}\xi).$$

is the regressand.

We now assume that $Q \geq 3$. Recall that $z(x) = y''(x) + \mathcal{O}(\Delta)$ and that y'' is of class $\mathcal{C}_b^{Q-1}$. By a Taylor expansion, we can achieve a projection error of the same order Δ, applying a basis of local monomials of degree up to $Q-2$ on subintervals of length $\Delta^{1/(Q-1)}$ of $[-R_\Delta, R_\Delta]$ as basis functions. Here, R_Δ denotes the $(1-\Delta^2)$-quantile of the standard normal distribution. This results in a total number of basis functions of the order $\Delta^{-1/(Q-1)}$ (up to a log-factor). Due to the variance explosion of the second derivative weight, as $\Delta \to 0$, one requires a number of simulations L of the order $\Delta^{-(4+\frac{1}{Q-1})}$ (up to

a log-factor) to make the statistical error also converge at the order Δ. So, the 'regression now'-algorithm approximates y'' at a rather slow rate of $L^{\frac{-1}{4+\frac{1}{Q+1}}}$ in the number of simulated samples, even in this one-dimensional setting.

Finally, consider the cost for evaluating this 'regression now'-estimate $\hat{z}^{L,\text{now}}$ of z at a single point x_0. As the basis functions are local polynomials of a fixed degree, this evaluation cost is proportional to determining, in which of the $\lceil 2R_D\Delta^{-1/(Q-1)}\rceil$ intervals x_0 is located, i.e. of order $\log(\Delta^{-1})$.

Remark 2.1. In the 'regression now'-setting, the variance can be reduced by the construction of control variates based on some preliminary approximations of

$$E\left[y(X_1+\sqrt{\Delta}\xi)\middle| X_1=x\right] \text{ and } E\left[\frac{\xi}{\Delta^{1/2}}y(X_1+\sqrt{\Delta}\xi)\middle| X_1=x\right],$$

see [1]. Their heuristics suggests, that, in the situation described above, the rate of convergence of the 'regression now' algorithm could possibly be improved to $L^{\frac{-1}{2+\frac{1}{Q+1}}}$ in the number of samples, almost reaching the convergence behavior of classical Monte Carlo simulation if the problem is sufficiently smooth.

2.2.2 From 'regression later' to 'regression anytime'

The 'regression later' approach was suggested in [16] in the context of optimal stopping, and was later on applied to backward stochastic differential equations in [8] under the name 'martingale basis method', to insurance liability modeling [34], and to discrete time stochastic control in [4], among others. To the best of our knowledge, the first 'regression later' algorithm with stochastic derivative weights is due to [8]. In contrast to 'regression now', the basis functions in the 'regression later' approach depend on $X_2 = X_1 + \sqrt{\Delta}\xi$ and not on X_1.

Precisely, one again chooses a set of basis functions $\eta(x) = (\eta_1(x), \dots, \eta_K(x))$, and generates independent samples $(X_{1,l}, \xi_l)$, $l = 1, \dots, L$ (which are assumed to be independent of (X_1, X_2)). After solving the linear least-squares problem

$$\hat{\alpha}^{L,\text{later}} = \operatorname*{arginf}_{\alpha\in\mathbb{R}^K} \frac{1}{L}\sum_{l=1}^{L}\left|y(X_{1,l}+\sqrt{\Delta}\xi_l) - \eta(X_{1,l}+\sqrt{\Delta}\xi_l)\alpha\right|^2, \tag{8}$$

one thinks of $\eta(X_2)\hat{\alpha}^{L,\text{later}}$ as an approximation of $y(X_2)$ and hence approximates the regression function z by

$$\hat{z}^{L,\text{later}}(x) = E\left[\frac{\xi^2-1}{\Delta}\eta(X_2)\hat{\alpha}^{L,\text{later}}\middle| (X_{1,l},\xi_l)_{l=1,\dots,L}, X_1=x\right] = \tilde{\eta}(x)\hat{\alpha}^{L,\text{now}}, \tag{9}$$

where the entries of $\tilde{\eta}$ are given by

$$\tilde{\eta}_k(x) := E\left[\frac{\xi^2-1}{\Delta}\eta_k(X_1+\sqrt{\Delta}\xi)\middle| X_1=x\right] = E\left[\frac{\xi^2-1}{\Delta}\eta_k(x+\sqrt{\Delta}\xi)\right]. \tag{10}$$

Note that the linear structure of the regression estimate $\eta(x)\hat{\alpha}^{L,\text{later}}$ of $y(x)$ is crucial for the closed-form computation of the conditional expectations in (9). Hence, in contrast to the 'regression now' case, no a-posteriori truncation $[\eta(x)\hat{\alpha}^{L,\text{later}}]_B$ can be applied to stabilize the empirical regression.

The obvious advantage of the 'regression later' approach is that the regression problem (8) has no noise in the dependent variable, i.e. the variance of $y(X_2)$ conditionally on

X_2 is zero. Moreover, the expectations involving the high-variance stochastic derivative weight are computed in closed form. Hence, one can hope that this approach is not subject to the statistical error in its classical form and leads to a tremendous variance reduction effect compared to 'regression now', in particular in the presence of derivative weights. Indeed, consider the extreme case, where the basis consists of one function η_1 only and this function equals y. Then a single sample $(X_{1,1}, \xi_1)$ is sufficient to find the optimal parameter $\hat{\alpha}^{1,\,\mathrm{later}} = 1$ and, hence, to obtain $z(x) = \hat{z}^{L,\,\mathrm{later}}(x)$ for every $x \in \mathbb{R}$.

The only paper, which we are aware of, which theoretically explores this potential benefit of 'regression later' is the unpublished preprint [9]. In order to accommodate to the setting of their Section 4, we now additionally assume that $y \in \mathcal{C}_b^2(\mathbb{R})$ has compact support, say $[-R, R]$. Moreover, we neglect log-factors in the subsequent analysis. Following [9], we decompose $[-R, R]$ into Δ^{-1} subintervals which are hit by X_2 with equal probability. On each subinterval monomials up to degree 1 are applied, and this basis consisting of locally linear functions is orthonormalized with respect to the law of X_2. In the special case of this subsection, all closed-form computations required in (10) are available (expressed via moments of a Gaussian random variable conditioned on an interval). Taking the number of samples L proportionally to Δ^{-2}, the error between $\eta(X_2)\hat{\alpha}^{L,\,\mathrm{later}}$ and $y(X_2)$ is of order Δ^2 according to Section 4 in [9]. By Hölder's inequality,

$$\begin{aligned} & |\hat{z}^{L,\,\mathrm{later}}(x) - z(x)| \\ \leq\ & E[|\xi^2 - 1|^2]^{1/2} \Delta^{-1} E\left[|y(X_2) - \eta(X_2)\hat{\alpha}^{L,\,\mathrm{later}}|^2 \Big| (X_{1,l}, \xi_l)_{l=1,\dots,L}, X_1 = x \right]^{1/2}, \end{aligned}$$

and, thus, the resulting error between z and the 'regression later'-estimate $\hat{z}^{L,\,\mathrm{later}}$ is of the order Δ as required. Hence, the 'regression later' algorithm converges to z at a rate of $L^{-1/2}$ in the number of samples and beats 'regression now'. If we stabilize the 'regression later' approach by our brute force SVD truncation, then by Theorem 4.9 below, we can achieve an error of the order Δ with $L = \Delta^{-1}$ samples (up to a log-factor), and, thus, further improve the rate of convergence to L^{-1} in the number of samples. Theorem 4.9 also covers the case of higher order local polynomials as basis functions (not treated in [9]), if y has additional smoothness: If $y \in \mathcal{C}^{Q+1}(\mathbb{R})$ with compact support, the 'regression later' algorithm with brute-force SVD converges to z as $L^{-(Q+1)/2}$ in the number of samples.

Note, however, that in the 'regression later' approach, the resulting estimator for z is a linear combination of the functions $\tilde{\eta}_k(x)$ in (10). These functions will typically have a global support, even if the basis functions η_k, which were applied for the regression, are supported on the small subintervals. Hence, the evaluation at a single point grows proportionally to the number of intervals into which $[-R, R]$ is decomposed. This is a drawback when compared to the use of local basis functions in the 'regression now' setup.

The most striking disadvantage of the 'regression later' algorithm is that one cannot run it for any generic choice of the basis functions, as one has to apply basis functions for which the expectations in (10) are available in closed form (or can be efficiently approximated). A partial remedy is to apply basis functions $\eta_k(x_1, x_2)$, which depend on (X_1, X_2) as in the 'stochastic grid bundling method' [25, 12] or in the LSMC algorithm of [6]. We call such an approach 'regression anytime'. Here, one requires closed-form expressions for the expectations

$$\tilde{\eta}_k(x) := E\left[\frac{\xi^2 - 1}{\Delta} \eta_k(x_1, x_1 + \sqrt{\Delta}\xi) \right].$$

If η is a polynomial in X_2 localized one step earlier, i.e. of the form $\eta(x_1, x_2) = \mathbf{1}_{[a,b]}(x_1)\mathcal{P}(x_2)$, such expressions are available in the setting of this subsection but also in many other models.

Our new 'regression anytime'-algorithm RAWBFST (Algorithm 3.3) applies such polynomials localized one time step earlier as basis functions in a much more general setting, and, crucially, combines it with a change of measure of the sampling distribution and with brute-force SVD truncation. In this way, we end up with an estimator which combines all the advantages of 'regression now' and 'regression later':

- generic applicability to Euler approximations of SDEs in contrast to 'regression later';
- the same (new) fast rates of convergence as in 'regression later' with SVD truncation;
- logarithmic evaluation costs as in 'regression now'.

3 RAWBFST: Algorithm, convergence result, and numerical examples

3.1 The algorithm and its convergence behavior

In this section, we introduce and discuss RAWBFST, our new algorithm for the approximation of $E[\mathcal{H}_{\iota,\Delta}(\xi)y(X_2)|X_1]$. It is a 'regression anytime'-type algorithm, but it also relies on a change of measure of the law of X_1 and employs stratification, similarly to the 'regression now'-algorithm for backward stochastic differential equations in [18]. In order to stabilize the empirical regression, we truncate the singular value decomposition of the empirical regression matrix, but not in the classical way, see Remark 4.2 below.

On the law of X_1 and the coefficients of the Euler scheme, we impose the following assumptions:

Assumption 3.1. *The law μ_1 of X_1 has a density f with respect to the Lebesgue measure such that the 'Aronson type' estimate*

$$f(x) \leq \frac{C_{1,f}}{(2\pi C_{2,f})^{D/2}} \exp\left\{\frac{-|x|_2^2}{2C_{2,f}}\right\}, \quad x \in \mathbb{R}^D,$$

is satisfied for constants $C_{1,f}, C_{2,f} > 0$. Moreover, b and σ are bounded, i.e. there is a constant $C_{b,\sigma} > 0$ such that

$$\sup_{u\in\mathbb{R}^D} (|b(u)|_2 + \|\sigma(u)\|_2) \leq C_{b,\sigma}.$$

Remark 3.2. The situation which we have in mind is the following one: $X_1 = X^e_{t_{i_0}}$ for some i_0, where

$$X^e_{t_{i+1}} = X^e_{t_i} + \bar{b}(t_i, X^e_{t_i})\Delta + \bar{\sigma}(t_i, X^e_{t_i})\sqrt{\Delta}\xi_i, \quad X^e_0 = X_0,$$

is an Euler scheme approximation to the stochastic differential equation

$$dX(t) = \bar{b}(t, X(t))dt + \bar{\sigma}(t, X(t))dW(t), \quad X(0) = X_0, \quad t \in [0, T].$$

Here, of course, $t_i = i\Delta$ and (ξ_i) is an i.i.d. family of D-dimensional vectors of independent standard normal variables. Suppose $\bar{b} : [0, T] \times \mathbb{R}^D \to \mathbb{R}^D$ is measurable and bounded, $\bar{\sigma} : [0, T] \times \mathbb{R}^D \to \mathbb{R}^{D\times D}$ is measurable, Hölder continuous in space (uniformly in time), and $\bar{\sigma}\bar{\sigma}^\top$ is uniformly elliptic. Moreover, assume that X_0 is independent of (ξ_i) and Gaussian with mean vector x_0 and covariance matrix $\Sigma_0 \geq c_0\mathbb{I}_D$ (where $\mathbb{I}_D$ is the identity matrix and $c_0 \geq 0$, i.e. Σ_0 may be degenerate). Then, by an application of Theorem 2.1 in [28], the Aronson estimate in Assumption 3.1 holds with $C_{2,f} = C'_{2,f}t_{i_0} + c_0$ for some constant $C'_{2,f} > 0$.

Before we precisely state the algorithm, let us first explain its several steps in a more intuitive way: In the course of Steps 1–4 of the algorithm, we construct an approximation $\hat{y}(X_1, X_2; \Theta)$ of the function $y(X_2)$. The function $\hat{y}(x_1, x_2; \Theta)$ is a linear combination of polynomials in x_2, which are localized in the x_1-variable (i.e. 'one time step earlier'). The approximation depends on a randomly generated sample $\Theta = (U_{\mathbf{i},l}, \xi_{\mathbf{i},l})$. The first two steps of the algorithm are preparations. In Step 1, a cubic partition $(\Gamma_{\mathbf{i}})$ of some subset $\Gamma \subset \mathbb{R}^D$ for the localization in the x_1-variable is constructed. In Step 2, we provide a suitable basis of the space of polynomials of degree at most Q in terms of Legendre polynomials. With these polynomials, we define our basis functions of the type 'polynomials localized one time step earlier'. In Steps 3–4, an empirical regression with SVD truncation is performed to compute the coefficients for $\hat{y}(x_1, x_2; \Theta)$. Here, we first change measure from the true distribution of X_1 to the uniform distribution on Γ and then stratify the uniform distribution on Γ on the cubic partition $(\Gamma_{\mathbf{i}})$. This change to a uniform distribution is in line with our choice of Legendre polynomials for the basis functions as these are the orthogonal polynomials for the uniform distribution. The sampling of X_2 in Step 3 involves an additional truncation of the Gaussian innovations at some level r_2. In the final Step 5, the algorithm returns our estimator $\hat{z}(x)$ for $E[\mathcal{H}_{\iota,\Delta}(\xi) y(X_2) | X_1 = x]$. This estimator is simply the closed-form expression for the conditional expectation

$$E[\mathcal{H}_{\iota,\Delta}([\xi]_{r_2}) \hat{y}(X_1, X_2^{(\Delta,r_2)}; \Theta) | \Theta, X_1 = x],$$

where (X_1, ξ) is independent of the sample Θ and $X_2^{(\Delta,r_2)}$ is the one-step Euler scheme (2) starting from X_1 with step size Δ and with the truncated Gaussian innovation $[\xi]_{r_2}$ in place of ξ.

Algorithm 3.3. *Input:* function $y : \mathbb{R}^D \to \mathbb{R}$, constants $L \in \mathbb{N}$, $Q \in \mathbb{N}$, $\Delta > 0$, $\iota \in \mathbb{N}_0^D$, $\tau \in (0,1)$, $\gamma_{\text{cube}} \in (0, 1/2)$, $c_{\text{cube}}, c_{1,\text{trunc}}, c_{2,\text{trunc}}, \gamma_{1,\text{trunc}}, \gamma_{2,\text{trunc}} > 0$.

- Step 1: Construction of the cubic partition for X_1 ('now').

  Let $h = c_{\text{cube}} \Delta^{\gamma_{\text{cube}}}$ and $r_1 = \sqrt{C_{2,f} \chi_D^2(c_{1,\text{trunc}}\, \Delta^{\gamma_{1,\text{trunc}}})}$. For every multi-index $\mathbf{i} = (i_1, \ldots, i_D) \in \mathbb{Z}^D$, consider the cube

  $$\Gamma_{\mathbf{i}} = \prod_{d=1}^{D} (h i_d, h(i_d + 1)]$$

  and let

  $$I = I_\Delta = \left\{ \mathbf{i} \in \mathbb{Z}^D;\ \Gamma_{\mathbf{i}} \cap \{ x \in \mathbb{R}^D;\ |x|_2 \le r_1 \} \neq \emptyset \right\}.$$

  Write $\Gamma = \cup_{\mathbf{i} \in I} \Gamma_{\mathbf{i}}$ and $a_{\mathbf{i}}$ for the center of the $\mathbf{i}$th cube.

- Step 2: Construction of the local polynomials for X_2 ('later'). Denote by $\mathcal{L}_q : \mathbb{R} \to \mathbb{R}$ the Legendre polynomial of degree q, which is normalized such that $\mathcal{L}_q(1) = 1$, i.e.

  $$\mathcal{L}_q(x) = \frac{1}{2^q} \sum_{r=0}^{\lfloor q/2 \rfloor} \frac{(-1)^r (2q - 2r)!}{r!(q-r)!(q-2r)!} x^{q-2r}.$$

  For any multi-index $\mathbf{j} \in \mathbb{N}_0^D$ such that $|\mathbf{j}|_1 \le Q$, let

  $$p_{\mathbf{j}}(x) = \prod_{d=1}^{D} \sqrt{2 j_d + 1}\, \mathcal{L}_{j_d}(x_d), \quad x = (x_1, \ldots, x_D).$$

Let $K = \binom{D+Q}{D}$. For every $\mathbf{i} \in I$ denote by $\eta_{\mathbf{i},k}$, $k = 1, \dots, K$, any fixed ordering of the polynomials

$$x \mapsto p_{\mathbf{j}}\left(\frac{x - a_{\mathbf{i}}}{h/2}\right), \quad \mathbf{j} \in \mathbb{N}_0^D,\ |\mathbf{j}|_1 \le Q.$$

- Step 3: Construction of the empirical regression matrices on the cubes.

  For every $\mathbf{i} \in I$, sample independent copies $(U_{\mathbf{i},l}, \xi_{\mathbf{i},l})_{l=1,\dots,L}$ where $U_{\mathbf{i},l}$ is uniformly distributed on $\Gamma_{\mathbf{i}}$ and $\xi_{\mathbf{i},l}$ is multivariate Gaussian with zero mean vector and unit covariance matrix independent of $U_{i,l}$. Let $r_2 = \sqrt{2\log(c_{2,\mathrm{trunc}}\,\Delta^{-\gamma_{2,\mathrm{trunc}}}\log(\Delta^{-1}))}$ and

  $$X_{\mathbf{i},l} = U_{\mathbf{i},l} + b(U_{\mathbf{i},l})\Delta + \sigma(U_{\mathbf{i},l})\sqrt{\Delta}\,[\xi_{\mathbf{i},l}]_{r_2}.$$

  Build the empirical regression matrices

  $$A_{\mathbf{i}} = (\eta_{\mathbf{i},k}(X_{\mathbf{i},l})_{l=1,\dots,L;\,k=1,\dots K}$$

- Step 4: Least-squares interpolation with brute-force SVD truncation.

  For every $\mathbf{i} \in I$ perform a singular value decomposition of $A_{\mathbf{i}}^\top$:

  $$A_{\mathbf{i}}^\top = \mathcal{U}\mathcal{D}\mathcal{V}, \quad \mathcal{U} \in O(K),\ \mathcal{V} \in O(L),$$

  where $\mathcal{D}$ is the $K \times L$-matrix which has the singular values $s_1 \ge s_2 \ge \cdots \ge s_K \ge 0$ of $A_{\mathbf{i}}$ on the diagonal and has zero entries otherwise. If $s_K^2 \ge \tau L$,

  $$\alpha_{L,\mathbf{i}} = \mathcal{U}\mathcal{D}^\dagger\mathcal{V}\,(y(X_{\mathbf{i},1}), \dots, y(X_{\mathbf{i},L}))^\top$$

  where $\mathcal{D}^\dagger$ is the $K \times L$-matrix which has $s_1^{-1}, \dots, s_K^{-1}$ on the diagonal and has zero entries otherwise (i.e., the pseudoinverse of $\mathcal{D}^\top$). Otherwise let

  $$\alpha_{L,\mathbf{i}} = 0 \in \mathbb{R}^K.$$

- Step 5: Return

  $$\hat{z}(x) := \sum_{\mathbf{i}\in I} \mathbf{1}_{\Gamma_{\mathbf{i}}}(x) \sum_{k=1}^{K} \alpha_{L,\mathbf{i},k} E[\eta_{\mathbf{i},k}(x + b(x)\Delta + \sigma(x)\sqrt{\Delta}[\xi]_{r_2})\mathcal{H}_{\iota,\Delta}([\xi]_{r_2})]$$

  as an approximation of $E[\mathcal{H}_{\iota,\Delta}(\xi)y(X_2)|X_1 = x]$.

Remark 3.4. Note that

$$\eta_{\mathbf{i},k}(x + b(x)\Delta + \sigma(x)\sqrt{\Delta}[\xi]_{r_2})\mathcal{H}_{\iota,\Delta}([\xi]_{r_2})$$

is a polynomial in $[\xi_1]_{r_2}, \dots [\xi_D]_{r_2}$, whose coefficients depend on x. By independence,

$$E\left[\prod_{d=1}^{D}[\xi_d]_{r_2}^{q_d}\right] = \prod_{d=1}^{D} E\left[[\xi_1]_{r_2}^{q_d}\right].$$

Hence, the expectation in Step 5 can be computed in closed form by the following recursion formula for the moments $m_{q,r} = E[[\xi_1]_r^q]$ of the truncated standard normal distribution:

$$\begin{aligned} m_{2q,r} &= (2q-1)m_{2q-2,r} + 2r^{2q-2}(r^2 - 2q + 1)(1 - \Phi(r)) - 2r^{2q-1}\varphi(r), \quad m_{0,r} = 0, \\ m_{2q-1,r} &= 0. \end{aligned}$$

The following theorem provides the error analysis for Algorithm 3.3. Its proof is postponed to Section 5.

Theorem 3.5. *Fix $\rho \in \mathbb{N}$. Suppose $y \in \mathcal{C}_b^{Q+1}(\mathbb{R}^D)$ for some $Q \geq |\iota|_1 + \rho$. Compute $\hat{z}$ via Algorithm 3.3 with*

$$\gamma_{\text{cube}} = \frac{\rho + |\iota|_1}{2(Q+1)}, \quad \gamma_{1,\text{trunc}} = \rho, \quad \gamma_{2,\text{trunc}} = 1.5(|\iota|_1 + \rho),$$

and

$$\begin{aligned} L &= L_\Delta = \lceil \rho\, c_{1,\text{paths}} \log(c_{2,\text{paths}}\, \Delta^{-1}) \rceil \\ \tau &\in \left(0, \quad 1 - \left(\frac{c^*_{\text{paths}}(Q,D)}{c_{1,\text{paths}}}\right)^{1/2}\right) \end{aligned}$$

for constants

$$c_{2,\text{paths}} > 0, \quad c_{1,\text{paths}} > c^*_{\text{paths}}(Q,D) := \frac{2}{3} + \frac{8}{3} \sum_{\mathbf{j} \in \mathbb{N}_0^D; |\mathbf{j}|_1 \leq Q} \prod_{d=1}^{D} (2j_d + 1).$$

Then there are constants $C > 0$ and $\Delta_0 > 0$ (depending on all the constants, including D, Q, $|\iota|_1$, τ, and the $\mathcal{C}_b^{Q+1}$-norm of y) such that for every $\Delta \leq \Delta_0$

$$E\left[\int_{\mathbb{R}^D} |E[\mathcal{H}_{\iota,\Delta}(\xi) y(X_2) | X_1 = x] - \hat{z}(x)|^2 \mu_1(dx)\right] \leq C \log(\Delta^{-1})^{D/2} \Delta^\rho.$$

Remark 3.6. The cost for performing the $|I_\Delta|$ singular value decompositions with L_Δ samples and a fixed number of basis functions (here: $\binom{D+Q}{D}$) is up to log-factors of the order $\Delta^{-D\gamma_{\text{cube}}}$. Hence, the $\mathcal{L}^2(P \otimes \mu_1)$-complexity of the algorithm is for $\rho = 2$ (up to log-factors) $\Delta^{-D(1+|\iota|_1/2)/(Q+1)}$, where D is the space dimension, $|\iota|_1$ corresponds to the order of the partial derivative, and $(Q+1)$ is the smoothness parameter of the problem.

3.2 Numerical Illustrations

In this subsection, we provide two numerical illustrations of Algorithm 3.3. In the first illustration, we approximate the second derivative of a univariate function similar to the setting of Section 2.2. This is a direct application of Theorem 3.5. The second illustration is option pricing in the uncertain volatility model, a challenging reference problem in the literature on second-order BSDEs and fully non-linear partial differential equations [20, 1, 26]. This is an exploratory study confirming the excellent performance of our algorithm beyond the setting that is strictly covered by our theoretical analysis. The problem here is typical of the applications we envision for our algorithm, as many layers of conditional expectations need to be iterated and we need highly accurate approximations of functions together with their derivatives.

3.2.1 Approximating a second derivative

We wish to apply Algorithm 3.3 to approximate

$$z(x) = E\left[\frac{\xi^2 - 1}{\Delta} y(X_1 + \sqrt{\Delta}\xi) \middle| X_1 = x\right], \quad x \in \mathbb{R}, \tag{11}$$

where X_1 and ξ are independent and standard normal. As in Section 2.2, this corresponds to (1) with $D = 1$, $\iota = 2$, $b = 0$, and $\sigma = 1$. For the function y, we consider $y(x) = x^2 \exp(-x^2/2)$. The function z thus approximates the second derivative y'' of y with $y''(x) = z(x) + O(\Delta)$. Specifically, we can benchmark the output of our algorithm against the closed-form expressions $y''(x) = (x^4 - 5x^2 + 2)\exp(-x^2/2)$ and

$$z(x) = \frac{x^4 - (5 + 4\Delta - \Delta^2)x^2 + 2 + 3\Delta - \Delta^3}{(1+\Delta)^{\frac{9}{2}}} \exp\left(-\frac{x^2}{2(1+\Delta)}\right).$$

In line with Theorem 3.5, our main error criterion is the root mean squared error

$$\mathcal{E}(\Delta, \rho) := \hat{E}\left[\int_{\mathbb{R}} |z(x) - \hat{z}(x|\Theta, \Delta, \rho)|^2 \mu_1(dx)\right]^{\frac{1}{2}}$$

where $\hat{z}(\cdot|\Theta, \Delta, \rho)$ denotes the output of one run of a MATLAB implementation of Algorithm 3.3 in dependence on the Monte Carlo sample Θ, the step size parameter Δ and the convergence rate parameter ρ. $\hat{E}$ denotes an empirical average over 100 runs of the algorithm, i.e., over 100 independent realizations of Θ. μ_1 is the standard normal distribution of X_1. The interior univariate integral over x is computed using adaptive quadrature as implemented in MATLAB's `integral` command. As a reference, we also compute the discretization error between y'' and z,

$$\bar{\mathcal{E}}(\Delta) := \left(\int_{\mathbb{R}} |z(x) - y''(x)|^2 \mu_1(dx)\right)^{\frac{1}{2}}.$$

In our numerical experiments, we vary $\rho = 2, 3, 4$ and $\Delta = 2^{-n}$, $n = 3, \dots 14$. The standard normal distribution for X_1 implies $C_{1,f} = C_{2,f} = 1$. The polynomial degree Q, we set as $Q = \rho + \iota + 1 = \rho + 3$. In dimension $D = 1$, a direct computation gives $c^*_{\text{paths}}(Q, 1) = \frac{2}{3} + \frac{8}{3}(Q+1)^2$. Accordingly, we choose $c_{1,\text{paths}} = 1.1\, c^*_{\text{paths}}(Q, 1)$ and $c_{2,\text{paths}} = 1$. The parameters γ_{cube}, $\gamma_{1,\text{trunc}}$, $\gamma_{2,\text{trunc}}$ and L are then simply chosen using the formulas given in Theorem 3.5 while τ is chosen as the midpoint of the admissible interval given there which implies $\tau = (1 - 1.1^{-0.5})/2 = 0.0233$. The three remaining parameters that scale the truncation levels and the density of cubes we set as $c_{\text{cube}} = c_{1,\text{trunc}} = c_{2,\text{trunc}} = 5$.

The three black curves in Figure 1 plot $\log_{10}(\mathcal{E}(\Delta, \rho))$ against $\log_{10}(\Delta^{-1})$ for $\rho = 2, 3, 4$. Each line is contrasted against a gray line through the final data point with slope equal to the theoretical convergence rate of $\rho/2$ guaranteed by Theorem 3.5. We observe that the empirical decay is broadly in line with theory but slightly faster. For comparison, the dotted gray line depicts the discretization error $\log_{10}(\bar{\mathcal{E}}(\Delta))$ which vanishes at a rate close to 1 as expected. Consequently, for $\rho = 2$ the approximation error $\mathcal{E}(\Delta, \rho)$ of our algorithm is of a similar magnitude as the discretization error $\bar{\mathcal{E}}(\Delta)$ for all considered values of Δ^{-1}.

For selected values of Δ^{-1}, Table 1 provides further details like number of cubes, number of samples per cube and run times. For sufficiently small Δ, run times should behave (up to log-factors) like $\Delta^{-\gamma_{\text{cube}}}$. In our implementation, we have $\gamma_{\text{cube}} = \frac{\rho+2}{2\rho+8}$ and essentially the number of cubes grows like $\Delta^{-\gamma_{\text{cube}}}$ while the number of samples per cube only depends on Δ logarithmically. Thus, thinking optimistically, increasing Δ^{-1} by a factor 8 should increase the total number of samples and thus run time by factors of 2, 2.1 and 2.2 for $\rho = 2, 3, 4$. Inspecting the actual run times in the table shows that this type of reasoning is too optimistic in our situation as it ignores logarithmic factors and rounding effects. Comparing, e.g., the columns associated with $\Delta^{-1} = 1024$ and $\Delta^{-1} = 8192$, we see that the numbers of cubes $|I_\Delta|$ increase by factors 2.2, 2.5, and 2.52 rather than 2, 2.1 and 2.2. Similarly, due to logarithmic growth, the number of samples per cube L_Δ is far from

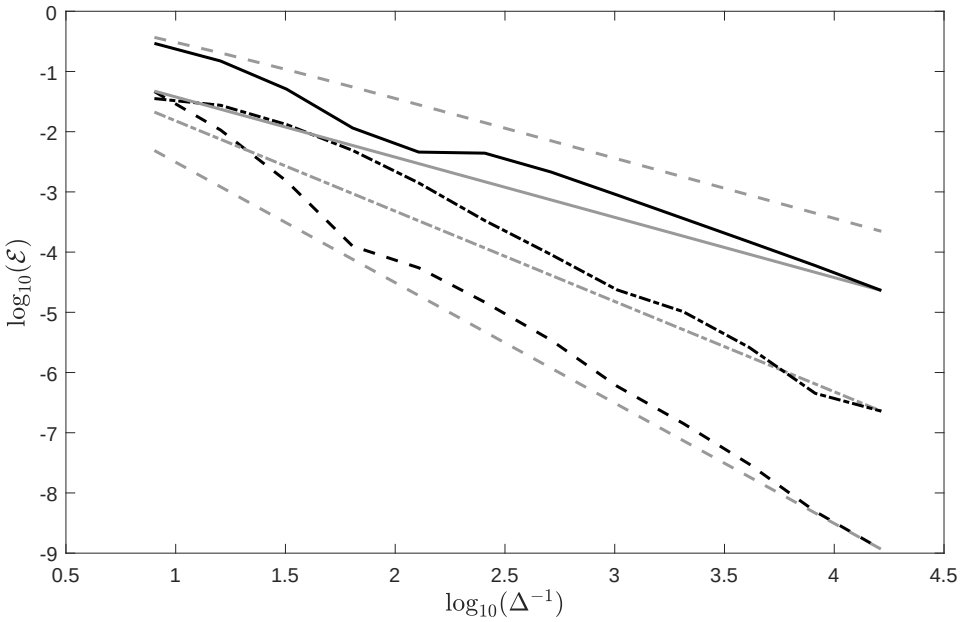

Figure 1: Approximation errors against Δ^{-1} in a $\log_{10}$-$\log_{10}$-plot. The black curves correspond to $\mathcal{E}(\cdot, 2)$ (solid line), $\mathcal{E}(\cdot, 3)$ (dash-dotted line) and $\mathcal{E}(\cdot, 3)$ (dashed line). The corresponding solid gray lines illustrate the theoretical slopes of 1, 1.5 and 2. The dotted gray line depicts the discretization error $\bar{\mathcal{E}}(\cdot)$.

constant, increasing by a factor of about 1.3. Thus, the total number of samples $L_\Delta|I_\Delta|$ increases by factors between 2.86 and 3.28 as Δ^{-1} is increased by a factor 8. As we implement the thin SVD, see Remark 4.2, this total number of samples should behave like run time. Indeed, the relative increases in run times we observe throughout the table are broadly consistent with those in the number of samples (but slightly smaller).

3.2.2 Uncertain Volatility Model

In this section, we apply Algorithm 3.3 to approximate option prices in the Uncertain Volatility Model (UVM) due to [3, 31]. In this model, it is assumed that the volatility process of the stock underlying an option contract is not known for certain but only known to lie in an interval $[\sigma_l, \sigma_h]$. Option prices are then computed as suprema over all admissible volatility processes in the interval. With this modification, the linear Black-Scholes partial differential equation that arises under a known, constant volatility is replaced by the fully non-linear Black-Scholes-Barenblatt equation. This makes pricing in the UVM a challenging problem even in low dimensions that is frequently used to test new algorithms [20, 1, 26, 7].

In this paper, we directly introduce the discrete time, non-linear backward recursion for pricing that arises when the discretization scheme of [14] is applied to the UVM. We refer, e.g., to [7] for a more detailed derivation from the continuous time setting. The time horizon $[0, T]$ is discretized into N subintervals of equal length $\Delta = T/N$ from $t_0 = 0$ to $t_N = T$. At maturity time T, the payoff of a given option contract is known to be some function $y_N : \mathbb{R} \to \mathbb{R}$. In line with the literature, we consider the pricing of a Call spread option which corresponds to the choice

$$y_N(x) = \max\left(0, s_0 e^{(\mu - \frac{1}{2}\sigma_r^2)T + \sigma_r x} - K_1\right) - \max\left(0, s_0 e^{(\mu - \frac{1}{2}\sigma_r^2)T + \sigma_r x} - K_2\right).$$

Δ^{-1}	16	128	1024	8192
$\bar{\mathcal{E}}$	0.2037	0.0280	0.0035	0.0004
$\rho = 2$				
$\mathcal{E}$	0.1493	0.0046	0.0009	$5.97 \cdot 10^{-5}$
$\lvert I_\Delta \rvert$	4	8	20	44
L_Δ	590	1032	1475	1917
run time in s	0.0036	0.0104	0.0277	0.0550
$\rho = 3$				
$\mathcal{E}$	0.0274	0.0014	$2.36 \cdot 10^{-5}$	$4.51 \cdot 10^{-7}$
$\lvert I_\Delta \rvert$	4	12	28	70
L_Δ	1202	2103	3005	3906
run time in s	0.0064	0.0248	0.0613	0.1411
$\rho = 4$				
$\mathcal{E}$	0.0107	$5.49 \cdot 10^{-5}$	$5.94 \cdot 10^{-7}$	$4.92 \cdot 10^{-9}$
$\lvert I_\Delta \rvert$	6	14	38	96
L_Δ	2091	3658	5226	6794
run time in s	0.0142	0.0496	0.1319	0.3746

Table 1: Approximation errors and algorithmic parameters in dependence on Δ and ρ. Run times are for a MATLAB2017a implementation on a Windows desktop PC with an Intel Core i7-6700 CPU with 3.4GHz.

Here, $s_0 > 0$ is the initial stock price, $\mu \in \mathbb{R}$ is the drift under the pricing measure, $K_1, K_2 > 0$ is a pair of strike prices, and σ_r is the so-called reference volatility, a choice parameter in the discretization. Notice that x takes the place of the Brownian motion driving the stock price and not that of the stock price itself. Then, for $i = 1, \dots, N$, price functions y_i and y_{i-1} at times t_i and t_{i-1} are related through the recursion

$$y_{i-1}(x) = G(z_{i-1,0}(x), z_{i-1,1}(x), z_{i-1,2}(x)) \tag{12}$$

where, for $\iota = 0, 1, 2$,

$$z_{i-1,\iota}(x) = E[\mathcal{H}_{\iota,\Delta}(\xi) y_i(X_1 + \sqrt{\Delta}\xi) | X_1 = x] \tag{13}$$

and where $G : \mathbb{R}^3 \to \mathbb{R}$ is given by

$$G(z_0, z_1, z_2) = z_0 + \frac{\Delta}{2}(z_2 - \sigma_r z_1) \left(\frac{\sigma_h^2}{\sigma_r^2} \mathbf{1}_{\{z_2 > \sigma_r z_1\}} + \frac{\sigma_l^2}{\sigma_r^2} \mathbf{1}_{\{z_2 \leq \sigma_r z_1\}} - 1 \right).$$

The quantity of interest is the option price at the initial time and initial value, $y_0(0)$. In the function G, the terms in the round brackets can be interpreted as switching from the reference volatility σ_r to either σ_h or σ_l for the time interval from t_{i-1} to t_i. Which of these two alternatives is chosen depends on the terms z_1 and z_2 which correspond to the first two derivatives. Thus, the pricing recursion depends in a non-linear way on the second derivative, underlining the fact that it is a discretization of a fully non-linear partial differential equation.

Evidently, the difficult part in solving the recursion (12) numerically is the computation of the conditional expectations in (13). This is exactly the problem RAWBFST is designed for. We thus propose the following algorithm for constructing a sequence $(\hat{y}_i)_i$, $i = 0, \dots, N$ of real-valued functions that approximate $(y_i)_i$.

Algorithm 3.7.

- Initialization: $\hat{y}_N \equiv y_N$.
- For $i = N, \dots, 1$:
  - For $\iota = 0, 1, 2$: call Algorithm 3.3 with input $y \equiv \hat{y}_i$ and output $\hat{z}_{i-1,\iota}$.
  - Define $\hat{y}_{i-1}$ via $\hat{y}_{i-1}(x) = G(\hat{z}_{i-1,0}(x), \hat{z}_{i-1,1}(x), \hat{z}_{i-1,2}(x))$.
- Return $\hat{y}_0(0)$.

Remark 3.8. (i) This algorithm is very similar to the usual LSMC algorithms for BSDEs. The one major innovation is to replace Regression Now (as, e.g., in [29]) or Regression Later (as in [8]) by RAWBFST, Algorithm 3.3.

(ii) In line with the formal statement of Algorithm 3.3 above, we call it three times in each time step of Algorithm 3.7, once for every relevant value of ι. In our practical implementation, we exploit that only the evaluation in Step 5 of the algorithm depends on ι. Thus, the first four steps of computing the regression coefficients need to be performed only once.

(iii) As discussed, e.g., in [18], memory usage and scope for parallelization are potential bottlenecks in LSMC algorithms. In both regards, Algorithm 3.7 has excellent properties. Monte Carlo samples are generated only within the calls of Algorithm 3.3. No N-step sample trajectories are produced or stored over time. Moreover, the regressions within each cube can be computed independently. Thus, in principle, one can implement Algorithm 3.3 in such a way that (only) the full set of $(1 + Q)|I|$ regression coefficients from the previous step is communicated to one processor for each of the $|I|$ cubes. This processor simulates L samples and computes and returns the $1 + Q$ coefficients that define the local polynomials on this cube. This results in very modest memory requirements for typical choices of Q, $|I|$ and L.

In order to set the parameters for RAWBFST, we need to make some 'guesses' about the time discretization error of the approximation scheme (12)–(13) to the Black-Scholes-Barenblatt equation and about the error propagation, which results from nesting the RAWBFST approximation of the true conditional expectations backwards in time. For the time-discretization error it is known from [17], that the probabilistic scheme converges at the order Δ in the case of a quasi-linear parabolic PDE, if the coefficient functions are sufficiently smooth and the forward SDE can be sampled without discretization error. Our numerical results below support this convergence behavior of the time discretization error in the UVM test case, although it is not backed by the theoretical error analysis for the non-linear case in [14]. The error propagation for the approximation of the conditional expectations is of the order $\Delta^{-1/2}$ for 'regression now' in the quasi-linear case, see [29]. For the parameter choice of the algorithm, we here assume that the same is true for RAWBFST in the non-linear case. Hence, we may hope for an overall convergence of the order Δ, if the conditional expectations in (13) are approximated to the order $\Delta^{3/2}$ for $\iota = 0$ and to the order Δ for $\iota = 1, 2$. In line with Theorem 3.5 and applying local polynomials of degree up to $Q = 4$, we, thus, set $\gamma_{\text{cube}} = 0.4$, $\gamma_{1,\text{trunc}} = 3$, $\gamma_{2,\text{trunc}} = 6$. We choose a slightly finer space discretization than before, setting $c_{\text{cube}} = 2$ while keeping $c_{1,\text{trunc}} = c_{2,\text{trunc}} = 5$. The above parameter choice implies $c^*_{\text{paths}}(Q, 1) = 67.33$. In our baseline implementation, we then choose $c_{1,\text{paths}} = 1.1\, c^*_{\text{paths}}(Q, 1) = 74.07$, $c_{2,\text{paths}} = 1$ and, thus, $L = \lceil 3 \cdot c_{1,\text{paths}} \log(\Delta^{-1}) \rceil$. As before, we let $\tau = 0.0233$. When we vary $c_{1,\text{paths}}$ in our numerical experiments, we adjust the value of L while keeping everything else (including τ) fixed.

The only parameter we choose adaptively at each step i is the parameter $C_{2,f}$. Since the underlying state process X is a Brownian motion with $x_0 = 0$, the mechanical choice would be to set $C_{2,f} = t_{i-1} = (i-1)\Delta$. This corresponds to the variance of the Brownian motion at time t_{i-1} which takes the role of X_1 in Algorithm 3.3 at step i. However, this would lead to degeneration at time $i = 0$. Intuitively, we need to approximate the functions y_i in a small interval around 0 if we wish to approximate derivatives in 0 well. We thus choose $C_{2,f} = \sigma_0^2 + t_{i-1}$ in our implementation, $\sigma_0^2 = 0.1$. This corresponds to replacing our standard Brownian motion by one that was started in 0 at time $-\sigma_0^2$. For the parameters of the spread option and the UVM, we follow [20] and the subsequent literature, choosing $s_0 = 100$, $\mu = 0$, $\sigma_l = 0.1$, $\sigma_h = 0.2$, $\sigma_r = 0.15$, $T = 1$, $K_1 = 90$ and $K_2 = 110$. In this setting, the continuous-time limit $\Delta \downarrow 0$ of $y_0(0)$ is given by 11.20456 as shown in [41] which provides closed-form pricing formulas for this type of product in the UVM.

Figure 2 is a log-log plot of the mean squared error $\mathcal{E}(\Delta) := \hat{E}\left[(\hat{y}_0(0) - 11.20456)^2\right]^{\frac{1}{2}}$ against the number of time steps $\Delta = 2^{-n}$, $n = 4, \dots 9$ for two instances of our algorithm. Here, the empirical mean $\hat{E}$ denotes an average over 100 independent realizations of $\hat{y}_0(0)$. In the first instance, depicted by the solid line, we choose $c_{1,\text{paths}} = 1.1\, c^*_{\text{paths}}(Q, 1) = 74.07$ as suggested by Theorem 3.5. In the second instance, depicted by the dashed line, we have cut computational costs by setting $c_{1,\text{paths}} = 10$. Both variants of the algorithm converge at a rate which is similar to the expected rate of 1 in the stepsize Δ, which is depicted in gray. While the cheap version of the algorithm has a somewhat higher variance, we note that ultimately it leads to very similar quantitative results as the approximation error is dominated by the bias. Figure 3 gives an analogous plot of $\mathcal{E}(\Delta)$ against the average run time of the algorithm. In this figure, the differences in computational cost between the two implementations become apparent. In our implementation, we have $\gamma_{\text{cube}} = 0.4$. As the number of time steps behaves like Δ^{-1}, we expect run time to behave like $\Delta^{-1.4}$. This suggests a convergence rate of $5/7$ for $\mathcal{E}(\Delta)$ against run time. The two gray lines with slope $-5/7$ in Figure 3 demonstrate that the empirical convergence rate are very much in line with this reasoning.

Table 2 reports further summary statistics such as the mean and standard deviation of $\hat{y}_0(0)$ for the two instances of our algorithm. Together with the two figures, the table confirms that our approximation converges stably towards its limit at a rate that is faster than standard Monte Carlo for the approximation of a single expectation, and is in line with our heuristics. This is in marked contrast to earlier implementations of this example in [20, 1, 7] which show clearly that a stable approximation at $\Delta^{-1} = 512$ should not be taken for granted for regression-based Monte Carlo algorithms.

4 Least-squares interpolation with brute-force SVD truncation

In this section, we analyze the main building block of Algorithm 3.3, namely the least-squares regression with brute-force SVD truncation, when there is no noise in the dependent variable. We also show that (up to log-factors) optimal rates for the interpolation problem with fixed random design can be achieved under suitable assumptions.

4.1 Algorithm and convergence analysis

Suppose X is an $\mathbb{R}^D$-valued random variable with law μ and $Y = y(X)$ for some measurable function $y : \mathbb{R}^D \to \mathbb{R}$. Given i.i.d copies $\mathcal{D} = (X^{(l)}, Y^{(l)})_{l=1,\dots,L}$, our aim is to estimate the function y. This problem can be interpreted as an interpolation problem with fixed random design, see [27, 5].

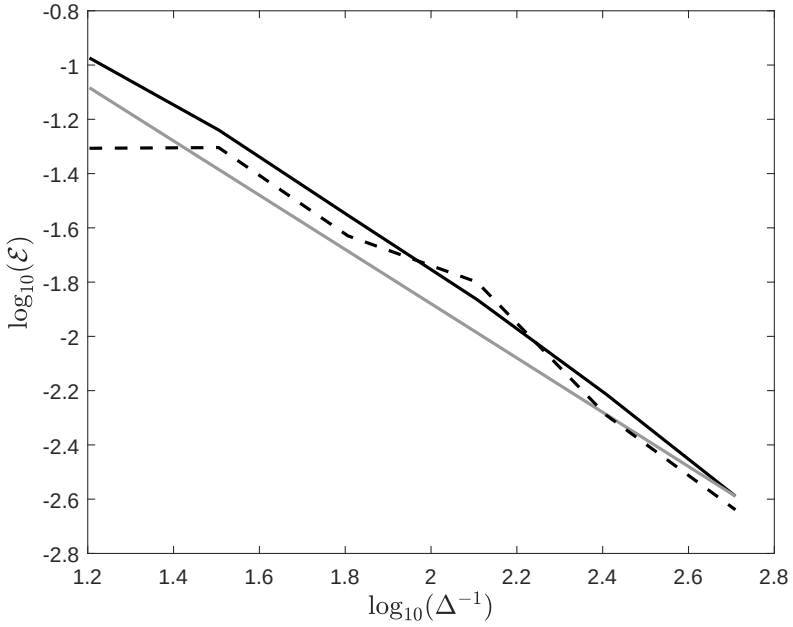

Figure 2: Approximation errors against Δ^{-1} in a $\log_{10}$-$\log_{10}$-plot.

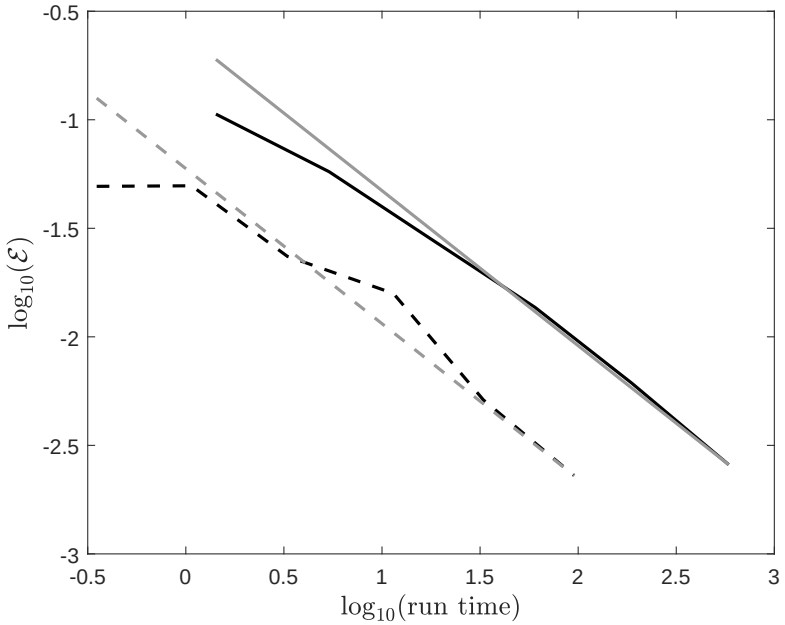

Figure 3: Approximation errors against run time in a $\log_{10}$-$\log_{10}$-plot. Run times are for a MATLAB2017a implementation on a Windows desktop PC with an Intel Core i7-6700 CPU with 3.4GHz.

Δ^{-1}	16	32	64	128	256	512
$c_{1,\text{paths}} = 74.07$						
mean	11.0988	11.1471	11.1767	11.1909	11.1985	11.2020
standard deviation	0.0087	0.0033	0.0019	0.0009	0.0005	0.0003
run time in s	1.42	5.38	18.06	60.30	187.58	585.13
$c_{1,\text{paths}} = 10$						
mean	11.1646	11.1561	11.1823	11.1889	11.1997	11.2025
standard deviation	0.0290	0.0108	0.0075	0.0023	0.0017	0.0011
run time in s	0.35	1.07	3.29	11.43	32.86	95.40

Table 2: Mean and standard deviation of $\hat{y}_0(0)$ across 100 runs of the algorithm and run time for one run for varying $c_{1,\text{paths}}$ and Δ.

We propose a modified least-squares approach, which utilizes a brute-force truncation of the singular value decomposition. To this end, let $\eta(x) = (\eta_1(x), \dots, \eta_K(x))^\top$ denote a vector of basis functions.

Algorithm 4.1. Fix a threshold $\tau > 0$.

- Build the empirical regression matrix

$$A = (\eta_k(X^{(l)}))_{l=1,\dots,L;\ k=1,\dots K}$$

and perform a singular value decomposition of $A^\top$:

$$A^\top = \mathcal{U}\mathcal{D}\mathcal{V}, \quad \mathcal{U} \in O(K),\ \mathcal{V} \in O(L),$$

where $\mathcal{D}$ is the $K \times L$-matrix which has the singular values $s_1 \geq s_2 \geq \cdots \geq s_K \geq 0$ of A on the diagonal and has zero entries otherwise.

- If $s_K^2 \geq L\tau$, let

$$\alpha_L = \mathcal{U}\mathcal{D}^\dagger\mathcal{V}\,(Y^{(1)}, \dots, Y^{(L)})^\top$$

where the pseudoinverse $\mathcal{D}^\dagger$ of $\mathcal{D}^\top$ is the $K \times L$-matrix which has $s_1^{-1}, \dots, s_K^{-1}$ on the diagonal and has zero entries otherwise.

If $s_K^2 < L\tau$, let

$$\alpha_L = 0 \in \mathbb{R}^K.$$

- Return

$$\hat{y}_L(x) := \alpha_L^\top \eta(x) = \sum_{k=1}^{K} \alpha_{L,k}\eta_k(x)$$

as an approximation of y.

Remark 4.2. (i) Truncated singular value decomposition is a popular regularization method for ill-conditioned linear regression problems, see e.g. [22]. One approximates the solution to the regression problem by using the largest t singular values only, i.e. one chooses the coefficient vector

$$\hat{\alpha}_t = \mathcal{U}\mathcal{D}_t^\dagger\mathcal{V}\,(Y^{(1)}, \dots, Y^{(L)})^\top,$$

where $\mathcal{D}_t^\dagger$ is the $K \times L$-matrix, which has $s_1^{-1}, \dots, s_t^{-1}, 0, \dots, 0$ on the diagonal and has zero entries otherwise, and t can be interpreted as a regularization parameter. In contrast, in

our brute-force SVD truncation, we either keep all singular values (if the smallest singular value is sufficiently large), or we completely discard all the singular values (otherwise).

(ii) In practical implementations with $L \gg K$, we recommend replacing the SVD above by the thin SVD ([19], p.72). To this end, consider the SVD $A^\top = \mathcal{U}\mathcal{D}\mathcal{V}$ from above. Denote by $\bar{\mathcal{D}}$ and $\bar{\mathcal{D}}^\dagger$ the $K \times K$ matrices containing the first K columns of $\mathcal{D}$ and $\mathcal{D}^\dagger$ and by $\bar{\mathcal{V}}$ the $K \times L$ matrix containing the first K rows of $\mathcal{V}$. Due to the identities $\mathcal{D}\mathcal{V} = \bar{\mathcal{D}}\bar{\mathcal{V}}$ and $\mathcal{D}^\dagger\mathcal{V} = \bar{\mathcal{D}}^\dagger\bar{\mathcal{V}}$, it suffices to compute $\bar{\mathcal{V}}$ and $\bar{\mathcal{D}}$ when implementing Algorithm 4.1, avoiding the large $L \times L$ matrix $\mathcal{V}$. In MATLAB, this thin SVD is implemented via the 'econ' option in the `svd` command.

For the error analysis, we write

$$R = (E[\eta_k(X)\eta_\kappa(X)])_{k,\kappa=1,\dots,K}.$$

We assume that R has full rank and that we have access to bounds on the extremal eigenvalues of R:

$$0 < \lambda_* \le \lambda_{\min}(R) \le \lambda_{\max}(R) \le \lambda^* < \infty.$$

Theorem 4.3. *Suppose that the basis functions η_k are bounded. Let*

$$m := \sup_{x \in \mathrm{supp}(X)} \sum_{k=1}^{K} |\eta_k(x)|^2$$

and $\tau = (1-\epsilon)\lambda_*$ for some $\epsilon \in (0,1)$. Then,

$$\begin{aligned} & E\left[\int_{\mathbb{R}^D} |y(x) - \hat{y}_L(x)|^2 \mu(dx)\right] \\ \le\ & \left(1 + \frac{\lambda^*}{\lambda_*(1-\epsilon)}\right) \inf_{\alpha \in \mathbb{R}^K} \int_{\mathbb{R}^D} |y(x) - \alpha^\top \eta(x)|^2 \mu(dx) \\ & + 2K \exp\left\{-\frac{3\epsilon^2 L}{6m\lambda^*/\lambda_*^2 + 2\epsilon(m/\lambda_* + \lambda^*/\lambda_*)}\right\} \int_{\mathbb{R}^D} |y(x)|^2 \mu(dx), \end{aligned}$$

where $\hat{y}_L$ is constructed by Algorithm 4.1.

Remark 4.4. (i) A remarkable feature of the algorithm is that for a fixed approximation architecture (i.e., a fixed function basis), the statistical error converges exponentially in the number L of samples.

(ii) If one scales the basis functions by a multiplicative constant $\gamma \neq 0$, then m, $\lambda_{\max}(R)$, and $\lambda_{\min}(R)$ are scaled by the factor γ^2. Hence, the error analysis in the above theorem is invariant against scaling of the basis functions.

(iii) The error analysis in the above theorem is not distribution-free, but it depends on the distribution of X only through bounds on the extremal eigenvalues of the matrix R. It is shown in [27] that the optimal rates for the interpolation problem with a random design, when the samples are drawn from a uniform distribution on the unit cube, are not valid for general distributions of X on the unit cube. Hence, some dependence of the error bounds in Theorem 4.3 on the distribution of X cannot be avoided.

The proof is prepared by several lemmas. These lemmas do not require the basis functions to be bounded, but they are merely assumed to be square-integrable with respect to the law of X. The first lemma reduces the problem to a problem of estimating the SVD truncation probability.

Lemma 4.5. *In the setting of Theorem 4.3,*

$$\begin{aligned} & E\left[\int_{\mathbb{R}^D} |y(x) - \hat{y}_L(x)|^2 \mu(dx)\right] \\ \leq \; & \left(1 + \frac{\lambda^*}{\lambda_*(1-\epsilon)}\right) \inf_{\alpha \in \mathbb{R}^K} \int_{\mathbb{R}^D} |y(x) - \alpha^\top \eta(x)|^2 \mu(dx) \\ & + P(\{s_K^2 < L(1-\epsilon)\lambda_*\}) \int_{\mathbb{R}^D} |y(x)|^2 \mu(dx) \end{aligned}$$

Proof. We decompose

$$\begin{aligned} & E\left[\int_{\mathbb{R}^D} |y(x) - \hat{y}_L(x)|^2 \mu(dx)\right] = E\left[\int_{\mathbb{R}^D} |y(x) - \hat{y}_L(x)|^2 \mu(dx) \mathbf{1}_{\{s_K^2 \geq L(1-\epsilon)\lambda_*\}}\right] \\ + \; & E\left[\int_{\mathbb{R}^D} |y(x) - \hat{y}_L(x)|^2 \mu(dx) \mathbf{1}_{\{s_K^2 < L(1-\epsilon)\lambda_*\}}\right] =: (I) + (II). \end{aligned}$$

As $\hat{y}_L = 0$ on $\{s_K^2 < L(1-\epsilon)\lambda_*\}$, we obtain

$$(II) = P(\{s_K^2 < L(1-\epsilon)\lambda_*\}) \int_{\mathbb{R}^D} |y(x)|^2 \mu(dx).$$

We now treat term (I). We are going to show that

$$\begin{aligned} & E\left[\int_{\mathbb{R}^D} |y(x) - \hat{y}_L(x)|^2 \mu(dx) \mathbf{1}_{\{s_K^2 \geq L(1-\epsilon)\lambda_*\}}\right] \\ \leq \; & \left(1 + \frac{\lambda^*}{\lambda_*(1-\epsilon)}\right) \inf_{\alpha \in \mathbb{R}^K} \int_{\mathbb{R}^D} |y(x) - \alpha^\top \eta(x)|^2 \mu(dx). \end{aligned} \tag{14}$$

To this end we denote by $\alpha_*^\top \eta(X)$ the orthogonal projection of Y on $\mathrm{span}(\eta_k(X);\, k = 1, \ldots, K)$. The full rank condition on R ensures that α_* is uniquely determined. Then, by orthogonality,

$$\begin{aligned} (I) \; &= \; E\left[\int_{\mathbb{R}^D} |(y(x) - \alpha_*^\top \eta(x)) + (\alpha_*^\top \eta(x) - \hat{y}_L(x))|^2 \mu(dx) \mathbf{1}_{\{s_K^2 \geq L(1-\epsilon)\lambda_*\}}\right] \\ &\leq \; E\left[\int_{\mathbb{R}^D} |(\alpha_L - \alpha_*)^\top \eta(x)|^2 \mu(dx) \mathbf{1}_{\{s_K^2 \geq L(1-\epsilon)\lambda_*\}}\right] \\ &\quad + \inf_{\alpha \in \mathbb{R}^K} \int_{\mathbb{R}^D} |y(x) - \alpha^\top \eta(x)|^2 \mu(dx) \end{aligned}$$

It remains to show that

$$\begin{aligned} & E\left[\int_{\mathbb{R}^D} |(\alpha_L - \alpha_*)^\top \eta(x)|^2 \mu(dx) \mathbf{1}_{\{s_K^2 \geq L(1-\epsilon)\lambda_*\}}\right] \\ \leq \; & \frac{\lambda^*}{\lambda_*(1-\epsilon)} \inf_{\alpha \in \mathbb{R}^K} \int_{\mathbb{R}^D} |y(x) - \alpha^\top \eta(x)|^2 \mu(dx) \end{aligned} \tag{15}$$

On the set $\{s_K^2 \geq L(1-\epsilon)\lambda_*\}$, the empirical regression matrix A has rank K. Hence, α_* is the unique solution z to the linear system

$$Az = (\alpha_*^\top \eta(X^{(l)}))_{l=1,\ldots,L}.$$

As $\mathcal{U}\mathcal{D}^\dagger\mathcal{V}$ is the pseudoinverse of A, we obtain,

$$\alpha_L - \alpha_* = \mathcal{U}\mathcal{D}^\dagger\mathcal{V}\,\xi_L, \quad \xi_L := (Y^{(l)} - \alpha_*^\top \eta(X^{(l)}))_{l=1,\ldots,L}.$$

Thus, on the set $\{s_K^2 \geq L(1-\epsilon)\lambda_*\}$,

$$\begin{aligned}
&\int_{\mathbb{R}^D} |(\alpha_L - \alpha_*)^\top \eta(x)|^2 \mu(dx) = (\alpha_L - \alpha_*)^\top \left(\int_{\mathbb{R}^D} \eta(x)\eta(x)^\top \mu(dx) \right) (\alpha_L - \alpha_*) \\
= \;& \xi_L^\top \mathcal{V}^\top (\mathcal{D}^\dagger)^\top \mathcal{U}^\top R \mathcal{U} \mathcal{D}^\dagger \mathcal{V} \xi_L \leq \lambda_{\max}(R)\, \xi_L^\top \mathcal{V}^\top (\mathcal{D}^\dagger)^\top \mathcal{D}^\dagger \mathcal{V} \xi_L \\
\leq \;& \lambda_{\max}(R)\lambda_{\max}((\mathcal{D}^\dagger)^\top \mathcal{D}^\dagger)\, \xi_L^\top \xi_L \leq \frac{\lambda^*}{s_K^2} \sum_{l=1}^{L} (Y^{(l)} - \alpha_*^\top \eta(X^{(l)}))^2, \qquad (16)
\end{aligned}$$

because $(\mathcal{D}^\dagger)^\top \mathcal{D}^\dagger = diag(s_1^{-2}, \ldots, s_K^{-2}, 0, \ldots, 0)$. Hence,

$$\begin{aligned}
& E\left[\int_{\mathbb{R}^D} |(\alpha_L - \alpha_*)^\top \eta(x)|^2 \mu(dx) \mathbf{1}_{\{s_K^2 \geq L(1-\epsilon)\lambda_*\}} \right] \\
\leq \;& \frac{\lambda^*}{\lambda_*(1-\epsilon)} E\left[\frac{1}{L} \sum_{l=1}^{L} (Y^{(l)} - \alpha_*^\top \eta(X^{(l)}))^2 \right] \\
= \;& \frac{\lambda^*}{\lambda_*(1-\epsilon)} \inf_{\alpha \in \mathbb{R}^K} \int_{\mathbb{R}^D} |y(x) - \alpha^\top \eta(x)|^2 \mu(dx).
\end{aligned}$$

Thus, (15) holds and the proof of the lemma is complete. □

Remark 4.6. One can straightforwardly modify the estimate in (16) in order to show that, thanks to the SVD truncation,

$$|\alpha_L|_2^2 \leq \frac{\|y\|_\infty}{\lambda_*(1-\epsilon)}.$$

Hence, the SVD truncation implies a control on the Euclidean norm of the coefficient vector α_L resulting from the empirical regression. A related approach can be found in [12], where the sample is rejected, if the Euclidean norm of the coefficient vector exceeds some given threshold. The main advantage of our approach is that we can apply concentration inequalities for random matrices to estimate the probability that the SVD truncation takes place, see the proof of Theorem 4.3 below. In contrast, the probability that a sample is rejected is only discussed heuristically in [12].

Lemma 4.7.

$$P(\{s_K^2 < L(1-\epsilon)\lambda_*\}) \leq P(\{\|\frac{1}{L}(A^\top A) - R\|_2 > \epsilon\lambda_*\}).$$

Proof. Note that s_K^2 is the smallest eigenvalue of $A^\top A$, and, then, $s_K^2/L = \lambda_{\min}(\frac{1}{L}(A^\top A))$. Hence,

$$\begin{aligned}
P(\{s_K^2 < L(1-\epsilon)\lambda_*\}) &\leq P(\{\lambda_{\min}(\frac{1}{L}(A^\top A)) - \lambda_{\min}(R) < -\epsilon\lambda_*\}) \\
&\leq P(\{|\lambda_{\min}(\frac{1}{L}(A^\top A)) - \lambda_{\min}(R)| > \epsilon\lambda_*\}.
\end{aligned}$$

Thus, the lemma follows from the fact that for positive semi-definite matrices Σ_1, Σ_2,

$$|\lambda_{\min}(\Sigma_1) - \lambda_{\min}(\Sigma_2)| \leq \|\Sigma_1 - \Sigma_2\|_2,$$

see e.g. [23], Corollary 7.3.8. □

The expression on the right-hand side in Lemma 4.7 can be estimated by a matrix Bernstein inequality. The following version is due to Tropp [40]:

Theorem 4.8 ([40], Theorem 1.6)**.** *Consider a finite sequence (Z_l) of independent, random matrices of size $K_1 \times K_2$. Assume that each random matrix satisfies*

$$E[Z_l] = 0, \quad \|Z_l\|_2 \leq B, \ P\text{-almost surely},$$

for some constant $B \geq 0$. Let

$$\sigma^2 := \max\left\{ \left\| \sum_l E[Z_l Z_l^\top] \right\|_2 , \ \left\| \sum_l E[Z_l^\top Z_l] \right\|_2 \right\}$$

Then, for every $t \geq 0$,

$$P(\{\| \sum_l Z_l\|_2 \geq t\}) \leq (K_1 + K_2) \exp\left\{ -\frac{t^2/2}{\sigma^2 + Bt/3} \right\}.$$

Proof of Theorem 4.3. In view of Lemmas 4.5 and 4.7, it suffices to show that

$$P(\{\|\frac{1}{L}(A^\top A) - R\|_2 > \epsilon\lambda_*\}) \leq 2K \exp\left\{ -\frac{3\epsilon^2 L}{6m\lambda^*/\lambda_*^2 + 2\epsilon(m/\lambda_* + \lambda^*/\lambda_*)} \right\}. \tag{17}$$

We consider the random $K \times K$ symmetric matrices

$$Z_l = \eta(X^{(l)})\eta(X^{(l)})^\top - R, \quad l = 1, \ldots, L.$$

Then,

$$P(\{\|\frac{1}{L}(A^\top A) - R\|_2 > \epsilon\lambda_*\}) = P(\{\| \sum_{l=1}^{L} Z_l\|_2 > L\epsilon\lambda_*\}).$$

As obviously $E[Z_l] = 0$, the matrix Bernstein inequality (Theorem 4.8) yields

$$P(\{\|\frac{1}{L}(A^\top A) - R\|_2 > \epsilon\lambda_*\}) \leq 2K \exp\left\{ -\frac{3\epsilon^2 L}{6\sigma^2/(\lambda_*^2 L) + 2B\epsilon/\lambda_*} \right\}, \tag{18}$$

where, by symmetry of Z_l,

$$\sigma^2 \ = \ \left\| \sum_{l=1}^{L} E[Z_l^2] \right\|_2 , \quad B = \operatorname*{esssup}_{\omega\in\Omega} \|Z_l(\omega)\|_2.$$

Recall that for a positive semi-definite matrix Σ,

$$\|\Sigma\|_2 = \max_{u\in\mathbb{R}^K;\, |u|_2=1} u^\top \Sigma u = \lambda_{\max}(\Sigma).$$

Hence, by the triangle inequality,

$$\begin{aligned} \|Z_l\|_2 \ &\leq \ \max_{u\in\mathbb{R}^K;\, |u|_2=1} (u^\top \eta(X^{(l)})\eta(X^{(l)})^\top u) + \lambda_{\max}(R) = |\eta(X^{(l)})|_2^2 + \lambda_{\max}(R) \\ &\leq \ m + \lambda^*. \end{aligned}$$

Thus,

$$B \leq (m + \lambda^*).$$

Now, note that the matrix $Z_l^2 = Z_l Z_l^\top$ is (ω-wise) positive semidefinite, and then so is the matrix $E[Z_l^2]$. As

$$E[Z_l^2] = E[\eta(X^{(l)})\eta(X^{(l)})^\top \eta(X^{(l)})\eta(X^{(l)})^\top] - RR \leq E[\eta(X^{(l)})\eta(X^{(l)})^\top \eta(X^{(l)})\eta(X^{(l)})^\top],$$

we obtain

$$\sigma^2 \ \leq \ L\lambda_{\max}(E[\eta(X^{(l)})\eta(X^{(l)})^\top \eta(X^{(l)})\eta(X^{(l)})^\top]) \leq Lm\lambda_{\max}(R) \leq Lm\lambda^*.$$

Combining the estimates for B and σ^2 with (18) we arrive at (17), and the proof is complete. □

4.2 Piecewise polynomial estimates

We now consider the case of piecewise polynomial estimates, when the random variable X is supported on the unit cube $[0,1]^D$. More precisely, we assume that X has a density with respect to the Lebesgue measure in $\mathbb{R}^D$ of the form

$$f(x)\mathbf{1}_{[0,1]^D}$$

such that f is continuous and strictly positive on the unit cube.

Our aim in this section is twofold. First, we show that our algorithm can achieve (up to a log-term) optimal rates of convergence for the interpolation problem with fixed random design. Second, we illustrate how to apply Theorem 4.3 in a simple setting before turning to the more involved proof of Theorem 3.5. Basically, all we need to establish as inputs for the theorem are suitable bounds on eigenvalues and on the supremum norm of the basis functions.

For fixed $N \in \mathbb{N}$, we apply a regular cubic partition

$$C_{\mathbf{i}} = \prod_{d=1}^{D} \left(\frac{i_d}{N}, \frac{i_d+1}{N} \right], \quad \mathbf{i} = (i_1, \dots i_D) \in \{0, \dots, N-1\}^D,$$

of the unit cube. We still denote by $\mathcal{L}_q : \mathbb{R} \to \mathbb{R}$ the Legendre polynomial of degree q, which is normalized such that $\mathcal{L}_q(1) = 1$, see Step 2 of Algorithm 3.3. Given a multi-index $\mathbf{j} \in \mathbb{N}_0^D$, we consider on each cube $C_{\mathbf{i}}$ the polynomials

$$p_{\mathbf{j},\mathbf{i}}(x) = N^{D/2} \prod_{d=1}^{D} \sqrt{2j_d+1}\, \mathcal{L}_{j_d}\left(2(Nx_d - i_d) - 1\right).$$

We now fix the maximal degree $Q \in \mathbb{N}_0$, and define the basis functions to be

$$\eta_{\mathbf{i},\mathbf{j}} = p_{\mathbf{i},\mathbf{j}} \mathbf{1}_{C_{\mathbf{i}}}, \quad \mathbf{i} \in \{0, \dots, N-1\}^D, \mathbf{j} \in \mathbb{N}_0^D, \sum_d j_d \le Q.$$

To simplify the notation we also write η_k, $k = 1, \dots K = N^D \binom{D+Q}{D}$, for any fixed ordering of these basis functions. We define

$$f_* := \inf_{x \in [0,1]^D} f(x), \quad f^* := \sup_{x \in [0,1]^D} f(x), \quad \varpi(h) := \sup_{x,z \in [0,1]^D,\, |x-z|_\infty \le h} |f(x) - f(z)|.$$

Hence, ϖ is the modulus of continuity of the density f with respect to the maximum norm. The continuity assumption on f ensures that $f_* > 0$, $f^* < \infty$, and $\varpi(h) \to 0$, as h tends to zero.

If we run Algorithm 4.1 in the above setting with $\tau = (1-\epsilon) f_*/2$ for an arbitrary $0 < \epsilon < 1$, we obtain the following convergence result.

Theorem 4.9. *Assume $Q = 0$ or $Q \in \mathbb{N}$ such that*

$$2\binom{D+Q}{D} e^{2Q} \le \frac{f_*}{\varpi(1/N)}. \tag{19}$$

Suppose $y : [0,1]^D \to \mathbb{R}$ is $(Q+\gamma, C)$-smooth, i.e. Q-times continuously differentiable and the partial derivatives of order Q are γ-Hölder-continuous for some $0 < \gamma \le 1$ with Hölder constant C (w.r.t the Euclidean norm). Let

$$L \ge \frac{N^D \binom{D+Q}{D} e^{2Q} (36 \frac{f^*}{f_*} + 4\epsilon) + 6\epsilon f^*}{3\epsilon^2 f_*} \log(c_0 N^{D+2(Q+\gamma)}) \tag{20}$$

for some constant $c_0 > 0$. Then,

$$\begin{aligned} & E\left[\int_{[0,1]^D} |y(x) - \hat{y}_L(x)|^2 \mu(dx)\right] \\ \leq \; & \frac{1}{N^{2(Q+\gamma)}} \left(\left(1 + \frac{3f^*}{f_*(1-\epsilon)}\right) \frac{D^{2Q+\gamma}}{(Q!)^2} C^2 + \frac{2\binom{D+Q}{D}}{c_0} \int_{[0,1]^D} |y(x)|^2 \mu(dx) \right). \end{aligned}$$

Remark 4.10. (i) Let the number of samples L tend to infinity, and choose N_L as the largest integer such that (20) is satisfied. Then, for every fixed Q, (19) is satisfied for sufficiently large L. Hence, the previous theorem states $\mathcal{L}^2$-convergence of the order $L^{-(Q+\gamma)/D}$ up to a logarithmic factor in the number of samples. This rate matches (up to the log-factor) the $\mathcal{L}^1$-minimax lower bound in [27] for the uniform distribution in the class of $(Q+\gamma, C)$-smooth functions and can, thus, be considered as optimal for the interpolation problem with fixed random design. Evidently, our algorithm can beat the Monte Carlo rate of 1/2, if the smoothness-to-dimension ratio is sufficiently large.

(ii) No attempt has been made to optimize the constants, but the focus of Theorem 4.9 is to derive the optimal rate of convergence. In particular the constant in front of the log-factor in (20) is very conservative. As this constant determines the convergence rate of the statistical error, the approximation error due to the basis choice actually is the leading error term in the setting of Theorem 4.9.

(iii) Recall that, in the setting above, $N^D\binom{D+Q}{D}$ equals the number of basis functions. So Theorem 4.9 reaches the optimal convergence rate, although the number of samples grows only proportionally to the number of basis functions (up to the log-factor).

(iv) In the case that $D = 1$ and y is twice continuously differentiable with bounded second derivative, we obtain (up to the log-factor) a rate of convergence of 2 in the number of samples L. Our analysis hence improves upon the results in Section 4 of [9], who achieve an O_P-rate arbitrarily close to 1 in the number of samples when applying a standard least-squares regression estimate (i.e., without the SVD truncation).

The proof of Theorem 4.9 relies on the following bounds of the supremum norm of the basis functions and of the eigenvalues.

Lemma 4.11. *Let $R = E[\eta(X)\eta^\top(X)]$ and $m = \sup_{x\in[0,1]^D} |\eta(x)|_2^2$. Assume $Q = 0$ or that $Q \in \mathbb{N}$ satisfies* (19)*. Then,*

$$\lambda_{\min}(R) \geq f_*/2, \quad \lambda_{\max}(R) \leq \frac{3}{2} f^*, \quad m \leq N^D e^{2Q} \binom{D+Q}{D}.$$

Proof. As the Legendre polynomials are bounded by 1 on $[-1, 1]$, we obtain for every $\mathbf{i} \in \{0, \dots, N-1\}^D$ and every $j \in \mathbb{N}_0^D$ such that $\sum_d j_d \leq Q$,

$$\sup_{x\in C_{\mathbf{i}}} |p_{\mathbf{i},\mathbf{j}}(x)| \leq N^{D/2} \prod_{d=1}^{D} \sqrt{2j_d + 1} \leq N^{D/2} e^{\sum_d j_d} \leq N^{D/2} e^Q. \tag{21}$$

As the cubes are disjoint and there are $\binom{D+Q}{D}$ basis functions per cube, the bound on m immediately follows.

If $Q = 0$, then R is a diagonal matrix with entries $N^D \mu(C_{\mathbf{i}})$, because $p_0 = 1$. Applying the bounds $N^{-D} f_* \leq \mu(C_{\mathbf{i}}) \leq N^{-D} f^*$, we obtain that $\lambda_{\min}(R) \geq f_*$ and $\lambda_{\max}(R) \leq f^*$. For $Q \in \mathbb{N}$, R has (after re-ordering of the basis functions, if necessary) block diagonal form, because the cubes are disjoint. Hence, it suffices to bound the eigenvalues separately

on each cube $C_{\mathbf{i}}$. In order to compute the entries of the block $R_{\mathbf{i}}$ of R, which stems from cube $C_{\mathbf{i}}$, we fix some point $x_{\mathbf{i}} \in C_{\mathbf{i}}$. Then,

$$\begin{aligned} & \int_{C_{\mathbf{i}}} p_{\mathbf{i},\mathbf{j}}(x) p_{\mathbf{i},\mathbf{j}'}(x) \mu(dx) \\ = \; & f(x_{\mathbf{i}}) \left(\int_{C_{\mathbf{i}}} p_{\mathbf{i},\mathbf{j}}(x) p_{\mathbf{i},\mathbf{j}'}(x) dx + \int_{C_{\mathbf{i}}} p_{\mathbf{i},\mathbf{j}}(x) p_{\mathbf{i},\mathbf{j}'}(x) \frac{f(x) - f(x_{\mathbf{i}})}{f(x_{\mathbf{i}})} dx \right) \\ =: \; & f(x_{\mathbf{i}}) \left((I) + (II) \right) \end{aligned}$$

Applying the orthonormality of the scaled Legendre polynomials $\sqrt{(2q+1)/2}\,\mathcal{L}_q$ on $[-1,1]$ with respect to the Lebesgue measure, we observe that

$$\begin{aligned} (I) \; &= \; \int_{[-1,1]^D} \prod_{d=1}^{D} \sqrt{\frac{2j_d+1}{2}}\, \mathcal{L}_{j_d}(x_d) \sqrt{\frac{2j'_d+1}{2}}\, \mathcal{L}_{j'_d}(x_d)\, dx \\ &= \; \prod_{d=1}^{D} \int_{[-1,1]} \sqrt{\frac{2j_d+1}{2}}\, \mathcal{L}_{j_d}(u) \sqrt{\frac{2j'_d+1}{2}}\, \mathcal{L}_{j'_d}(u)\, du = \mathbf{1}_{\mathbf{j}=\mathbf{j}'}. \end{aligned}$$

Moreover, by (21)

$$|(II)| \; \leq \; \frac{\varpi(1/N)}{f_*} e^{2Q}.$$

Hence, by Gershgorin's theorem (see [23], Theorem 6.1.1),

$$\lambda_{\max}(R_{\mathbf{i}}) \leq f(x_{\mathbf{i}}) \left(1 + \binom{D+Q}{D} \frac{\varpi(1/N)}{f_*} e^{2Q} \right) \leq \frac{3}{2} f^*,$$

if condition (19) is satisfied. In the same way we get the bound $\lambda_{\min}(R_{\mathbf{i}}) \geq f_*/2$. □

Proof of Theorem 4.9. In view of the previous lemma, we may apply Theorem 4.3 with

$$\lambda^* = \frac{3f^*}{2}, \quad \lambda_* = \frac{f_*}{2}, \quad m \leq N^D e^{2Q} \binom{D+Q}{D}.$$

Then,

$$-\frac{3\epsilon^2 L}{(36\frac{f^*}{f_*^2} + 4\frac{\epsilon}{f_*}) N^D e^{2Q} \binom{D+Q}{D} + 6\epsilon \frac{f^*}{f_*}} \leq -\log(c_0 N^{D+2(Q+\gamma)}).$$

Taking into account that $K = N^D \binom{D+Q}{D}$, Theorem 4.3 yields,

$$\begin{aligned} & E\left[\int_{[0,1]^D} |y(x) - \hat{y}_L(x)|^2 \mu(dx) \right] \\ \leq \; & \left(1 + \frac{3f^*}{f_*(1-\epsilon)} \right) \inf_{\alpha \in \mathbb{R}^K} \int_{[0,1]^D} |y(x) - \alpha^\top \eta(x)|^2 \mu(dx) \\ & + N^{-2(Q+\gamma)} \frac{2\binom{D+Q}{D}}{c_0} \int_{[0,1]^D} |y(x)|^2 \mu(dx). \end{aligned}$$

It remains to estimate the approximation error due to the basis choice. Note that

$$\inf_{\alpha \in \mathbb{R}^K} \int_{[0,1]^D} |y(x) - \alpha^\top \eta(x)|^2 \mu(dx) = \sum_{\mathbf{i}} \inf_{p_Q} \int_{C_{\mathbf{i}}} |y(x) - p_Q(x)|^2 \, \mu(dx),$$

where the infimum runs over the polynomials of degree at most Q. On each cube $C_{\mathbf{i}}$ fix some $x_{\mathbf{i}} \in C_{\mathbf{i}}$ and denote by p_Q the Taylor polynomial of degree Q around $x_{\mathbf{i}}$. Then, for $x \in C_{\mathbf{i}}$ there is a point ζ on the line connecting $x_{\mathbf{i}}$ and x such that

$$y(x) - p_Q(x) = \frac{1}{Q!} \sum_{(d_1,\dots,d_Q)\in\{1,\dots,D\}^Q} \left(\frac{\partial^Q y(\zeta)}{\partial_{x_{d_1},\dots,x_{d_Q}}} - \frac{\partial^Q y(x)}{\partial_{x_{d_1},\dots,x_{d_Q}}} \right) \prod_{j=1}^{Q} (x_{d_j} - x_{\mathbf{i},d_j})$$

Hence,

$$|y(x) - p_Q(x)| \le \frac{D^Q}{Q!} C |x-\zeta|_2^\gamma \, |x - x_{\mathbf{i}}|_\infty^Q \le \frac{D^Q}{Q!} C \sqrt{D}^\gamma N^{-(Q+\gamma)},$$

which implies

$$\inf_{\alpha\in\mathbb{R}^K} \int_{[0,1]^D} |y(x) - \alpha^\top \eta(x)|^2 \mu(dx) \le C^2 \frac{D^{2Q+\gamma}}{(Q!)^2} N^{-2(Q+\gamma)}.$$

□

5 Proof of Theorem 3.5

This section is devoted to the proof of Theorem 3.5. Thus, throughout this section all the notation introduced in Algorithm 3.3 and Theorem 3.5 is in force. We define

$$\begin{aligned} X_2^{(\Delta)} &= X_2 = X_1 + b(X_1)\Delta + \sigma(X_1)\sqrt{\Delta}\xi, \\ X_2^{(\Delta,r_2)} &= X_1 + b(X_1)\Delta + \sigma(X_1)\sqrt{\Delta}[\xi]_{r_2}, \\ X_2^{(\Delta,r_2,\mathbf{i})} &= U_{\mathbf{i}} + b(U_{\mathbf{i}})\Delta + \sigma(U_{\mathbf{i}})\sqrt{\Delta}[\xi]_{r_2}, \end{aligned}$$

where, for $\mathbf{i} \in I_\Delta$, $U_{\mathbf{i}}$ is uniformly distributed on the cube $\Gamma_{\mathbf{i}} = \prod_{d=1}^D (hi_d, h(i_d+1)]$. Here, $h = c_{\text{cube}}\Delta^{\gamma_{\text{cube}}}$.

The first lemma estimates the influence of truncating the Gaussian innovations at level r_2, where

$$r_2 = \sqrt{2\log(c_{2,\text{trunc}}\,\Delta^{-\gamma_{2,\text{trunc}}}\log(\Delta^{-1}))}.$$

Lemma 5.1 (Truncation of Gaussian innovations)**.** *Suppose $y \in \mathcal{C}_b^1(\mathbb{R}^D)$ and $\Delta < e^{-1}$, $r_2 \ge 1$. Then, there is a constant C_1 such that*

$$E[|\mathcal{H}_{\iota,\Delta}(\xi)y(X_2^{(\Delta)}) - \mathcal{H}_{\iota,\Delta}([\xi]_{r_2})y(X_2^{(\Delta,r_2)})|^2 | X_1] \le C_1 \Delta^{2\gamma_{2,\text{trunc}}/3 - |\iota|_1}$$

Proof. Let

$$g(x) = g(x; X_1) = \mathcal{H}_{\iota,\Delta}(x) y(X_1 + b(X_1)\Delta + \sigma(X_1)\sqrt{\Delta}x).$$

Then,

$$\begin{aligned} & |\mathcal{H}_{\iota,\Delta}(\xi)y(X_2^{(\Delta)}) - \mathcal{H}_{\iota,\Delta}([\xi]_{r_2})y(X_2^{(\Delta,r_2)})| = |g(\xi) - g([\xi]_{r_2})| \\ = & \left| \int_0^1 \langle \nabla g(\xi + u([\xi]_{r_2} - \xi)), [\xi]_{r_2} - \xi \rangle du \right| \\ \le & \int_0^1 |[\xi]_{r_2} - \xi|_2 \, |\nabla g(\xi + u([\xi]_{r_2} - \xi))|_2 du. \end{aligned} \tag{22}$$

We next estimate the gradient of g. By the product rule,

$$|\nabla g(x)|_2 \le \|y\|_\infty |\nabla \mathcal{H}_{\iota,\Delta}(x)|_2 + \sqrt{\Delta} C_{b,\sigma} \|\nabla y\|_\infty |\mathcal{H}_{\iota,\Delta}(x)|.$$

Denote by $C_{\mathcal{H},q}$ a positive constant such that

$$|\mathcal{H}_{\iota_0,1}(x)| \leq C_{\mathcal{H},q}(1+|x|_2^q)$$

for every multi-index ι_0 satisfying $|\iota_0|_1 \leq q$. As, for $\iota_j \geq 1$,

$$\frac{\partial}{\partial x_j}\mathcal{H}_{\iota,\Delta}(x) = \Delta^{-|\iota|_1/2}\iota_j \mathcal{H}_{\iota_j-1}(x_j)\prod_{d\neq j}\mathcal{H}_{\iota_d}(x_d) = \Delta^{-|\iota|_1/2}\iota_j \mathcal{H}_{\iota-e_j,1}(x),$$

(e_j denoting the jth unit vector in $\mathbb{R}^D$), we obtain for $\Delta \leq 1$

$$\begin{aligned} |\nabla g(x)|_2 &\leq (\|y\|_\infty\sqrt{D}|\iota|_\infty + C_{b,\sigma}\|\nabla y\|_\infty)C_{\mathcal{H},|\iota|_1}(1+|x|_2^{|\iota|_1})\Delta^{-|\iota|_1/2} \\ &=: \hat{C}_1(1+|x|_2^{|\iota|_1})\Delta^{-|\iota|_1/2}. \end{aligned}$$

Plugging this estimate into (22) and applying Jensen's inequality, Fubini's theorem, and Hölder's inequality yields

$$\begin{aligned} & E[|\mathcal{H}_{\iota,\Delta}(\xi)y(X_2^{(\Delta)}) - \mathcal{H}_{\iota,\Delta}([\xi]_{r_2})y(X_2^{(\Delta,r_2)})|^2|X_1] \\ \leq\ & E[|[\xi]_{r_2}-\xi|_2^3]^{2/3}\int_0^1 E[(\hat{C}_1(1+|\xi+u([\xi]_{r_2}-\xi)|_2^{|\iota|_1})\Delta^{-|\iota|_1/2})^6]^{1/3}du \\ \leq\ & \hat{C}_1^2\Delta^{-|\iota|_1}E[|[\xi]_{r_2}-\xi|_2^3]^{2/3}\max_{u\in[0,1]}E[(1+|\xi+u([\xi]_{r_2}-\xi)|_2^{|\iota|_1})^6]^{1/3}. \end{aligned}$$

We next estimate the last factor. By Jensen's inequality (twice),

$$E[(1+|\xi+u([\xi]_{r_2}-\xi)|_2^{|\iota|_1})^6] \leq 2^5(1+D^{3|\iota|_1}E[|\xi_1+u([\xi_1]_{r_2}-\xi_1)|^{6|\iota|_1}]$$

By convexity, the supremum over $u\in[0,1]$ of the righthand side is attained at $u=0$ or $u=1$. As the absolute moments of the truncated normal distribution are smaller than those of the normal distribution, it is, in fact, attained at $u=0$. Recalling that the qth moment of a normal distribution is given by $(q-1)!!$ for even q, we thus obtain

$$\begin{aligned} & E[|\mathcal{H}_{\iota,\Delta}(\xi)y(X_2^{(\Delta)}) - \mathcal{H}_{\iota,\Delta}([\xi]_{r_2})y(X_2^{(\Delta,r_2)})|^2|X_1] \\ \leq\ & \hat{C}_1^2 2^{5/3}(1+D^{3|\iota|_1+1}(6|\iota|_1-1)!!)^{1/3}\Delta^{-|\iota|_1}E[|[\xi]_{r_2}-\xi|_2^3]^{2/3}. \end{aligned}$$

Finally,

$$\begin{aligned} E[|[\xi]_{r_2}-\xi|_2^3] &= E[(\sum_{d=1}^D|\xi_d-[\xi_d]_{r_2}|^2])^{3/2}] \leq D^{3/2}E[|\xi_1-[\xi_1]_{r_2}|^3] \\ &\leq 2D^{3/2}\int_{r_2}^\infty u^3\varphi(u)du = D^{3/2}(2r_2^2\varphi(r_2)+4\varphi(r_2)) \leq 6D^{3/2}r_2^2\varphi(r_2). \end{aligned}$$

Combining the previous estimates, we arrive at

$$\begin{aligned} & E[|\mathcal{H}_{\iota,\Delta}(\xi)y(X_2^{(\Delta)}) - \mathcal{H}_{\iota,\Delta}([\xi]_{r_2})y(X_2^{(\Delta,r_2)})|^2|X_1] \\ \leq\ & 2^{5/3}6^{2/3}D(\|y\|_\infty\sqrt{D}|\iota|_\infty + C_{b,\sigma}\|\nabla y\|_\infty)^2 C_{\mathcal{H},|\iota|_1}^2(1+D^{3|\iota|_1+1}(6|\iota|_1-1)!!)^{1/3} \\ & \times\Delta^{-|\iota|_1}(r_2^2\varphi(r_2))^{2/3}. \end{aligned}$$

Taking the form of r_2 into account, we get

$$r_2^2\varphi(r_2) \leq \sqrt{\frac{2}{\pi}}(\log(c_{2,\text{trunc}})+\gamma_{2,\text{trunc}}+1)\frac{\Delta^{\gamma_{2,\text{trunc}}}}{c_{2,\text{trunc}}},$$

which finishes the proof. □

We next denote by Θ a (finite) family of random variables independent of $(X_1, \xi, U_i)_{i\in I_\Delta}$. We think of Θ as containing the simulated samples which are applied for estimating y, and assume that some measurable estimator $\hat{y}(x_1, x_2; \theta)$ of $y(x_2)$ is given. Recall that the truncation in space for the x_1-variable (i.e. the set Γ in Step 1 of Algorithm 3.3) depends on the constant

$$r_1 = \sqrt{C_{2,f}\chi^2_D(c_{1,\text{trunc}}\,\Delta^{\gamma_{1,\text{trunc}}})}.$$

The next lemma takes care of the change in the sampling distribution and removes the derivative weight.

Lemma 5.2 (Removal of the derivative weight and change of measure)**.** *Suppose $y \in \mathcal{C}_b^Q(\mathbb{R}^D)$ for $Q \in \mathbb{N}$ s.t. $Q \geq |\iota|_1$, and let $\Delta < \min(e^{-1}, c_{1,\text{trunc}}^{-1/\gamma_{1,\text{trunc}}})$, $r_2 \geq 1$, $\gamma_{2,\text{trunc}} \geq 3|\iota|_1/2$. Let $\Gamma := \Gamma^{(\Delta)} := \cup_{\mathbf{i}\in I_\Delta}\Gamma_{\mathbf{i}}$ and assume $\hat{y}(x_1, x_2; \theta) = 0$ for $x_1 \notin \Gamma$. Choose $\sigma(\Theta)$-measurable sets $\tilde{\Gamma}_{\mathbf{i}}$ $(\mathbf{i} \in I_\Delta)$ such that for every $\mathbf{i} \in I_\Delta$,*

$$\hat{y}(x_1, x_2; \Theta(\omega)) = 0 \text{ for every } x_1 \in \Gamma_{\mathbf{i}},\ x_2 \in \mathbb{R}^D,\ \omega \in \tilde{\Gamma}_{\mathbf{i}}^c.$$

(One may choose $\tilde{\Gamma}_{\mathbf{i}} = \Omega$ to ensure that this condition is always satisfied).
Then, there is a constant C_2 such that

$$\begin{aligned}
& E[|E[\mathcal{H}_{\iota,\Delta}([\xi]_{r_2})y(X_2^{(\Delta,r_2)})|X_1] - E[\mathcal{H}_{\iota,\Delta}([\xi]_{r_2})\hat{y}(X_1, X_2^{(\Delta,r_2)}, \Theta)|X_1, \Theta]|^2] \\
\leq\ & C_2\left(\Delta^{\gamma_{1,\text{trunc}}} + (1 + \log(\Delta^{-1})^{D/2}) \max_{\mathbf{i}\in I_\Delta} P(\tilde{\Gamma}_{\mathbf{i}}^c)\right. \\
& \left. + \Delta^{-|\iota|_1}(1 + \log(\Delta^{-1})^{D/2}) \max_{\mathbf{i}\in I_\Delta} E[\mathbf{1}_{\tilde{\Gamma}_i}|y(X_2^{(\Delta,r_2,\mathbf{i})}) - \hat{y}(U_{\mathbf{i}}, X_2^{(\Delta,r_2,\mathbf{i})}, \Theta)|^2]\right).
\end{aligned}$$

Proof. We decompose, using Hölder's inequality and exploiting that Θ is independent of (X_1, X_2, ξ),

$$\begin{aligned}
& |E[\mathcal{H}_{\iota,\Delta}([\xi]_{r_2})y(X_2^{(\Delta,r_2)})|X_1] - E[\mathcal{H}_{\iota,\Delta}([\xi]_{r_2})\hat{y}(X_1, X_2^{(\Delta,r_2)}, \Theta)|X_1, \Theta]|^2 \\
\leq\ & \left(\mathbf{1}_{\{X_1\notin\Gamma\}} + \sum_{\mathbf{i}\in I_\Delta} \mathbf{1}_{\tilde{\Gamma}_{\mathbf{i}}^c}\mathbf{1}_{\{X_1\in\Gamma_{\mathbf{i}}\}}\right) E[\mathcal{H}_{\iota,\Delta}([\xi]_{r_2})y(X_2^{(\Delta,r_2)})|X_1]^2 \\
& + \sum_{\mathbf{i}\in I_\Delta} \mathbf{1}_{\tilde{\Gamma}_{\mathbf{i}}}\mathbf{1}_{\{X_1\in\Gamma_{\mathbf{i}}\}} E[|\mathcal{H}_{\iota,\Delta}([\xi]_{r_2})|^2]E[|y(X_2^{(\Delta,r_2)}) - \hat{y}(X_1, X_2^{(\Delta,r_2)}, \Theta)|^2|X_1, \Theta] \\
=\ & (I) + (II).
\end{aligned}$$

Let

$$z(x) = E[\mathcal{H}_{\iota,\Delta}(\xi)y(X_2^{(\Delta)})|X_1 = x].$$

By (3), the regression function z is bounded. Applying the previous lemma, we get,

$$(I) \leq \left(\mathbf{1}_{\{X_1\notin\Gamma\}} + \sum_{\mathbf{i}\in I_\Delta} \mathbf{1}_{\tilde{\Gamma}_{\mathbf{i}}^c}\mathbf{1}_{\{X_1\in\Gamma\mathbf{i}\}}\right) 2(\|z\|_\infty^2 + C_1\Delta^{2\gamma_{2,\text{trunc}}/3-|\iota|_1}),$$

and, thus, by independence of X_1 and Θ,

$$E[(I)] \leq 2\left(P(\{X_1 \notin \Gamma\}) + P(\{X_1 \in \Gamma\}) \max_{i\in I_\Delta} P(\tilde{\Gamma}_i^c)\right)(\|z\|_\infty^2 + C_1). \tag{23}$$

However,

$$\begin{aligned} P(\{X_1 \notin \Gamma\}) &= \int_{\Gamma^c} f(x)dx \le \int_{\{|x|_2>r_1\}} \frac{C_{1,f}}{(2\pi C_{2,f})^{D/2}} \exp\left\{\frac{-|x|_2^2}{2C_{2,f}}\right\} dx \\ &= C_{1,f} \int_{\{|x|_2^2>r_1^2/C_{2,f}\}} \frac{1}{(2\pi)^{D/2}} \exp\left\{\frac{-|x|_2^2}{2}\right\} dx \\ &= C_{1,f} P(\{\Xi > r_1^2/C_{2,f}\}), \end{aligned}$$

where Ξ is χ^2-distributed with D degrees of freedom. Taking the particular form of r_1 into account, we obtain

$$P(\{X_1 \notin \Gamma\}) \le C_{1,f} c_{1,\mathrm{trunc}}\, \Delta^{\gamma_{1,\mathrm{trunc}}}. \tag{24}$$

Moreover,

$$P(\{X_1 \in \Gamma\}) \le \frac{C_{1,f}}{(2\pi C_{2,f})^{D/2}} \lambda^{\otimes D}(\Gamma). \tag{25}$$

Since $\Gamma \subset [-(r_1+h), r_1+h]^D$ we obtain,

$$\lambda^{\otimes D}(\Gamma) \le 2^D (r_1+h)^D.$$

As the $(1-\alpha)$-quantiles of a χ^2-distribution with D degrees of freedom satisfy

$$\chi^2_D(\alpha) \le D + 2\log(1/\alpha) + 2\sqrt{D\log(1/\alpha)} \le 2D + 3\log(1/\alpha),$$

see e.g. [24], we observe that

$$\chi^2_D(c_{1,\mathrm{trunc}}\, \Delta^{\gamma_{1,\mathrm{trunc}}}) \le 2D + 3\log(c_{1,\mathrm{trunc}}^{-1}) + 3\gamma_{1,\mathrm{trunc}} \log(\Delta^{-1}).$$

Hence,

$$\lambda^{\otimes D}(\Gamma) \le 2^D (c_{\mathrm{cube}} + \sqrt{2C_{2,f}D + 3C_{2,f}(\log(c_{1,\mathrm{trunc}}^{-1}) \vee 0)} + \sqrt{3C_{2,f}\gamma_{1,\mathrm{trunc}} \log(\Delta^{-1})})^D. \tag{26}$$

Combining (23)–(26), we arrive at

$$E[(I)] \le C_2 \left(\Delta^{\gamma_{1,\mathrm{trunc}}} + (1+\log(\Delta^{-1})^{D/2}) \max_{\mathbf{i}\in I_\Delta} P(\tilde{\Gamma}_i^c) \right).$$

We next turn to term (II). Applying the previous lemma once more with $y \equiv 1$, we get by the orthonormality of the Hermite polynomials $\mathcal{H}_{\iota,1}$ with respect to the distribution of ξ

$$E[|\mathcal{H}_{\iota,\Delta}([\xi]_{r_2})|^2] \le 2(E[|\mathcal{H}_{\iota,\Delta}(\xi)|^2] + C_1 \Delta^{2\gamma_{2,\mathrm{trunc}}/3-|\iota|_1}) \le 2(1+C_1)\Delta^{-|\iota|_1}.$$

Thus,

$$\begin{aligned} & E[(II)] \\ \le\ & 2(1+C_1)\Delta^{-|\iota|_1} \sum_{\mathbf{i}\in I_\Delta} E[\mathbf{1}_{\tilde{\Gamma}_\mathbf{i}} \int_{\mathbb{R}^D} \int_{\Gamma_\mathbf{i}} |y(x_1 + b(x_1)\Delta + \sigma(x_1)\sqrt{\Delta}[w]_{r_2}) \\ & \quad -\hat{y}(x_1, x_1 + b(x_1)\Delta + \sigma(x_1)\sqrt{\Delta}[w]_{r_2}; \Theta)|^2 f(x_1)\varphi^{\otimes D}(w)dx_1 dw] \\ \le\ & \Delta^{-|\iota|_1} \frac{2(1+C_1)C_{1,f}}{(2\pi C_{2,f})^{D/2}} \lambda^{\otimes D}(\Gamma) \max_{\mathbf{i}\in I_\Delta} E[\mathbf{1}_{\tilde{\Gamma}_\mathbf{i}} \int_{\mathbb{R}^D} \int_{\Gamma_\mathbf{i}} |y(x_1 + b(x_1)\Delta + \sigma(x_1)\sqrt{\Delta}[w]_{r_2}) \\ & \quad -\hat{y}(x_1, x_1 + b(x_1)\Delta + \sigma(x_1)\sqrt{\Delta}[w]_{r_2}; \Theta)|^2 \frac{1}{\lambda^{\otimes D}(\Gamma_\mathbf{i})} \varphi^{\otimes D}(w)dx_1 dw] \\ \le\ & \frac{2(1+C_1)C_{1,f}}{(2\pi C_{2,f})^{D/2}} \lambda^{\otimes D}(\Gamma) \Delta^{-|\iota|_1} \max_{\mathbf{i}\in I_\Delta} E[\mathbf{1}_{\tilde{\Gamma}_\mathbf{i}} |y(X_2^{(\Delta,r_2,\mathbf{i})}) - \hat{y}(U_\mathbf{i}, X_2^{(\Delta,r_2,\mathbf{i})}, \Theta)|^2]. \end{aligned}$$

In view of (26), the proof is complete. □

We next consider

$$\tilde{X}_\Delta = U_h + b(U_h)\Delta + \sigma(U_h)\sqrt{\Delta}[\xi]_{r_2},$$

where U_h is uniformly distributed on a cube $\tilde{\Gamma} = a_0 + (-h/2, h/2]^D$, $a_0 \in \mathbb{R}^D$, and ξ is a D-dimensional vector of independent standard normal variables. We suppose that $h = c_{\text{cube}}\Delta^{\gamma_{\text{cube}}}$ for some $0 < \gamma_{\text{cube}} < 1/2$ and $c_{\text{cube}} > 0$. Hence, $\tilde{X}_\Delta$ corresponds to $X_2^{(\Delta,r_2,i)}$ on a 'generic' cube of volume h^D.

Recall the construction of the basis functions in Algorithm 3.3 for fixed $Q \in \mathbb{N}_0$: Denoting by $\mathcal{L}_q$ the Legendre polynomial of degree $q \le Q$, and, given a multi-index $\mathbf{j} \in \mathbb{N}_0^D$ such that $|\mathbf{j}|_1 \le Q$, we consider the polynomials

$$p_{\mathbf{j}}(x) = \prod_{d=1}^{D} \sqrt{2j_d + 1}\, \mathcal{L}_{j_d}(x_d),$$

which are finally rescaled to

$$\eta_{\mathbf{j}}(x) = p_{\mathbf{j}}\left(\frac{x - a_0}{h/2}\right), \quad \mathbf{j} \in \mathbb{N}_0^D, \sum_d j_d \le Q.$$

We fix some ordering of these $\binom{D+Q}{D}$ basis functions and write η_k, $k = 1, \dots K := \binom{D+Q}{D}$,

$$R_\Delta = (E[\eta_k(\tilde{X}_\Delta)\eta_\kappa(\tilde{X}_\Delta)])_{k,\kappa=1,\dots,K}.$$

In order to apply Theorem 4.3, we need to estimate

$$\lambda_{\min,\Delta} := \lambda_{\min}(R_\Delta), \quad \lambda_{\max,\Delta} := \lambda_{\max}(R_\Delta), \quad m_\Delta := \sup_{x \in \text{supp}(\tilde{X}_\Delta))} \sum_{k=1}^{K} |\eta_k(x)|^2.$$

Lemma 5.3. *As $\Delta \to 0$, we have*

$$\lambda_{\min,\Delta} \to 1, \quad \lambda_{\max,\Delta} \to 1, \quad m_\Delta \to \sum_{\mathbf{j} \in \mathbb{N}_0^D; |\mathbf{j}|_1 \le Q} \prod_{d=1}^{D} (2j_d + 1).$$

As a preparation for the eigenvalue estimates we first prove the following lemma.

Lemma 5.4. *Suppose $\tilde{\eta} = (\tilde{\eta}_1, \dots, \tilde{\eta}_K)^\top$ is a system of orthonormal polynomials of degree at most $Q \in \mathbb{N}$ for the law of some $\mathbb{R}^D$-valued random variable $X^{(1)}$. Let $X^{(2)}$ be another $\mathbb{R}^D$-valued random variable. We consider the matrix*

$$R^{(2)} = E[\tilde{\eta}(X^{(2)})\tilde{\eta}^\top(X^{(2)})].$$

Then, there is a constant $c_{\tilde{\eta}}$, which only depends on the coefficients of the polynomials $\tilde{\eta}_k$ (and on Q, D) such that

$$\begin{aligned}
\lambda_{\min}(R^{(2)}) &\ge 1 - Kc_\eta \left(E[|X^{(1)} - X^{(2)}|_2^2]^{1/2} (1 + E[|X^{(1)}|_2^{4Q-2}]^{1/2}) + E[|X^{(1)} - X^{(2)}|_2^{2Q}] \right) \\
\lambda_{\max}(R^{(2)}) &\le 1 + Kc_\eta \left(E[|X^{(1)} - X^{(2)}|_2^2]^{1/2} (1 + E[|X^{(1)}|_2^{4Q-2}]^{1/2}) + E[|X^{(1)} - X^{(2)}|_2^{2Q}] \right)
\end{aligned}$$

Proof. Recall that $R^{(1)} = E[\tilde{\eta}(X^{(1)})\tilde{\eta}^\top(X^{(1)})]$ is the identity matrix. In view of Gershgorin's theorem ([23], Theorem 6.1.1), it hence suffices to show that for every $k, k' = 1, \dots K$,

$$\begin{aligned}
&|E[\tilde{\eta}_k(X^{(2)})\tilde{\eta}_{k'}(X^{(2)})] - E[\tilde{\eta}_k(X^{(1)})\tilde{\eta}_{k'}(X^{(1)})]| \\
\le\ & c_\eta \left(E[|X^{(1)} - X^{(2)}|_2^2]^{1/2} (1 + E[|X^{(1)}|_2^{4Q-2}]^{1/2}) + E[|X^{(1)} - X^{(2)}|_2^{2Q}] \right)
\end{aligned}$$

As $\nabla(\tilde\eta_k\tilde\eta_{k'})$ is a vector of polynomials of degree at most $2Q-1$ whose coefficients only depend on the coefficients of the polynomials in the system $\tilde\eta$, there is a constant $c_{\tilde\eta}$ (depending also on Q, D) such that

$$|\nabla(\tilde\eta_k\tilde\eta_{k'})(x)|_2 \le c_\eta(1+|x|_2^{2Q-1})$$

for every $k,k'=1,\dots K$. Hence, by Jensen's and Hölder's inequality,

$$\begin{aligned}
&|E[\tilde\eta_k(X^{(1)})\tilde\eta_{k'}(X^{(1)})] - E[\tilde\eta_k(X^{(1)})\tilde\eta_{k'}(X^{(1)})]|\\
=\;& |\int_0^1 E[(X^{(2)}-X^{(1)})^\top\nabla(\eta_k\eta_{k'})(X^{(1)}+u(X^{(2)}-X^{(1)}))]du|\\
\le\;& c_\eta E[|(X^{(2)}-X^{(1)})|_2\,(1+(|X^{(1)}|_2+|X^{(2)}-X^{(1)}|_2)^{2Q-1})]\\
\le\;& c_\eta 2^{2Q-2}E[|(X^{(2)}-X^{(1)})|_2^2]^{1/2}(1+E[|X^{(1)}|_2^{4Q-2}]^{1/2}) + c_\eta 2^{2Q-2}E[|(X^{(2)}-X^{(1)})|_2^{2Q}].
\end{aligned}$$

□

Proof of Lemma 5.3. For the eigenvalue estimates we apply the previous lemma with

$$X^{(1)} = \frac{U_h - a_0}{h/2}, \quad X^{(2)} = \frac{\tilde X_\Delta - a_0}{h/2}$$

to the system of multivariate Legendre polynomials $(p_{\mathbf j})_{|\mathbf j|_1\le Q}$. Since $X^{(1)}$ is uniformly distributed on the cube $(-1,1]^D$, we obtain,

$$E[p_{\mathbf j}(X^{(1)})p_{\mathbf j'}(X^{(1)})] = \prod_{d=1}^D\int_{-1}^1\sqrt{\frac{2j_d+1}{2}}\sqrt{\frac{2j_d'+1}{2}}\mathcal L_{j_d'}(x_d)\mathcal L_{j_d}(x_d)dx_d = \mathbf 1_{j=j'}.$$

Hence, $(p_{\mathbf j})_{|\mathbf j|_1\le Q}$ are orthonormal with respect to the law of $X^{(1)}$. Moreover,

$$\begin{aligned}
|X^{(2)}-X^{(1)}|_2 &= 2|b(U_h)c_{\text{cube}}^{-1}\Delta^{1-\gamma_{\text{cube}}} + c_{\text{cube}}^{-1}\sigma(U_h)\Delta^{1/2-\gamma_{\text{cube}}}[\xi]_{r_2}|_2\\
&\le \frac{2c_{b,\sigma}}{c_{\text{cube}}}(\Delta^{1-\gamma_{\text{cube}}} + \Delta^{1/2-\gamma_{\text{cube}}}|\xi|_2).
\end{aligned}$$

As $\gamma_{\text{cube}} = \frac{\rho+|\iota|_1}{2(Q+1)} < \frac12$, we observe that

$$E[|X^{(2)}-X^{(1)}|_2^2] + E[|X^{(2)}-X^{(1)}|_2^{2Q}] \to 0$$

for $\Delta\to 0$, and, consequently, by the previous lemma, $\lambda_{\min,\Delta}$, $\lambda_{\max,\Delta}\to 1$.

We now turn to the limiting behavior of m_Δ. Note first that

$$m_\Delta := \sup_{x\in\text{supp}(X^{(2)}))}\sum_{|\mathbf j|_1\le Q}|p_{\mathbf j}(x)|^2,$$

and recall that the squared univariate Legendre polynomials achieve their maximum on $[-u,u]$ at u, if $u\ge 1$. As $X^{(1)}$ is uniformly distributed on $(-1,1]^D$ and, because, similarly, to the above estimate

$$|X^{(2)}-X^{(1)}|_\infty \le \frac{2c_{b,\sigma}}{c_{\text{cube}}}(\Delta^{1-\gamma_{\text{cube}}} + \sqrt{D}\Delta^{1/2-\gamma_{\text{cube}}}r_2),$$

we observe that

$$m_\Delta \to \sup_{x\in[-1,1]^D}\sum_{|\mathbf j|_1\le Q}|p_{\mathbf j}(x)|^2 = \sum_{\mathbf j\in\mathbb N_0^D;|\mathbf j|_1\le Q}\prod_{d=1}^D(2j_d+1).$$

□

In view of Theorem 4.3, the following lemma is the key to control the statistical error in Algorithm 3.3.

Lemma 5.5. *Let*

$$L = L_\Delta = \lceil \gamma_{\text{paths}}\, c_{1,\text{paths}} \log(c_{2,\text{paths}}\, \Delta^{-1}) \rceil$$

for constants

$$\gamma_{\text{paths}} > 0, \quad c_{2,\text{paths}} > 0, \quad c_{1,\text{paths}} > c^*_{\text{paths}}(Q,D) := \frac{2}{3} + \frac{8}{3} \sum_{\mathbf{j}\in\mathbb{N}_0^D;|\mathbf{j}|_1\le Q} \prod_{d=1}^{D} (2j_d+1).$$

Let

$$\tau \in \left(0, \quad 1 - \left(\frac{c^*_{\text{paths}}(Q,D)}{c_{1,\text{paths}}}\right)^{1/2}\right).$$

Then, there is a constant $\Delta_0 > 0$ such that, for every $\Delta \le \Delta_0$, $\epsilon_\Delta := 1 - \tau/\lambda_{\min,\Delta} \in (0, 1-\tau/2]$ and

$$2K \exp\left\{-\frac{3\epsilon_\Delta^2 L}{6m_\Delta \lambda_{\max,\Delta}/\lambda^2_{\min,\Delta} + 2\epsilon_\Delta(m_\Delta/\lambda_{\min,\Delta} + \lambda_{\max,\Delta}/\lambda_{\min,\Delta})}\right\} \le \frac{2\binom{D+Q}{D}}{c^{\gamma_{\text{paths}}}_{2,\text{paths}}} \Delta^{\gamma_{\text{paths}}}.$$

Proof. We denote by $\beta < 1$ the unique constant such that

$$\tau = \beta\left(1 - \left(\frac{c^*_{\text{paths}}(Q,D)}{c_{1,\text{paths}}}\right)^{\beta/2}\right).$$

Let

$$m_\infty := \sum_{\mathbf{j}\in\mathbb{N}_0^D;|\mathbf{j}|_1\le Q} \prod_{d=1}^{D} (2j_d+1).$$

By Lemma 5.3, there is a $\Delta_0 > 0$ such that for every $\Delta \le \Delta_0$

$$\begin{aligned}
\lambda_{\max,\Delta}/\lambda_{\min,\Delta} &\le \left(\frac{c^*_{\text{paths}}(Q,D)}{c_{1,\text{paths}}}\right)^{-(1-\beta)}, \\
m_\Delta \lambda_{\max,\Delta}/\lambda^2_{\min,\Delta} &\le m_\infty \left(\frac{c^*_{\text{paths}}(Q,D)}{c_{1,\text{paths}}}\right)^{-(1-\beta)}, \\
m_\Delta/\lambda_{\min,\Delta} &\le m_\infty \left(\frac{c^*_{\text{paths}}(Q,D)}{c_{1,\text{paths}}}\right)^{-(1-\beta)}, \\
\lambda_{\min,\Delta} &\in [\beta, 2].
\end{aligned}$$

The last property ensures that

$$\epsilon_\Delta \ge \left(\frac{c^*_{\text{paths}}(Q,D)}{c_{1,\text{paths}}}\right)^{\beta/2} > 0,$$

and $\epsilon_\Delta \le 1 - \tau/2$. Hence,

$$3\epsilon_\Delta^2 c_{1,\text{paths}} \ge \left(\frac{c^*_{\text{paths}}(Q,D)}{c_{1,\text{paths}}}\right)^{\beta-1} (2 + 8m_\infty).$$

Inserting the above inequalities and the definition of L yields

$$\frac{3\epsilon_\Delta^2 L}{6m_\Delta \lambda_{\max,\Delta}/\lambda_{\min,\Delta}^2 + 2\epsilon_\Delta(m_\Delta/\lambda_{\min,\Delta} + \lambda_{\max,\Delta}/\lambda_{\min,\Delta})} \geq \gamma_{\text{paths}} \log(c_{2,\text{paths}}\, \Delta^{-1}).$$

Recalling that $K = \binom{D+Q}{D}$, the assertion follows. □

Proof of Theorem 3.5. Write

$$\hat{y}(x_1, x_2, \Theta) := \sum_{\mathbf{i}\in I_\Delta} \mathbf{1}_{\Gamma_i}(x_1) \sum_{k=1}^{K} \alpha_{L,\mathbf{i},k} \eta_{i,k}(x_2)$$

where the coefficients $\alpha_{L,\mathbf{i},k}$ are computed via Algorithm 3.3 and depend on the simulated samples

$$\Theta = (U_{\mathbf{i},l}, \xi_{\mathbf{i},l})_{l=1,\ldots,L;\, \mathbf{i}\in I_\Delta}.$$

Let $(X_1, \xi, U_\mathbf{i})_{\mathbf{i}\in I_\Delta}$ be an independent family, which is also independent of Θ, and such that X_1 is μ_1-distributed, ξ is a vector of length D of independent standard normals, and $U_\mathbf{i}$ is uniformly distributed on $\Gamma_\mathbf{i}$. Then,

$$\hat{z}(x) \quad := \quad \sum_{\mathbf{i}\in I_\Delta} \mathbf{1}_{\Gamma_\mathbf{i}}(x) \sum_{k=1}^{K} \alpha_{L,\mathbf{i},k} E[\eta_{\mathbf{i},k}(x + b(x)\Delta + \sigma(x)\sqrt{\Delta}[\xi]_{r_2}) \mathcal{H}_{\iota,\Delta}([\xi]_{r_2})]$$

satisfies

$$\hat{z}(X_1) = E[\mathcal{H}_{\iota,\Delta}([\xi]_{r_2}) \hat{y}(X_1, X_2^{(\Delta,r_2)}, \Theta) | X_1, \Theta].$$

Let

$$\tilde{\Gamma}_\mathbf{i} := \{s_{\mathbf{i},K}^2 \geq \tau L\},$$

where $s_{\mathbf{i},K}$ is the smallest singular value of the random regression matrix $A_\mathbf{i}$. By Lemmas 5.1 and 5.2, we obtain for sufficiently small Δ,

$$\begin{aligned}
& E\left[\int_{\mathbb{R}^D} |E[\mathcal{H}_{\iota,\Delta}(\xi) y(X_2)|X_1 = x] - \hat{z}(x)|^2 \mu_1(dx)\right] \\
= \;& E\left[|E[\mathcal{H}_{\iota,\Delta}(\xi) y(X_2)|X_1] - \hat{z}(X_1)|^2\right] \\
\leq \;& 2E\left[|E[\mathcal{H}_{\iota,\Delta}([\xi]_{r_2}) y(X_2^{(\Delta,r_2)})|X_1] - \hat{z}(X_1)|^2\right] + 2C_1 \Delta^{2\gamma_{2,\text{trunc}}/3 - |\iota|_1} \\
\leq \;& 2C_2 \log(\Delta^{-1})^{D/2} \Delta^{-|\iota|_1} \max_{\mathbf{i}\in I_\Delta} E[\mathbf{1}_{\{s_{\mathbf{i},K}^2 \geq \tau L\}} |y(X_2^{(\Delta,r_2,i)}) - \hat{y}(U_i, X_2^{(\Delta,r_2,\mathbf{i})}, \Theta)|^2] \\
& + 2C_2 \Delta^{\gamma_{1,\text{trunc}}} + 2C_1 \Delta^{2\gamma_{2,\text{trunc}}/3 - |\iota|_1} + 2C_2 \log(\Delta^{-1})^{D/2} \max_{\mathbf{i}\in I_\Delta} P(\{s_{\mathbf{i},K}^2 \geq \tau L\})
\end{aligned}$$

By the choice of $\gamma_{1,\text{trunc}} = \rho$ and $\gamma_{2,\text{trunc}} = 1.5(\rho + |\iota|_1)$, the last two terms are of order Δ^ρ. It thus remains to show that there is a constant $C_3 \geq 0$ and a $\Delta_0 > 0$ such that for every $\Delta \leq \Delta_0$ and $\mathbf{i} \in I_\Delta$

$$\begin{aligned}
P(\{s_{\mathbf{i},K}^2 \geq \tau L\}) \quad &\leq \quad C_3 \Delta^\rho, \\
E[\mathbf{1}_{\{s_{\mathbf{i},K}^2 \geq \tau L\}} |y(X_2^{(\Delta,r_2,\mathbf{i})}) - \hat{y}(U_\mathbf{i}, X_2^{(\Delta,r_2,\mathbf{i})}, \Theta)|^2] \quad &\leq \quad C_3 \Delta^{\rho + |\iota|_1}. \qquad (27)
\end{aligned}$$

Note that

$$\hat{y}(U_\mathbf{i}, X_2^{(\Delta,r_2,\mathbf{i})}, \Theta) = \sum_{k=1}^{K} \alpha_{L,\mathbf{i},k}\, \eta_{\mathbf{i},k}(X_2^{(\Delta,r_2,\mathbf{i})}),$$

where the coefficients $(\alpha_{L,\mathbf{i},k})_{k=1,\dots,K}$ are computed on the $\mathbf{i}$th cube via Algorithm 4.1. Let $\epsilon_\Delta := 1 - \tau/\lambda_{\min,\Delta}$ Applying Lemma 4.7 and (17) in conjunction with Lemma 5.5, there is a $\Delta_0 > 0$ such that for $\Delta \le \Delta_0$

$$P(\{s^2_{\mathbf{i},K} \ge \tau L\}) \quad \le \quad \frac{2\|y\|^2_\infty \binom{D+Q}{D}}{c^\rho_{2,\text{paths}}} \Delta^\rho$$

Hence, the first term in (27) is of the required order as well.

Concerning the second term in (27), we obtain in view of (14),

$$\begin{aligned} & E[\mathbf{1}_{\{s^2_{\mathbf{i},K} \ge \tau L\}} |y(X_2^{(\Delta,r_2,\mathbf{i})}) - \hat{y}(U_\mathbf{i}, X_2^{(\Delta,r_2,\mathbf{i})}, \Theta)|^2] \\ \le \quad & \left(1 + \frac{\lambda_{\max,\Delta}}{\lambda_{\min,\Delta}(1-\epsilon_\Delta)}\right) \inf_{\alpha \in \mathbb{R}^K} E[|y(X_2^{(\Delta,r_2,\mathbf{i})}) - \sum_{k=1}^K \alpha_k \eta_{\mathbf{i},k}(X_2^{(\Delta,r_2,\mathbf{i})})|^2] \end{aligned}$$

By Lemma 5.3, we may assume without loss of generality that $\lambda_{\max,\Delta} \le 2$ for $\Delta \le \Delta_0$ (by decreasing Δ_0, if necessary). Hence, for $\Delta \le \Delta_0$,

$$\frac{\lambda_{\max,\Delta}}{\lambda_{\min,\Delta}(1-\epsilon_\Delta)} \le 2\tau^{-1}.$$

Concerning the approximation error due to the choice of the basis functions, we perform a Taylor expansion of order Q around the center $a_\mathbf{i}$ of the $\mathbf{i}$th cube. Then, as in the proof of Theorem 4.9, there is a polynomial $\mathcal{P}$ of degree at most Q such that

$$|y(X_2^{(\Delta,r_2,\mathbf{i})}) - \mathcal{P}(X_2^{(\Delta,r_2,\mathbf{i})})| \le C_4 |X_2^{(\Delta,r_2,\mathbf{i})} - a_\mathbf{i}|_2^{Q+1}$$

for some constant $C_4 \ge 0$, which depends only on D, Q, and the C_b^{Q+1}-norm of y. As this polynomial can be expressed as linear combination of the rescaled Legendre polynomials $\eta_{\mathbf{i},k}$, $k = 1, \dots, K$, we observe that

$$\inf_{\alpha \in \mathbb{R}^K} E[|y(X_2^{(\Delta,r_2,\mathbf{i})}) - \sum_{k=1}^K \alpha_k \, \eta_{\mathbf{i},k}(X_2^{(\Delta,r_2,\mathbf{i})})|^2] \le C_4^2 E[|X_2^{(\Delta,r_2,\mathbf{i})} - a_\mathbf{i}|_2^{2(Q+1)}]$$

Since,

$$|X_2^{(\Delta,r_2,\mathbf{i})} - a_i|_2 \le \sqrt{D}\frac{h}{2} + C_{b,\sigma}(\Delta + \sqrt{\Delta}[\xi]_{r_2}),$$

the term

$$E[|X_2^{(\Delta,r_2,\mathbf{i})} - a_\mathbf{i}|_2^{2(|\iota|_1+\rho+1)}]$$

is of the order $\Delta^{(2Q+2)\gamma_{cube}} = \Delta^{\rho+|\iota|_1}$. Thus, the proof of (27) is complete. □

Acknowledgements

We thank Christian Gärtner for his contributions to unpublished predecessors of this project.

References

[1] Alanko, S., Avellaneda, M. (2013) Reducing variance in the numerical solution of BSDEs. *C. R. Math. Acad. Sci. Paris* 351, 135–138.

[2] Andersen, L., Broadie, M. (2004) Primal-dual simulation algorithm for pricing multidimensional American options. *Manage. Sci.* 50, 1222–1234.

[3] Avellaneda, M., Levy, A., Parás, A. (1995) Pricing and hedging derivative securities in markets with uncertain volatilities. *Appl. Math. Finance*, 2, 73–88.

[4] Balata, A., Palczewski, J. (2018) Regress-later Monte Carlo for optimal control of Markov processes. arXiv:1712.09705.

[5] Bauer, B., Devroye, L., Kohler, M., Krzyzak, A., Walk, H. (2017) Nonparametric estimation of a function from noiseless observations at random points. *J. Multivariate Anal.* 160, 93–104.

[6] Bender, C., Gärtner, C., Schweizer, N. (2017) Iterative improvement of lower and upper bounds for backward SDEs. *SIAM J. Sci. Comput.* 39, B442–B466.

[7] Bender, C., Gärtner, C., Schweizer, N. (2018) Pathwise dynamic programming. *Math. Oper. Res.*, 43, 965–995.

[8] Bender, C., Steiner, J. (2012) Least squares Monte Carlo for BSDEs. In: Carmona, R. et al. (eds.) *Numerical Methods in Finance*, Springer, Berlin, pp. 257–289.

[9] Beutner, E., Schweizer, J., Pelsser, A. (2013) Fast convergence of regress-later estimates in least squares Monte Carlo. arXiv:1309.5274.

[10] Bouchard, B, Touzi, N. (2004) Discrete-time approximation and Monte Carlo simulation of backward stochastic differential equations. *Stochastic Process. Appl.* 111, 175–206.

[11] Brandt, M. W., Goyal, A., Santa-Clara, P., Stroud, J. R. (2005) A simulation approach to dynamic portfolio choice with an application to learning about return predictability. *Rev. Financ. Stud.* 18, 831–873.

[12] Chau, K. W. , Oosterlee C. W. (2018) Stochastic grid bundling method for backward stochastic differential equations. arXiv:1801.05180.

[13] Denault, M., Simonato, J. G., Stentoft, L. (2013) A simulation-and-regression approach for stochastic dynamic programs with endogenous state variables. *Comput. Oper. Res.* 40, 2760–2769.

[14] Fahim, A., Touzi, N., Warin, X. (2011) A probabilistic numerical method for fully nonlinear parabolic PDEs. *Ann. Appl. Probab.* 21, 1322–1364.

[15] Glasserman, P. (2004) *Monte Carlo Methods in Financial Engineering.* Springer, New York.

[16] Glasserman, P., Yu, B. (2004) Simulation for American options: Regression now or regression later? In: Niederreiter, H. (ed.) *Monte Carlo and Quasi-Monte Carlo Methods 2002*, Springer, Berlin, pp. 213–226.

[17] Gobet, E., Labart, C. (2007) Error expansion for the discretization of backward stochastic differential equations. *Stochastic Process. Appl.* 117, 803–829.

[18] Gobet, E., López-Salas, J. G., Turkedjiev, P., Vázquez, C. (2016) Stratified regression Monte-Carlo scheme for semilinear PDEs and BSDEs with large scale parallelization on GPUs. *SIAM J. Sci. Comput.* 38, C652–C677.

[19] Golub, G. H., van Loan, C. H. (1996) *Matrix Computations.* 3rd edition, Johns Hopkins University Press, Baltimore.

[20] Guyon, J., Henry-Labordère, P. (2011) The uncertain volatility model: a Monte Carlo approach. *J. Comput. Finance* 14, 37–71.

[21] Györfi, L., Kohler, M., Krzyzak, A., Walk, H. (2002) *A Distribution-Free Theory of Nonparametric Regression.* Springer, New York.

[22] Hansen, P. C. (1987) The truncated SVD as a method for regularization. *BIT* 27, 534–553.

[23] Horn, R. A., Johnson, C. R. (1990) *Matrix Analysis.* Cambridge University Press, Cambridge.

[24] Inglot, T. (2010) Inequalities for quantiles of the chi-square distribution. *Probab. Math. Statist.* 30, 339–351.

[25] Jain, S., Oosterlee, C. W. (2015) The stochastic grid bundling method: efficient pricing of Bermudan options and their Greeks. *Appl. Math. Comput.* 269, 412–431.

[26] Kharroubi, I., Langrené, N., Pham, H. (2014) A numerical algorithm for fully nonlinear HJB equations: An approach by control randomization. *Monte Carlo Methods Appl.* 20, 145–165.

[27] Kohler, M., Krzyzak, A. (2013) Optimal global rates of convergence for interpolation problems with random design. *Stat. Probab. Lett.* 83, 1871–1879.

[28] Lemaire, V., Menozzi, S. (2010) On some non asymptotic bounds for the Euler scheme. *Electr. J. Probab.* 15, 1645–1681.

[29] Lemor, J.-P., Gobet, E., Warin, X. (2006) Rate of convergence of an empirical regression method for solving generalized backward stochastic differential equations, *Bernoulli* 12, 889–916.

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

[31] Lyons, T. J. (1995) Uncertain volatility and the risk-free synthesis of derivatives. *Appl. Math. Finance* 2, 117–133.

[32] Maxwell, M. S., Restrepo, M., Henderson, S. G., Topaloglu, H. (2010) Approximate dynamic programming for ambulance redeployment. *INFORMS J. Comput.* 22, 266–281.

[33] Nadarajah, S., Margot, F., Secomandi, N. (2017) Comparison of least squares Monte Carlo methods with applications to energy real options. *Eur. J. Oper. Res.* 256, 196–204.

[34] Pelsser, A., Schweizer, J. (2016) The difference between LSMC and replicating portfolio in insurance liability modeling. *Eur. Actuar. J.* 6, 441–494.

[35] Powell, W. B. (2007) *Approximate Dynamic Programming: Solving the curses of dimensionality.* John Wiley & Sons, Hoboken.

[36] Rust J. (2018) Dynamic Programming. In: Macmillan Publishers (eds.) *The New Palgrave Dictionary of Economics*, Palgrave Macmillan, London.

[37] Schwartz, E. S. (2004) Patents and R&D as real options. *Econ. Notes* 33, 23–54.

[38] Tan, X. (2014) Discrete-time probabilistic approximation of path-dependent stochastic control problems, *Ann. Appl. Probab.* 24, 1803–1834.

[39] Tsitsiklis, J. N., Van Roy, B. (2001) Regression methods for pricing complex American-style options. *IEEE Trans. Neural Networks* 12, 694–703.

[40] Tropp, J. A. (2012) User-friendly tail bounds for sums of random matrices. *Found. Comput. Math.* 12, 389–434.

[41] Vanden J. M. (2006) Exact superreplication strategies for a class of derivative assets. *Appl. Math. Finance* 13, 61–87.

[42] Zhang, J. (2004) A numerical scheme for BSDEs. *Ann. Appl. Probab.* 14, 459–488.