The Deep Multi-FBSDE Method: A Robust Deep Learning Method for Coupled FBSDEs

Journal of Scientific Computing (2026) 106:77
https://doi.org/10.1007/s10915-026-03202-1

The Deep Multi-FBSDE Method: A Robust Deep Learning Method for Coupled FBSDEs

Kristoffer Andersson[1] · **Adam Andersson**[2] · **Cornelis W. Oosterlee**[3]

Received: 31 May 2025 / Revised: 2 January 2026 / Accepted: 18 January 2026

Abstract
We introduce the deep multi-FBSDE method for robust approximation of coupled forward-backward stochastic differential equations (FBSDEs), focusing on cases where the deep BSDE method of Han, Jentzen, and E (2018) fails to converge. To overcome the convergence issues, we consider a family of FBSDEs that are equivalent to the original problem in the sense that they satisfy the same associated partial differential equation (PDE) and initial value. Our algorithm proceeds in two phases: first, we approximate the initial condition jointly for a small number of FBSDEs from the FBSDE family, and second, we approximate the original FBSDE using the initial condition approximated in the first phase. Numerical experiments show that our method converges even when the standard deep FBSDE method does not.

Keywords FBSDEs · Parabolic PDEs · Numerical methods · Neural networks

Mathematics Subject Classification 60H10 · 65C30 · 60H35 · 35R60 · 93E20 · 68T07

1 Introduction

Solving forward-backward stochastic differential equations (FBSDEs) using deep learning has emerged as a powerful technique to overcome the curse of dimensionality inherent in classical numerical methods. Han, Jentzen, and E's seminal work [21] introduced the *deep BSDE method*, employing neural networks to approximate solutions of high-dimensional parabolic partial differential equations (PDEs) by reformulating them as FBSDEs. This breakthrough demonstrated remarkable performance on problems previously considered intractable, showcasing the ability of neural networks to efficiently handle complex nonlinearities at scales unattainable by traditional approaches. Following this success, several deep learning-based strategies have emerged, notably [3, 5–7, 12, 18, 23, 29, 30, 35, 37], with convergence analyses provided in, e.g., [2, 10, 16, 20, 22, 27, 28, 31, 36]. Concurrently, a separate branch known as backward-type methods, closer in spirit to classical dynamic programming algorithms, has

✉ Kristoffer Andersson
kristofferherbert.andersson@univr.it

[1] Department of Mathematics, University of Utrecht, Utrecht, The Netherlands

[2] Chalmers University of Technology, Gothenburg, Sweden

[3] University of Utrecht, Utrecht, The Netherlands

Published online: 09 February 2026

developed, see e.g., [14, 17, 19, 26, 26, 32, 33]. For a comprehensive overview of machine learning algorithms for PDE approximation, we refer to [8].

Despite these advancements, forward-type deep BSDE methods, including the original method in [21], still face significant convergence challenges when addressing certain classes of FBSDEs, such as those encountered in stochastic control problems [1] and beyond. Specifically, these forward-type approaches often fail to converge or become trapped in suboptimal solutions due to ill-conditioned optimization landscapes [1, 11, 26]. Although backward-type methods may not suffer from precisely the same convergence issues, they encounter different challenges, especially implementation challenges when dealing with coupled FBSDEs (see [9, 15, 25] for further discussions of implementation challenges for backward type methods when dealing with coupled FBSDEs) and accumulated errors in general (see e.g., [33]). For FBSDEs related to stochastic control problems through dynamic programming, a robust deep FBSDE method was proposed in our paper [1]. The scope of the present paper is the introduction of another robust modification of the deep FBSDE method, the *deep multi-FBSDE method*. It is applicable to a broader class of FBSDEs and their PDEs, not strictly defined due to the experimental nature of our paper, but not restricted to only control problems. By robustness, we mean that the method can be optimized for the equation of interest, and not the other way around, which is sometimes seen in the deep learning-based scientific computing literature. Of course, this is an inprecise definition and no method allows convergence for any equation. Still, our method has been working on the vast majority of equations we have tested and we have deliberately chosen challenging equations. We refer to the original deep BSDE method, applied directly to coupled FBSDE without modification, as *the deep FBSDE method* [1, 29].

Our proposed approach alleviates optimization difficulties by utilizing a family of FBSDEs that share the same initial condition as well as the same functional forms as the original FBSDE. In other words, all FBSDEs in this family are solved by the same PDE. We propose a two-phase optimization procedure. In the first phase, we approximate the joint initial conditions for all FBSDEs. In the second phase, we apply a reduced version of the deep FBSDE method, where we optimize only over the control processes while using the initial condition obtained from the first phase. This leads to a numerical scheme which we experimentally show has consistent and accurate convergence, even for strongly coupled, nonlinear FBSDEs. Our numerical experiments demonstrate stable and accurate solutions for problems for which the classical deep FBSDE methods fail.

In summary, the primary contributions of this paper are as follows.

- A robust deep FBSDE method that we empirically demonstrate solves the convergence issues faced by existing deep learning-based methods for strongly coupled FBSDEs.
- A general purpose training strategy that is demonstrated numerically to be applicable to general FBSDE systems, without restrictive assumptions such as the existence of an equivalent stochastic control problem, as in [1].
- Comprehensive numerical demonstrations illustrating the effectiveness and robustness of our method, especially in challenging scenarios involving stochastic control and nonlinear PDEs, thus underscoring its broad applicability and efficiency.

The rest of this paper is organized as follows: Section 2 presents the FBSDE problem and its solution by the non-linear Feynman–Kac formula through the PDE. Section 3 introduces our proposed robust deep learning method, detailing its formulation and implementation. Numerical results demonstrating its advantages are presented in Section 4, and we end in Section 5 with conclusions and future directions.

2 Forward Backward Stochastic Differential Equations

Throughout this paper, we let $T \in (0, \infty), d, k \in \mathbb{N}, x_0 \in \mathbb{R}^d, (W_t)_{t\in[0,T]}$ be a k-dimensional standard Brownian motion on a filtered probability space $(\Omega, \mathcal{F}, (\mathcal{F}_t)_{t\in[0,T]}, \mathbb{P})$ and the problem coefficients be denoted by $b\colon [0,T]\times\mathbb{R}^d\times\mathbb{R}\times\mathbb{R}^d \to \mathbb{R}^d, \sigma\colon [0,T]\times\mathbb{R}^d \to \mathbb{R}^{d\times k}$, $g\colon \mathbb{R}^d \to \mathbb{R}$ and $f\colon [0,T]\times\mathbb{R}^d\times\mathbb{R}\times\mathbb{R}^d \to \mathbb{R}$. The general form of the FBSDEs considered is given by the coupled system of equations

$$\begin{aligned} X_t =& x_0 + \int_0^t b(s, X_s, Y_s, Z_s)\mathrm{d}s + \int_0^t \sigma(s, X_s)\mathrm{d}W_s, \\ Y_t =& g(X_T) + \int_t^T f(s, X_s, Y_s, Z_s)\mathrm{d}s - \int_t^T \langle Z_s, \sigma(s, X_s)\mathrm{d}W_s\rangle, \quad t \in [0,T]. \end{aligned} \tag{1}$$

The associate PDE is then a semilinear parabolic PDE on the form

$$\begin{aligned} \frac{\partial v}{\partial t}(t,x) + \mathcal{L}v(t,x) + f\big(t, x, v(t,x), \mathrm{D}_x v(t,x)\big) =& 0, \quad (t,x) \in [0,T)\times\mathbb{R}^d, \\ v(T,x) =& g(x), \quad x \in \mathbb{R}^d. \end{aligned} \tag{2}$$

Here $\mathcal{L}$ is the infinitesimal generator, acting on a twice continuously differentiable functions ρ by:

$$\begin{aligned} \mathcal{L}\rho(t,x) =& b\big(t, x, \rho(t,x), \mathrm{D}_x\rho(t,x)\big)^\top \mathrm{D}_x\rho(t,x) + \frac{1}{2}\mathrm{Tr}\big(a(t,x)\mathrm{D}_x^2\rho(t,x)\big) \\ =& \sum_{i=1}^d b_i\big(t, x, \rho(t,x), \mathrm{D}_x\rho(t,x)\big)\partial_i\rho(t,x) + \frac{1}{2}\sum_{i,j=1}^d a_{ij}(t,x)\partial_{i,j}\rho(t,x), \end{aligned}$$

where $a = \sigma\sigma^\top$ and $\mathrm{Tr}(\cdot)$ denotes the trace of a matrix. The connection between (1) and (2) is established through the nonlinear Feynman–Kac formula, see *e.g.,* [38], which states that

$$Y_t = v(t, X_t), \text{ and } Z_t = \mathrm{D}_x v(t, X_t).$$

Note that in the above, we have implicitly assumed that v is differentiable with respect to the spatial variable. Although this holds true when (2) possesses a classical solution, we can often only expect a solution in the viscosity sense. In this situation, derivatives of v with respect to the space variable should be interpreted as set-valued sub-derivatives. Nevertheless, from our perspective, which is to find approximations of (1) and (2), we can overlook this obstacle and view Z as defined above.

3 The Deep FBSDE and Deep Multi-FBSDE Methods

In this section, we first introduce the classical deep FBSDE method and briefly discuss its convergence problems for coupled FBSDEs. Most importantly, we introduce our novel modification of the scheme. We use a Linear Quadratic (LQ) problem to demonstrate both the convergence problems of the deep FBSDE method and the convergence of our method. The details of the LQ-problem can be found in Section 4.1.2 together complementary experimental results.

3.1 The Deep FBSDE Method

The deep FBSDE method relies on a reformulation of the FBSDE (1) into two forward SDEs, one with an a priori unknown initial value. It relies moreover on the Markov property of the FBSDE, which guarantees that $Z_t = \zeta^*(t, X_t)$, for some function $\zeta^*\colon [0,T] \times \mathbb{R}^d \to \mathbb{R}^d$, that we refer to as a Markov map. Optimization is done with respect to such functions and the initial values y_0 with the objective to satisfy the terminal condition. More precisely, the FBSDE (1) is reformulated into the following variational problem

$$\begin{cases} \underset{y_0,\zeta}{\text{minimize}}\ \mathbb{E}|Y_T^{y_0,\zeta} - g(X_T^{y_0,\zeta})|^2, \quad \text{where} \\ X_t^{y_0,\zeta} = x_0 + \displaystyle\int_0^t b(s, X_s^{y_0,\zeta}, Y_s^{y_0,\zeta}, Z_s^{y_0,\zeta})\mathrm{d}s + \int_0^t \sigma(s, X_s^{y_0,\zeta})\mathrm{d}W_s, \\ Y_t^{y_0,\zeta} = y_0 - \displaystyle\int_0^t f(s, X_s^{y_0,\zeta}, Y_s^{y_0,\zeta}, Z_s^{y_0,\zeta})\mathrm{d}s + \int_0^t \langle Z_s^{y_0,\zeta}, \sigma(s, X_s^{y_0,\zeta})\mathrm{d}W_s\rangle, \\ Z_t^{y_0,\zeta} = \zeta(t, X_t^{y_0,\zeta}), \quad t \in [0,T]. \end{cases} \tag{3}$$

Here y_0 and ζ are sought in appropriate spaces. It is clear that setting $y_0 = Y_0$ and $\zeta = \zeta^*$ yields a minimizer of the objective function which attains zero and hence, solves (3). Moreover, assuming well-posedness and sufficient regularity of (1), existence and uniqueness of the minimizer are guaranteed. The connection between (1) and (2) yields $y_0^* = Y_0 = v(0, x_0)$ and $\zeta^* = Z = \mathrm{D}_x v$.

We obtain a semi-discretized version of (3) by approximating the Itô integrals using the Euler–Maruyama scheme. We assume an equidistant temporal grid with $N \in \mathbb{N}$ time steps given by $t_0 < t_1 < \ldots < t_N$ with $t_0 = 0, t_N = T$ and step size $h = t_{n+1} - t_n$, and denote the *i.i.d.* Wiener increments by $\Delta W_n = W_{t_{n+1}} - W_{t_n} \sim \mathcal{N}(\mathbf{0}_k, I_k h)$. The semi-discrete version of (3) is given by:

$$\begin{cases} \underset{y_0,\zeta=(\zeta_0,\ldots,\zeta_{N-1})}{\text{minimize}}\ \mathbb{E}\left|Y_N^{h,y_0,\zeta} - g(X_N^{h,y_0,\zeta})\right|^2, \quad \text{where} \\ X_{n+1}^{h,y_0,\zeta} = X_n^{h,y_0,\zeta} + b\left(t_n, X_n^{h,y_0,\zeta}, Y_n^{h,y_0,\zeta}, Z_n^{h,y_0,\zeta}\right)h + \sigma(t_n, X_n^{h,y_0,\zeta})\Delta W_n, \\ Y_{n+1}^{h,y_0,\zeta} = Y_n^{h,y_0,\zeta} - f\left(t_n, X_n^{h,y_0,\zeta}, Y_n^{h,y_0,\zeta}, Z_n^{h,y_0,\zeta}\right)h \\ +\left\langle Z_n^{h,y_0,\zeta}, \sigma\left(t_n, X_n^{h,y_0,\zeta}\right)\Delta W_n\right\rangle, \\ Z_n^{h,y_0,\zeta} = \zeta_n(X_n^{h,y_0,\zeta}),\ Y_0^{h,y_0,\zeta} = y_0,\ X_0^{h,y_0,\zeta} = x_0, \quad n \in \{0,1,\ldots,N-1\}. \end{cases} \tag{4}$$

In the deep FBSDE method, we search for $y_0 \in \mathbb{R}$, a simple trainable parameter, and for $\zeta = (\zeta_n)_{n=0}^{N-1}$ in a parametric space defined by the architecture of a specified neural network.

In [22], an a posteriori error analysis of (4) is provided under certain, rather restrictive assumptions, and in a setting where the FBSDE is coupled only through the Y-component. This was later extended to allow coupling also in Z in [34, 36].The full simulation error for the FBSDE can be bounded (up to a constant) by the time step h and the a posteriori term given by the mean-squared error of the discrete terminal condition. Concretely, there exists a constant C, independent of h, such that for sufficiently small h

$$\sup_{t\in[0,T]} \mathbb{E}|X_t - \hat{X}_t^{h,y_0,\zeta}|^2 + \sup_{t\in[0,T]} \mathbb{E}|Y_t - \hat{Y}_t^{h,y_0,\zeta}|^2 + \mathbb{E}\left[\int_0^T |Z_t - \hat{Z}_t^{h,y_0,\zeta}|^2 \mathrm{d}t\right] \leq C\big(h + \mathbb{E}|\hat{Y}_N^{h,y_0,\zeta} - g(X_N^{h,y_0,\zeta})|^2\big), \tag{5}$$

where for $n \in \{0, 1, \ldots, N-1\}$, and $t \in [t_n, t_{n+1})$ we have defined

$$\hat{X}_t^{h,y_0,\zeta} = X_n^{h,y_0,\zeta}, \quad \hat{Y}_t^{h,y_0,\zeta} = Y_n^{h,y_0,\zeta}, \quad \hat{Z}_t^{h,y_0,\zeta} = Z_n^{h,y_0,\zeta}.$$

Under conditions that guarantee the same or similar a posteriori bound, such as those in [22], the function $\mathbb{E}|\hat{Y}_N^{h,y_0,\zeta} - g(X_N^{h,y_0,\zeta})|^2$ is a highly suitable loss function and guarantees an approximation with an error bounded by the loss after training and the time step. When the assumptions are not met, the situation is less fortunate. Counterexamples of FBSDEs from stochastic control are presented in [1] for which it is possible to make the right-hand side of (5) arbitrarily small, but with the left-hand side remaining large and discrete solutions converging to severely wrong processes. In this paper, we provide four more counterexamples. Thus, while the a posteriori bound in [22] is a powerful tool under its specific conditions, extrapolating it beyond these conditions is a questionable practice.

For instructive purposes and since we address the convergence problem documented in [1], with a novel method, we briefly repeat the problem with a numerical example. In order to demonstrate that the optimization landscape for the deep FBSDE method is problematic for strongly coupled FBSDEs, we use the mean-squared error (MSE)

$$\mathrm{MSE}(y_0) \coloneqq \inf_{\zeta=(\zeta_0,\ldots,\zeta_{N-1})} \mathbb{E}\Big[\big|Y_N^{h,y_0,\zeta} - g(X_N^{h,y_0,\zeta})\big|^2\Big]. \tag{6}$$

This is the MSE with y_0 fixed and only ζ being optimized. If joint optimization of y_0 and ζ, as in the deep FBSDE method, is supposed to yield a good approximation of the FBSDE, then by necessity the minimum of $y_0 \mapsto \mathrm{MSE}(y_0)$ must be close to the true Y_0 of the FBSDE. For an LQ problem, detailed in Section 4.1.2 below, Figure 1 shows that the optimization landscape $y_0 \mapsto \mathrm{MSE}(y_0)$ instead has a minimum y_0^* very far from the true Y_0. Figure 2 shows the exact Y and the output of the deep FBSDE method, pathwise and in mean. We note that although the initial condition is significantly incorrect, the terminal condition is well approximated, i.e., $g(X_N^{h,y_0,\zeta}) - Y_N^{h,y_0,\zeta}$ is small both on the pathwise level and in the mean-squared sense. It should be noted that the numerical algorithm has converged, as the loss function is no longer decreasing and y_0 remains constant (not shown here but Figure 6 qualitatively shows the same behavior for a different problem). Moreover, the algorithm is consistent, with repeated runs yielding similar values for the loss function and y_0. The reader might wonder if a smaller time step h in the approximation would improve approximation of Y_0. Figure 5 indicates a negative answer to this valid thought by showing non-convergence of our implementation of the deep FBSDE method (after thorough hyperparameter optimization) (the case $K = 1$ in that figure).

The behavior depicted in Figures 1–2 is not confined to a narrow class of specific problems or special cases. For example, it is not limited to strongly coupled FBSDEs, and one can easily construct low-dimensional (e.g., $d = 2$) examples of decoupled BSDEs with smooth solutions to the associated PDE, wherein the deep FBSDE method fails. In particular, quadratic BSDEs appear susceptible to this phenomenon. A more detailed, although mostly qualitative and empirical, discussion of the reasons for this lack of convergence is provided in [1].

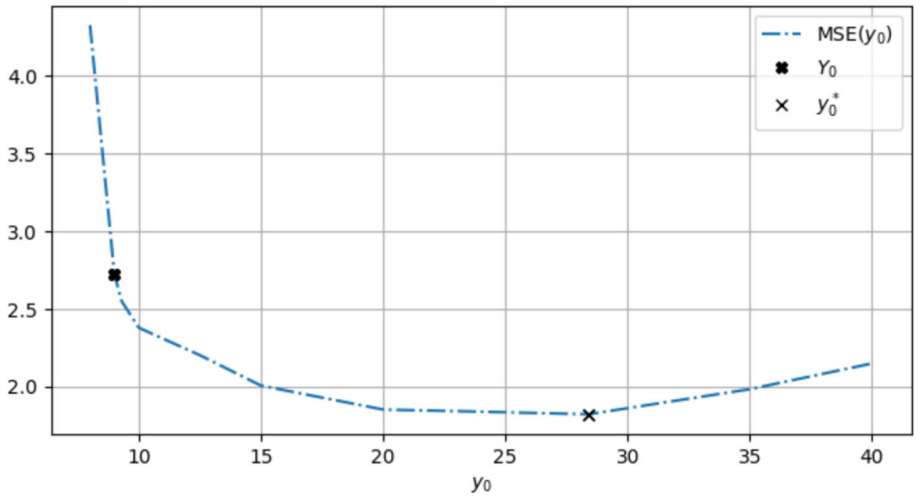

Fig. 1 MSE plotted for different values of y_0 for the LQ-problem of Section 4.1.2

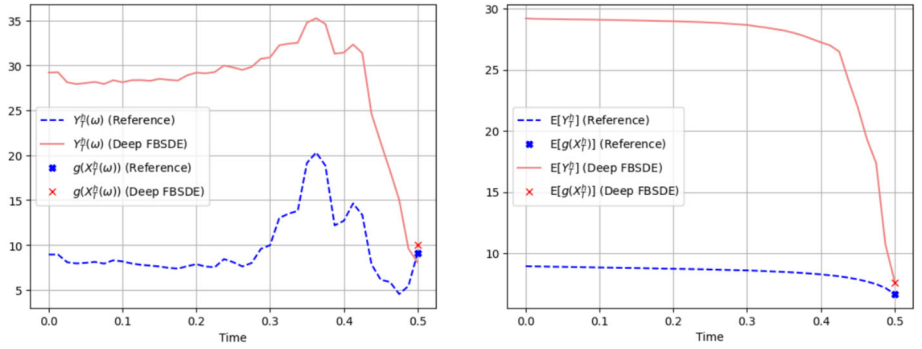

Fig. 2 The deep FBSDE method failing to approximate the true reference solution. **Left:** One representative path. **Right:** A sample mean of size 2^{12}

3.2 The Deep Multi-FBSDE Method

Our deep multi-FBSDE method is based on a slightly modified variational formulation. To arrive at it, we notice that for sufficiently regular $\psi : [0, T] \times \mathbb{R}^d \times \mathbb{R} \times \mathbb{R}^d \to \mathbb{R}^d$, it holds by the Itô formula, for $Y_t^\psi = v(t, X_t^\psi)$ and $Z_t^\psi = \mathrm{D}_x v(t, X_t^\psi)$ that

$$\begin{aligned} X_t^\psi &= x_0 + \int_0^t \big(b(s, X_s^\psi, Y_s^\psi, Z_s^\psi) - \psi(s, X_s^\psi, Y_s^\psi, Z_s^\psi)\big)\mathrm{d}s + \int_0^t \sigma(s, X_s^\psi)\mathrm{d}W_s, \\ Y_t^\psi &= g(X_T^\psi) + \int_t^T \big(f(s, X_s^\psi, Y_s^\psi, Z_s^\psi) + \big\langle \psi(s, X_s^\psi, Y_s^\psi, Z_s^\psi), Z_s^\psi \big\rangle\big)\mathrm{d}s \\ &\quad - \int_t^T \big\langle Z_s^\psi, \sigma(s, X_s^\psi)\mathrm{d}W_s \big\rangle, \quad t \in [0, T]. \end{aligned} \tag{7}$$

This defines a family of FBSDEs, indexed by ψ, which are all equivalent to the PDE (2). In particular, for all ψ we have $Y_0^\psi = v(0, X_0^\psi) = v(0, x_0)$ and thus the initial value is shared

for all ψ, which we utilize. Introducing the shorthand notation

$$b^{\psi}(t,x,y,z) := b(t,x,y,z) - \psi(t,x,y,z)$$
$$f^{\psi}(t,x,y,z) := f(t,x,y,z) + \langle z, \psi(t,x,y,z)\rangle,$$

we have for any ψ the variational formulation

$$\begin{cases} \underset{y_0,\zeta}{\text{minimize}}\ \mathbb{E}\big|Y_T^{y_0,\zeta,\psi} - g(X_T^{y_0,\zeta,\psi})\big|^2, \quad \text{where} \\ X_t^{y_0,\zeta,\psi} = x_0 + \displaystyle\int_0^t b^{\psi}(s, X_s^{y_0,\zeta,\psi}, Y_s^{y_0,\zeta,\psi}, Z_s^{y_0,\zeta,\psi})\mathrm{d}s + \int_0^t \sigma(s, X_s^{y_0,\zeta,\psi})\mathrm{d}W_s, \\ Y_t^{y_0,\zeta,\psi} = y_0 - \displaystyle\int_0^t f^{\psi}(s, X_s^{y_0,\zeta,\psi}, Y_s^{y_0,\zeta,\psi}, Z_s^{y_0,\zeta,\psi})\mathrm{d}s \\ \qquad\qquad + \displaystyle\int_0^t \big\langle Z_s^{y_0,\zeta,\psi}, \sigma(s, X_s^{y_0,\zeta,\psi})\mathrm{d}W_s\big\rangle, \\ Z_t^{y_0,\zeta,\psi} = \zeta(t, X_t^{y_0,\zeta,\psi}), \quad t \in [0,T]. \end{cases} \tag{8}$$

Since the optimal y_0^* and ζ^* are optimal for all ψ, we trivially have, for $K \in \mathbb{N}$, $\psi_1, \ldots, \psi_K : [0,T] \times \mathbb{R}^d \times \mathbb{R} \times \mathbb{R}^d \to \mathbb{R}^d$, the equivalent variational formulation,

$$\begin{cases} \underset{y_0,\zeta}{\text{minimize}}\ \displaystyle\sum_{i=1}^{K} \mathbb{E}\big|Y_T^{y_0,\zeta,\psi_i} - g(X_T^{y_0,\zeta,\psi_i})\big|^2, \quad \text{where for } \psi \in \{\psi_1, \ldots, \psi_K\} \\ X_t^{y_0,\zeta,\psi} = x_0 + \displaystyle\int_0^t b^{\psi}(s, X_s^{y_0,\zeta,\psi}, Y_s^{y_0,\zeta,\psi}, Z_s^{y_0,\zeta,\psi})\mathrm{d}s + \int_0^t \sigma(s, X_s^{y_0,\zeta,\psi})\mathrm{d}W_s, \\ Y_t^{y_0,\zeta,\psi} = y_0 - \displaystyle\int_0^t f^{\psi}(s, X_s^{y_0,\zeta,\psi}, Y_s^{y_0,\zeta,\psi}, Z_s^{y_0,\zeta,\psi})\mathrm{d}s \\ \qquad\qquad + \displaystyle\int_0^t \big\langle Z_s^{y_0,\zeta,\psi}, \sigma(s, X_s^{y_0,\zeta,\psi})\mathrm{d}W_s\big\rangle, \\ Z_t^{y_0,\zeta,\psi} = \zeta(t, X_t^{y_0,\zeta,\psi}), \quad t \in [0,T]. \end{cases} \tag{9}$$

The deep multi-FBSDE is divided into two phases. Phase I is based on approximating (9) for suitable K and $\psi_1, \ldots, \psi_K$, with time discretization, neural network approximation of ζ and stochastic gradient descent. In the second phase, a better approximation of ζ is obtained by solving the classical deep FBSDE method with the exception that only ζ is optimized and y_0 is taken from Phase I. For easy and low-dimensional problems, Phase I gives an accurate approximation of ζ and Phase II is not needed. For more challenging and high-dimensional problems, we have observed that Phase II improves the approximation accuracy. We believe that this has to do with the fact that in Phase I the Markov map ζ is optimized for good fit along typical trajectories of K different SDEs, i.e., in a larger domain than in Phase II, where it is only trained along typical trajectories of the original forward SDE, i.e., for a more limited task.

We next state the scheme and similar to the deep FBSDE method let ζ represent $\zeta_0, \ldots, \zeta_{N-1}$.

Phase I - Approximation of Y_0:

$$\begin{cases} \underset{y_0,\zeta_0,\ldots,\zeta_{N-1}}{\text{minimize}} \sum_{i=1}^{K} \mathbb{E}\left|Y_N^{h,y_0,\zeta,\psi_i} - g(X_N^{h,y_0,\zeta,\psi_i})\right|^2, \quad \text{where for } \psi \in \{\psi_1,\ldots,\psi_K\} \\ X_{n+1}^{h,y_0,\zeta,\psi} = X_n^{h,y_0,\zeta,\psi} + b^{\psi}\big(t_n, X_n^{h,y_0,\zeta,\psi}, Y_n^{h,y_0,\zeta,\psi}, Z_n^{h,y_0,\zeta,\psi}\big)h \\ \qquad\qquad +\sigma\left(t_n, X_n^{h,y_0,\zeta,\psi}\right)\Delta W_n, \\ Y_{n+1}^{h,y_0,\zeta,\psi} = Y_n^{h,y_0,\zeta,\psi} - f^{\psi}\big(t_n, X_n^{h,y_0,\zeta,\psi}, Y_n^{h,y_0,\zeta,\psi}, Z_n^{h,y_0,\zeta,\psi}\big)h \\ \qquad\qquad +\big\langle Z_n^{h,y_0,\zeta,\psi}, \sigma\left(t_n, X_n^{h,y_0,\zeta,\psi}\right)\Delta W_n\big\rangle, \\ Z_n^{h,y_0,\zeta,\psi} = \zeta_n(X_n^{h,y_0,\zeta,\psi}), \ X_0^{h,y_0,\zeta,\psi} = x_0, \ Y_0^{h,y_0,\zeta,\psi} = y_0, \\ \text{for } n \in \{0,1,\ldots,N-1\}. \end{cases} \tag{10}$$

Phase II - Approximation of the full FBSDE with known initial value:

$$\begin{cases} \underset{\zeta_0,\ldots,\zeta_{N-1}}{\text{minimize}}\ \mathbb{E}\left|Y_N^{h,\zeta} - g(X_N^{h,\zeta})\right|^2, \quad \text{where} \\ X_{n+1}^{h,\zeta} = X_n^{h,\zeta} + b\big(t_n, X_n^{h,\zeta}, Y_n^{h,\zeta}, Z_n^{h,\zeta}\big)h + \sigma\left(t_n, X_n^{h,\zeta}\right)\Delta W_n, \\ Y_{n+1}^{h,\zeta} = Y_n^{h,\zeta} - f\big(t_n, X_n^{h,\zeta}, Y_n^{h,\zeta}, Z_n^{h,\zeta}\big)h + \big\langle Z_n^{h,\zeta}, \sigma\left(t_n, X_n^{h,\zeta}\right)\Delta W_n\big\rangle, \\ Z_n^{h,\zeta} = \zeta_n\big(X_n^{h,\zeta}\big), \ Y_0^{h,\zeta} = y_0^*, \quad \text{for } n \in \{0,1,\ldots,N-1\}. \end{cases} \tag{11}$$

We demonstrate the new numerical scheme on the LQG-problem, detailed in Section 4.1.2, for which the deep FBSDE method was shown to fail in Section 3.1. In Phase I, we take $K = 1, 2, 3, 4$, with the choices $\psi_1 = 0$, $(\psi_1, \psi_2) = (0, b_1)$, $(\psi_1, \psi_2, \psi_3) = (0, b_1, -b_1)$ and $(\psi_1, \psi_2, \psi_3, \psi_4) = (0, b_1, -b_1, -0.5b_1)$, where the drift has the form $b(x, y, z) = b_1(x) + b_2(z)$, and plot the MSE curves in Figure 3. For $K = 2, 3, 4$ the optimization landscape is much more satisfactory than for $K = 1$, which is the deep FBSDE method. In fact the minima are close to the real $Y_0 = 8.94$. To further indicate that the method is robust to the choice of $K \geq 3$ (and possibly $K = 2$) and $(\psi_1, \ldots, \psi_K)$ we take $K = 3$ and rather arbitrary choices $\psi_1 = b_1 \circ \xi$, $\psi_2 = -b_1 \circ \xi$, $\psi_3 = 0$, for $\xi \in \{x, -\sin(x), x\cos(x), 1 - \exp(\min(1, \max(-1, x)))\}$, where ξ acts coordinate-wise on vectors. We obtain the four loss surfaces in Figure 4, which, based on visual inspection, have the same minima. This robustness is a very important property because when the method is applied to a problem without a reference solution, one must be able to trust that the choice of ψ_1, ψ_2, ψ_3 does not need to be tuned based on comparison with a reference solution. Our results indicate a robustness in the choice of $K \geq 2$ and $(\psi_1, \ldots, \psi_K)$, but it needs to be investigated further, particularly theoretically. The deviation between Y_0 and the optimized y_0^* is the discretization error due to the time stepping. This is seen empirically in the convergence plot in Figure 5 for $\xi(x) = x$, showing the experimental convergence order one, i.e., $|Y_0 - y_0^*(h)| = \mathcal{O}(h)$. The convergence of (X, Y, Z) as processes after Phase II is shown in Section 4.1.2.

Remark 1 (*More on the choice of* $\boldsymbol{\psi}$) The numerical behavior of the method depends on the family $\boldsymbol{\psi} = (\psi_1, \ldots, \psi_K)$ used in Phase I. In practice, we have observed that choosing each ψ_i to be constant in time (or at worst Hölder–1/2 continuous) and Lipschitz continuous in

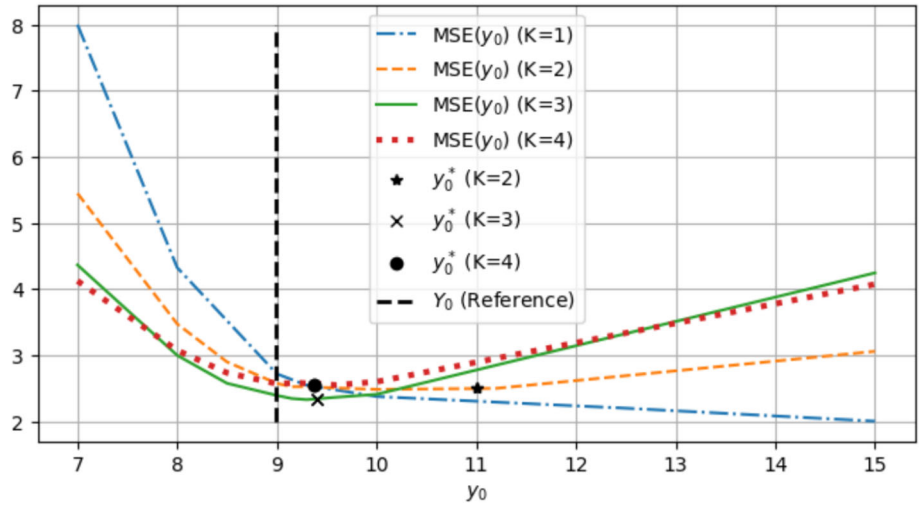

Fig. 3 Four MSE curves plotted for $K = 1, 2, 3, 4$, with $\psi_1 = 0$, $(\psi_1, \psi_2) = (0, b_1)$, $(\psi_1, \psi_2, \psi_3) = (0, b_1, -b_1)$ and $(\psi_1, \psi_2, \psi_3, \psi_4) = (0, b_1, -b_1, -0.5b_1)$, where $b(x, y, z) = b_1(x) + b_2(z)$. In Phase II, y_0^* for $K = 1, 2, 3, 4$ are the initial conditions approximated with Phase I. Note that for $K = 1$, $y_0^* = 28.44$ as can be visualized in Figure 1. For visualization purposes, we have scaled the curves. For $K = 1, 2, 3, 4$, the training times are approximately 400s, 500s, 600s, and 700s, respectively

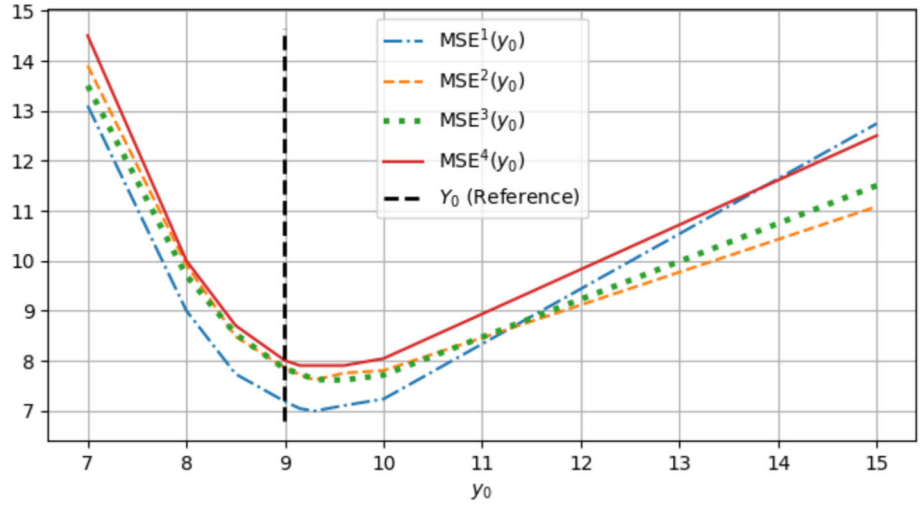

Fig. 4 Four MSE curves plotted for $\psi_1 = b_1 \circ \xi$, $\psi_2 = -b_1 \circ \xi$, $\psi_3 = 0$, for $\xi \in \{x, -\sin(x), x\cos(x), 1 - \exp((x \wedge 1) \vee (-1))\}$, where ξ acts coordinate-wise on vectors. The order coincides with that of the set

the spatial variables leads to stable optimization, as it preserves the qualitative regularity of the modified coefficients b^{ψ} and f^{ψ}. Figures 3–4 illustrate that the approximation of Y_0 is robust with respect to different choices of $\boldsymbol{\psi}$. Enlarging the family by adding new functions (Figure 3) leads the minimizer progressively closer to the true initial condition, and for sufficiently large K the method becomes stable with respect to different choices of $\boldsymbol{\psi}$ (Figure 4). This provides a practical way to assess robustness in situations without a reference

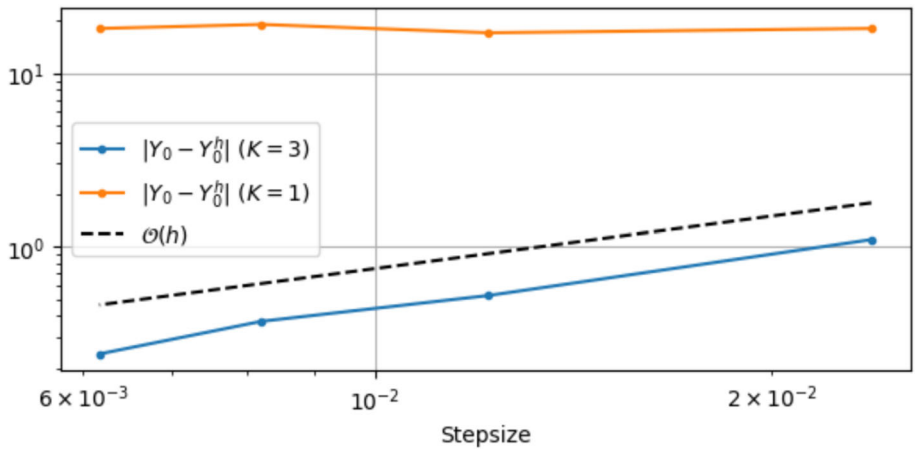

Fig. 5 Empirical convergence plot for approximations of Y_0 using the deep multi-FBSDE method Phase I. Here the case $K = 1$ coincides with the original deep FBSDE method

solution. If Phase I yields consistent values of Y_0 across two (or more) different $\boldsymbol{\psi}$-families, this provides strong evidence that the method produces a reliable approximation of Y_0.

Remark 2 (*Scope of applicability*) The deep multi-FBSDE method is designed to improve robustness for classes of FBSDEs where the classical deep BSDE method may face convergence difficulties. A complete mathematical characterization of all such equations is currently not available, as existing a posteriori error bounds apply only under restrictive structural conditions. Better understanding why the multi-FBSDE formulation converges in cases where the deep BSDE method remains an important direction for future research. In this paper we use the term "robustness" in an empirical sense: the method continues to perform well in cases where the classical deep BSDE method fails, and its output is largely insensitive to the particular choice of $\boldsymbol{\psi} = (\psi_1, \ldots, \psi_K)$ once K is moderately large. Figures 3–4 provide numerical evidence for this behaviour. A theoretical explanation for this robustness is though not yet available. Developing such an analysis, and understanding more precisely why the multi-FBSDE formulation succeeds in situations where the deep BSDE method breaks down, remains an important direction for future research.

3.3 Fully Implementable Algorithms

Without further specification, the variational problems (10) and (11) assume exact optimization over unspecified function classes and exact evaluation of the appearing expectations. To obtain fully implementable numerical schemes, both the initial value and the Markovian control functions are replaced by finite-dimensional parametrizations, and expectations are approximated by Monte Carlo sampling.

In Phase I, corresponding to (10), the initial value y_0 is approximated by a single trainable scalar parameter $\theta_{y_0} \in \mathbb{R}$. The Markovian functions $\{\zeta_0, \ldots, \zeta_{N-1}\}$ are approximated by N fully connected neural networks $\{\phi_0^{\theta_0^I}, \ldots, \phi_{N-1}^{\theta_{N-1}^I}\}$, where each network $\phi_n^{I,\theta_n^I} : \mathbb{R}^d \to \mathbb{R}^d$ is parametrized by weights

and biases θ_n^I. The collection of trainable parameters in Phase I is denoted by $\theta^I := (\theta_{y_0}, \theta_0^I, \ldots, \theta_{N-1}^I) \in \Theta^I$.

Phase II corresponds to (11) and is implemented similarly. The initial value is fixed to the output $y_0^{*,I}$ obtained in Phase I, and only the Markovian control functions are optimized. These are again approximated by fully connected neural networks $\{\phi_0^{\theta_0^{II}}, \ldots, \phi_{N-1}^{\theta_{N-1}^{II}}\}$, with the same input–output structure as in Phase I, but with independent parameters. We collect the trainable parameters in Phase II into $\theta^{II} := (\theta_0^{II}, \ldots, \theta_{N-1}^{II}) \in \Theta^{II}$. In the above, we have implicitly assumed that the neural networks used to approximate ζ_n in Phase I and Phase II share the same architecture and differ only in their parameters. While this assumption can be relaxed, it is a natural choice in practice, since both phases aim to approximate the same underlying Markovian function. Moreover, expectations are approximated by batch Monte Carlo simulation. Let $L_{\text{epochs}} \geq 1$ and $L_{\text{batch}} \geq 1$ denote the number of epochs and batches per epoch, respectively. Let $M_{\text{train}} \geq 1$ and $M_{\text{batch}} \geq 1$ be the size of the training data set and the batch size, respectively, and assume that $M_{\text{train}}/M_{\text{batch}} = L_{\text{batch}} \in \mathbb{N}$. The training data consist of M_{train} independent realizations of the Wiener increments $\Delta W_0, \ldots, \Delta W_{N-1} \sim \mathcal{N}(0, h)$, which are reused over L_{epochs} training epochs. The training is initialized by random sampling of an initial parameter vector $\theta^I \in \Theta^I$ (or $\theta^{II} \in \Theta^{II}$). During each epoch, the training data are partitioned into L_{batch} disjoint batches of size M_{batch}. For each update step within an epoch, we select M_{batch} realizations $\{\Delta W_0(m), \ldots, \Delta W_{N-1}(m)\}_{m=1}^{M_{\text{batch}}}$ from the current epoch's unused training data and update θ by approximately minimizing the corresponding empirical loss function, obtained by replacing the expectation in the variational objective with the sample average over the selected batch. The above training procedure applies to both Phase I and Phase II; the only difference is the set of trainable parameters θ, which corresponds to θ^I or θ^{II} depending on the phase. The algorithm in pseudocode in Algorithm 1.

Phase II is implemented analogously with $K = 1$ and $\psi_1 \equiv 0$, fixing the initial value to $Y_0 = y_0^{*,I}$ and optimizing only over the network parameters $\{\theta_n^{II}\}_{n=0}^{N-1}$ using the loss $\frac{1}{M_{\text{batch}}} \sum_{m=1}^{M_{\text{batch}}} |Y_N(m) - g(X_N(m))|^2$.

3.4 Specification of the Neural Networks

Here, we introduce the neural networks that we use in our implementations in Section 3.3. The generality is kept to a minimum and more general architectures are of course possible. For each $\phi_n^{\theta_n} : \mathbb{R}^d \to \mathbb{R}^d$, a fully connected neural network with three hidden layers with 20 nodes in each layer and a ReLU activation function $\mathfrak{R}(x) = \max(0, x)$ acting elementwise is employed. More precisely, the $\phi_n^{\theta_n}$ is of the form

$$\phi_n^{\theta_n}(x) = W_n^4 \mathfrak{R}\big(W_n^3 \mathfrak{R}\big(W_n^2 \mathfrak{R}(W_n^1 x + b_n^1) + b_n^2\big) + b_n^3\big) + b_n^4,$$

with weight matrices $W_n^1 \in \mathbb{R}^{20\times d}$, $W_n^2 \in \mathbb{R}^{20\times 20}$, $W_n^3 \in \mathbb{R}^{20\times 20}$, $W_n^4 \in \mathbb{R}^{d\times 20}$, and bias vectors $b_n^1, b_n^2, b_n^3 \in \mathbb{R}^{20}$, $b_n^4 \in \mathbb{R}^d$, and $\theta_n = (W_n^1, W_n^2, W_n^3, W_n^4, b_n^1, b_n^2, b_n^3, b_n^4)$, where the matrices are considered vectorized before concatenation.

Input: Initial parameters $\theta^I = (\theta_{y_0}, \theta_0^I, \ldots, \theta_{N-1}^I)$; family $\{\psi_i\}_{i=1}^K$; training data $\{\Delta W_n(m)\}_{n=0}^{N-1}$, $m = 1, \ldots, M_{\text{train}}$; batch size M_{batch}
Output: Updated parameters θ^I
Partition the training data into $L_{\text{batch}} = M_{\text{train}}/M_{\text{batch}}$ disjoint batches;
for $\ell = 1, \ldots, L_{\text{batch}}$ **do**
  Select one batch $\{\Delta W_n(m)\}_{n=0}^{N-1}$, $m = 1, \ldots, M_{\text{batch}}$ (re-indexed locally within the batch);
  for $m = 1, \ldots, M_{\text{batch}}$ (in parallel) **do**
    Set $X_0^{(i)}(m) = x_0$ and $Y_0^{(i)}(m) = \theta_{y_0}$ for all $i = 1, \ldots, K$;
    for $n = 0, \ldots, N-1$ (sequentially) **do**
      Compute $Z_n^{(i)}(m) = \phi_n^{\theta_n^I}(X_n^{(i)}(m))$ for all $i = 1, \ldots, K$;
      Propagate, for all $i = 1, \ldots, K$,

$$
\begin{aligned}
X_{n+1}^{(i)}(m) &= X_n^{(i)}(m) + b_{\psi_i}\big(t_n, X_n^{(i)}(m), Y_n^{(i)}(m), Z_n^{(i)}(m)\big)h + \sigma(t_n, X_n^{(i)}(m))\,\Delta W_n(m), \\
Y_{n+1}^{(i)}(m) &= Y_n^{(i)}(m) - f_{\psi_i}\big(t_n, X_n^{(i)}(m), Y_n^{(i)}(m), Z_n^{(i)}(m)\big)h \\
&\quad + \big\langle Z_n^{(i)}(m), \sigma(t_n, X_n^{(i)}(m))\,\Delta W_n(m)\big\rangle.
\end{aligned}
$$

    end
  end
  Define the empirical loss on the batch by

$$
\mathcal{L}(\theta^I) = \sum_{i=1}^{K} \frac{1}{M_{\text{batch}}} \sum_{m=1}^{M_{\text{batch}}} \left| Y_N^{(i)}(m) - g\big(X_N^{(i)}(m)\big) \right|^2.
$$

  Update parameters:

$$
\theta^I \leftarrow \text{OptimizerStep}\Big(\theta^I, \nabla_{\theta^I}\mathcal{L}(\theta^I)\Big).
$$

end

Algorithm 1: One training epoch of Phase I (deep multi-FBSDE)

4 Numerical Experiments

We present four numerical experiments, two with an associated stochastic optimal control problem, and two without such a connection, all with analytic or semi-analytic reference solutions so that we can analyze the error. The first problem is a controlled Brownian motion in two dimensions, the second problem is a six-dimensional LQ-problem, and the last two examples are FBSDEs related to two-dimensional advection diffusion reaction PDEs, one without a reaction term and one with a linear one. We demonstrate the performance of both the deep FBSDE and the deep multi-FBSDE methods.

In the literature, it is most common to demonstrate deep BSDE-type methods on very high-dimensional problems. However, these problems are most often very symmetric, with solutions being permutation-invariant in its space variables. It is highly questionable how important these examples are for applications and also if they fairly represent more general problems of the same dimensions. We have instead chosen difficult low-dimensional problems for which the classical deep FBSDE methods fail, rather than high-dimensional problems.

In the implementation of the deep FBSDE method and both Phases I–II of the deep multi-FBSDE method, we follow the fully implementable framework introduced in Section 3.3 together with the neural network specification given in Section 3.4. The training data consist of $M_{\text{train}} = 2^{20}$ independent realizations of the k-dimensional Wiener increment vectors $\Delta W_0, \ldots, \Delta W_{N-1}$, where each component is *i.i.d.* and normally distributed with mean zero

and variance h. The training data are reused over $L_{\text{epochs}} = 10$ epochs. Training is initiated by randomly sampling the network parameters and proceeds using the Adam optimizer with mini-batches of size $M_{\text{batch}} = 2^{12}$.

4.1 FBSDEs Stemming From Stochastic Optimal Control Problems

In this section, we look into two main types of stochastic control problems, controlled Brownian motions and linear quadratic Gaussian control problems. Before discussing these specific topics, we define the general form of the stochastic control problem:

$$\begin{cases} \underset{u}{\text{minimize}} \;\; J(u;0,x), \quad \text{where} \\ \mathrm{d}X_t = b(t,X_t,u_t)\mathrm{d}t + \sigma(t,X_t)\mathrm{d}W_t, \\ J(u;t,x) = \mathbb{E}\Big[\int_t^T f(s,X_s,u_s)\mathrm{d}s + g(X_T)\,\Big|\, X_t = x\Big], \quad t\in[0,T],\ x\in\mathbb{R}^d. \end{cases} \tag{12}$$

The controls u belong to a suitable subset of adapted stochastic processes, and we refrain from details The value function associated with the control problem is formally defined as

$$v(t,x) = \inf_u J(u;t,x).$$

Under sufficient regularity assumptions on b, σ, f, g, a solution to the following Hamilton–Jacobi–Bellman (HJB) equation exists and coincides with the value function of the control problem

$$\begin{aligned} \frac{\partial v}{\partial t} + \frac{1}{2}\mathrm{Tr}\big(\sigma^\top \sigma \mathrm{Hess}_x v\big) + \inf_u \big[\langle \mathrm{D}_x v, b\rangle + f\big] &= 0, \quad (t,x)\in[0,T)\times\mathbb{R}^d, \\ v(T,x) &= g(x), \quad x\in\mathbb{R}^d. \end{aligned} \tag{13}$$

In this paper we tacitly assume such regularity. The resulting HJB-equation is a semilinear parabolic PDE of the type discussed in Section 2; therefore, it admits an equivalent formulation as an FBSDE.

4.1.1 Controlled Brownian Motions with General Terminal Cost

In this section, we set $k = d = \ell \in \mathbb{N}$, $x_0 \in \mathbb{R}^d$, and $\sigma, r \in \mathbb{R}^+$. For $t\in[0,T]$, $x\in\mathbb{R}^d$, and $u\in\mathbb{R}^\ell$, the drift, diffusion and cost fu nctions are defined, as follows,

$$b(t,x,u) = u, \quad \sigma(t,x) = \sigma I_d, \quad f(t,x,u) = \frac{r}{2}\|u\|^2, \tag{14}$$

together with a flexibly chosen terminal cost function $g\colon \mathbb{R}^d \to \mathbb{R}$. Straightforward calculations yield the optimal control $u^* = -\frac{1}{r}\mathrm{D}_x v = -\frac{1}{r}Z$, from which we obtain the FBSDE

$$\begin{aligned} X_t &= x_0 - \int_0^t \frac{1}{r} Z_s \mathrm{d}s + \sigma W_t, \\ Y_t &= g(X_T) + \int_t^T \frac{1}{2r}\|Z_s\|^2 \mathrm{d}s - \int_t^T \langle Z_s, \sigma\,\mathrm{d}W_s\rangle, \quad t\in[0,T]. \end{aligned} \tag{15}$$

Using the Cole–Hopf transformation, the unique Y-component of the solution to the above FBSDE is

$$Y_t = -r\sigma^2 \log\left(\mathbb{E}\big[\mathrm{e}^{-\frac{1}{r\sigma^2}g(X_t+\sigma\sqrt{T-t}\,\xi)}\big|X_t\big]\right), \quad t\in[0,T], \tag{16}$$

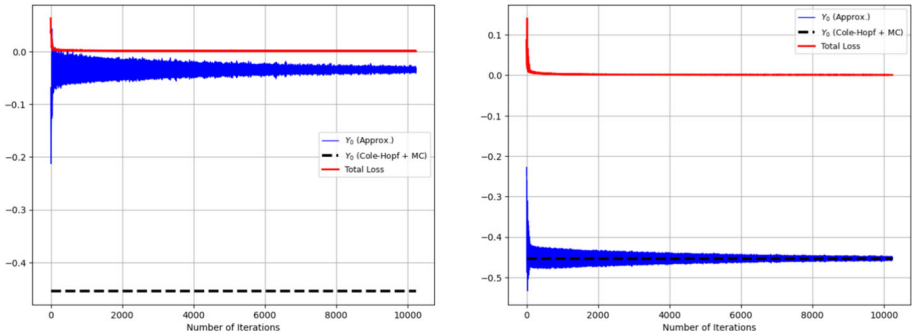

Fig. 6 Loss and Y_0 during training for the deep FBSDE method (left) and the deep multi-FBSDE method (right). "Number of iterations" refers to gradient updates. The losses of the first two iterations were removed due to large values and for better visual representation

where ξ is a standard d-dimensional normally distributed random vector.

The FBSDE (15) can then be approximated using Monte Carlo sampling. It is important to note that, to approximate $(Y_t(\omega))_{t\in[0,T]}$, nested expectations are required along the trajectory of $(X_t(\omega))_{t\in[0,T]}$. Moreover, since X is coupled with Z in the drift component, a straightforward application of this strategy to approximate Y at times other than $t = 0$ (which is independent of the trajectory of X) is challenging. However, if our approximation of (X, Y, Z) is accurate, it should follow that our approximation of Y aligns with the Monte Carlo approximation of Y using (16), along the simulated trajectories of X.

Problem specific settings:
For the deep multi-FBSDE method, we opt for $K = 2$, setting $\psi_1(t, x, y, z) = 0$ and $\psi_2(t, x, y, z) = \frac{r_1^2}{2r_2}z$. This configuration implies that $(X^{\psi_1}, Y^{\psi_1}, Z^{\psi_1})$ is the solution to (15), whereas $(X^{\psi_2}, Y^{\psi_2}, Z^{\psi_2})$ satisfies the following decoupled equation:

$$\begin{aligned} X_t^{\psi_2} =& x_0 + \sigma W_t, \\ Y_t^{\psi_2} =& g(X_T^{\psi_2}) - \int_t^T \frac{1}{2r}\|Z_s^{\psi_2}\|^2 \mathrm{d}s - \int_t^T \left\langle Z_s^{\psi_2}, \sigma \mathrm{d}W_s \right\rangle, \quad t \in [0, T]. \end{aligned} \tag{17}$$

For this relatively straightforward problem, we have observed that only the first phase of the deep multi-FBSDE method is necessary, so Phase II is skipped.

We set $T = 0.5$, $d = \ell = k = 2$, $r = 1$, and $\sigma = 0.25$. The initial state is set at $x_0 = (-0.1, 0.1)^\top$. We employ a terminal cost function $g(x) = -|x_1 - x_2|$, where $x = (x_1, x_2)^\top$. This configuration aims to control the two components of the Brownian motion such that they diverge as much as possible by time $t = T$, while minimizing the use of control force. The discretization of the problem is set with $N = 40$.

Results and discussion:
Figure 6 illustrates the loss and the approximation of Y_0 during training for both the deep FBSDE method and the deep multi-FBSDE method. Notably, while the loss approaches zero for both methods, only the deep multi-FBSDE method converges to the true Y_0.

The deep FBSDE method fails to converge to the true solution. Although we do not have theoretical understanding for the reason, to gain some partial insight, we illustrate in Figure 7 the sample mean and a representative path of X and Y, for the two numerical methods and

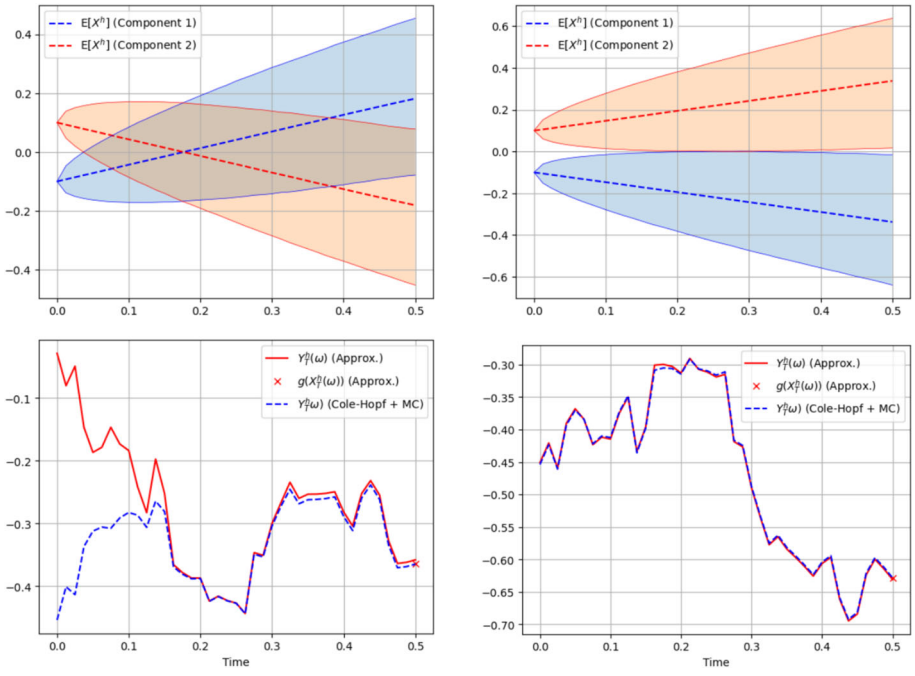

Fig. 7 The upper plots display the empirical mean of X as well as the 95th and 5th percentiles for the deep FBSDE method (left) and the deep multi-FBSDE method (right). The lower plots show a representative trajectory of the approximate Y for the deep FBSDE method (left) and the deep multi-FBSDE method (right), compared with their semi-analytic counterparts

the reference solution. We see that the optimal strategy pushes X^1 to lower values and X^2 to higher values, which is intuitive since $X_0^1 < X_0^2$ and the task is to maximize $|X_T^1 - X_T^2|$ with low control cost. The deep FBSDE method, on the other hand, causes X^1 and X^2 to instead cross each other, and naturally uses a more expensive control for the task, reading the initially high values of the Y-process. After the crossing, the control seems to resemble the optimal control well, which can be read from the resulting Y-process being close to the reference. A similar MSE-analysis as in Figure 1 (omitted) shows that the problem with the deep FBSDE solution is due to a bad global minimum.

4.1.2 Linear Quadratic Gaussian Control Problems

Among all stochastic control problems, the linear quadratic Gaussian control problem is notably the most structured and extensively studied, see, for example, [4]. For our purposes, it presents a closed-form analytic solution, allowing us to benchmark our numerical approximations against it.

Consider the following settings: let $k = d \in \mathbb{N}$, $\ell \in \mathbb{N}$, $x_0 \in \mathbb{R}^d$, $A, \sigma \in \mathbb{R}^{d\times d}$, $R_x, G \in \mathbb{S}_+^d$, $R_u \in \mathbb{S}_+^\ell$, and $B \in \mathbb{R}^{d\times\ell}$ with full rank, alongside $C \in \mathbb{R}^d$. For $t \in [0, T]$, $x \in \mathbb{R}^d$, and $u \in \mathbb{R}^\ell$ let

$$b(t, x, u) = A(C - x) + Bu, \quad f(t, x, u) = \langle R_x x, x\rangle + \langle R_u u, u\rangle \quad g(x) = \langle Gx, x\rangle.$$

The above, together with (12), defines the state equation and cost functional for a LQ control problem. The optimal feedback control, minimizing the Hamiltonian, $\inf_{u\in U}\langle \mathrm{D}_x v, Bu\rangle + \langle R_u u, u\rangle$, is given by

$$u_t^* = -\frac{1}{2}R_u^{-1}B^\top \mathrm{D}_x v(t, X_t). \tag{18}$$

Recalling that v is the solution to the associated HJB-equation, its solution can be expressed as

$$v(t, x) = x^\top P(t)x + x^\top Q(t) + R(t), \tag{19}$$

where (P, Q, R) are the solutions to the system of ordinary differential equations,

$$\begin{aligned}
&\dot{P}(t) - A^\top P(t) - P(t)A - P(t)BR_u^{-1}B^\top P(t) + R_x = \mathbf{0}_{d\times d},\\
&\dot{Q}(t) + 2P(t)AC - A^\top Q(t) - P(t)BR_u^{-1}B^\top Q(t) = \mathbf{0}_d,\\
&\dot{R}(t) + \mathrm{Tr}\big(\sigma\sigma^\top P(t)\big) + Q(t)^\top AC - \frac{1}{4}Q(t)^\top BR_u^{-1}B^\top Q(t) = 0, \quad t\in[0, T],\\
&P(T) = G; \quad Q(T) = \mathbf{0}_d; \quad R(T) = 0.
\end{aligned}$$

We refer to the entire system colloquially as the Riccati equation, although the first equation is strictly a matrix Riccati equation. The gradient of v, $\mathrm{D}_x v(t, x) = 2P(t)x + Q(t)$, leads to the associated FBSDE:

$$\begin{aligned}
X_t =& x_0 + \int_0^t \big[A(C - X_s) - \frac{1}{2}BR_u^{-1}B^\top Z_s\big]\mathrm{d}s + \int_0^t \sigma\,\mathrm{d}W_s,\\
Y_t =& \langle GX_T, X_T\rangle + \int_t^T \Big(\langle R_x X_s, X_s\rangle + \frac{1}{4}\big\langle R_u^{-1}B^\top Z_s, B^\top Z_s\big\rangle\Big)\mathrm{d}s\\
&- \int_t^T \langle Z_s, \sigma\,\mathrm{d}W_s\rangle, \quad t\in[0, T].
\end{aligned} \tag{20}$$

The solution to (20) is provided by

$$Y_t = X_t^\top P(t)X_t + X_t^\top Q(t) + R(t); \quad Z_t = 2P(t)X_t + Q(t). \tag{21}$$

In our experiments, we use the Euler approximation of the Riccati equation with 160×2^7 time steps, and for X, we use 160 time steps. The processes (Y, Z) are approximated iteratively, as per (21).

Problem specific settings:
For the deep multi-FBSDE method, we set $K = 3$ and define $\psi_1(t, x, y, z) = A(C - x)$, $\psi_2(t, x, y, z) = -A(C - x)$, and $\psi_3(t, x, y, z) = 0$. With this configuration, we incorporate both the doubling effect (through ψ_2) and the removal of the mean-reverting term (through ψ_1). Furthermore, since ψ_3 is identically zero, the original control problem is also explicitly included in the training.

The matrices used for the state equation are the same, up to σ, R_x and G, as in [1, 24]. In fact $R_x = \mathrm{diag}(5, 1, 5, 1, 5)$ is $G = \mathrm{diag}(1, 5, 1, 5, 1)$ in the implementation of [1], but reads wrongly $R_x = \mathrm{diag}(25, 1, 25, 1, 25)$ and $G = \mathrm{diag}(1, 25, 1, 25, 1)$ in the presentation

of that paper). Our setting reads

$$A = \mathrm{diag}([1, 2, 3, 1, 2, 3]), \quad B = \begin{pmatrix} 1 & -1 \\ 1 & 1 \\ 0.5 & 1 \\ 1 & -1 \\ 0 & -1 \\ 0 & 1 \end{pmatrix},$$
$$C = \mathrm{diag}([-0.2, -0.1, 0, 0, 0.1, 0.2]), \quad \sigma = \mathrm{diag}([0.2, 1, 0.2, 1, 0.2, 1]),$$
$$x_0 = (0.1, 0.1, 0.1, 0.1, 0.1, 0.1)^\top, \quad T = 0.5,$$

and

$$R_x = \mathrm{diag}([25, 1, 25, 1, 25, 1]), \; R_u = \mathrm{diag}([1, 1]), \; G = \mathrm{diag}([1, 25, 1, 25, 1, 25]).$$

In [24], the authors use the deep FBSDE method on their control problem, to approximate the solution of a BSDE derived from the stochastic maximum principle. However, this problem is structurally less complex because, at each time t, the control process in the associated BSDE is constant in the state space. It remains to understand to what extent this approach would generalize to more complex problems where the control process is non-constant.

Results and discussion:
Figure 8 clearly demonstrates that our method yields accurate approximations for (X, Y, Z), both pathwise and in distribution. We refrain from comparing these results with those obtained via the deep FBSDE method, as the latter produces completely incorrect approximations for this problem. The approximate initial condition obtained by the deep FBSDE method is around 28.44, whereas the true value is approximately 9 (see Figure 1).

Part of the numerical experiments of this problem is the convergence to Y_0, shown in Figure 5, as the time step h tends to zero, or in the experiment for $N \in \{20, 40, 60, 80\}$ time steps. To complete the empirical convergence study, Figure 9 shows the convergence of X and Y with orders 1 and 0.5, respectively, and a decreasing error in Z without a clear rate. We note that convergence orders 1 and 0.5 are the expected rates for forward SDEs with additive and multiplicative noise, respectively. The norms for the convergence are

$$\|A\|_{\mathcal{S}^2_{h,M}(\mathbb{R}^q)} = \max_{n\in\{0,1,\ldots,N\}} \left(\frac{1}{M} \sum_{m=1}^{M} \|A_n(m)\|^2 \right)^{\frac{1}{2}},$$
$$\|A\|_{\mathcal{H}^2_{h,M}(\mathbb{R}^q)} = \frac{1}{N} \sum_{n=0}^{N-1} \left(\frac{1}{M} \sum_{m=1}^{M} \|A_n(m)\|^2 \right)^{\frac{1}{2}}.$$

Here, $A(m) = \{A_1(m), A_2(m), \cdot, A_N(m)\}$, $m = 1, 2, \ldots, M$, are *i.i.d.* realizations of some adapted stochastic processes A on the grid.

4.2 FBSDEs From Advection Diffusion Reaction PDEs

In this section, we consider semilinear parabolic PDEs with quadratic gradient nonlinearity.

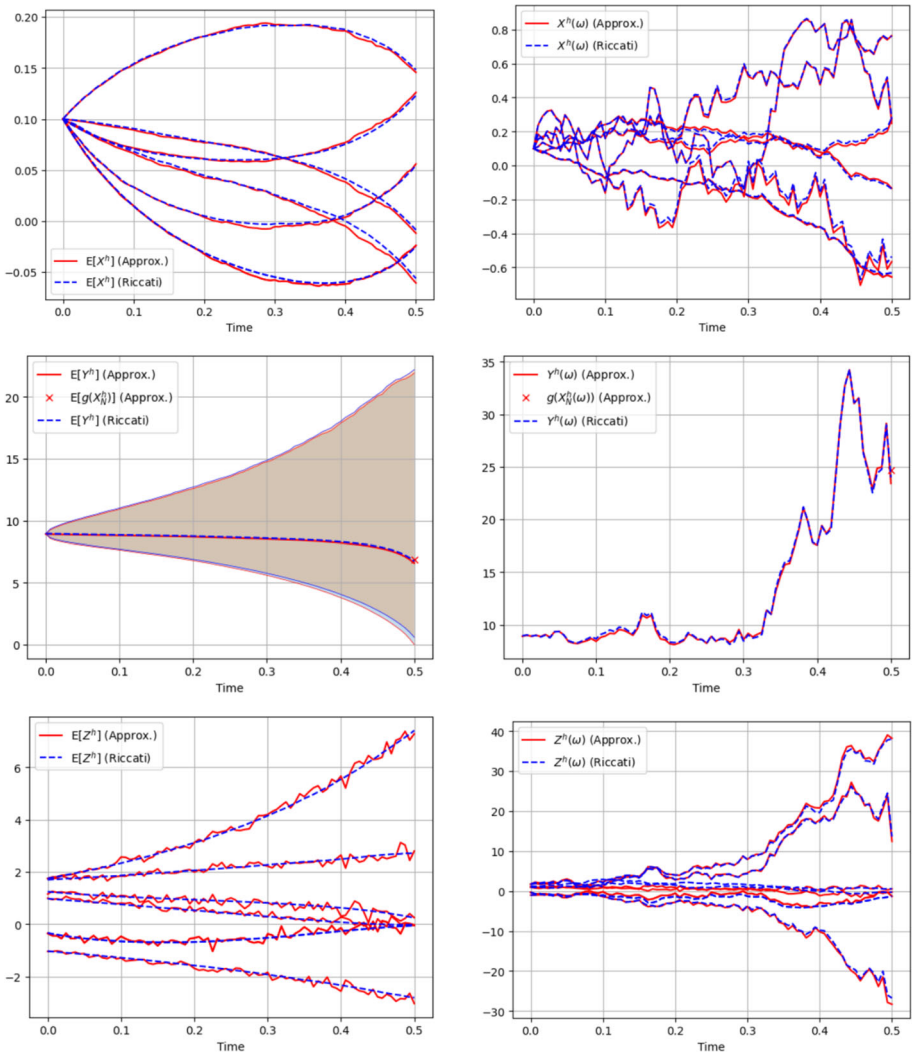

Fig. 8 Empirical mean of solutions (left) and a single solution trajectory (right) compared to the reference solution for the LQ-control problem from Section 4.1.2. The shaded area represents an empirical credible interval for Y, defined as the area between the 5:th and the 95:th percentiles at each time point. We do not include credible intervals for X and Z in this figure to facilitate visualization. For X and Z, we see one realization of each of the six components

Let $\alpha, \beta \in \mathbb{R}^+$, $\gamma \in \mathbb{R}$, and $x_0 \in \mathbb{R}^d$. For a general function $g : \mathbb{R}^d \to \mathbb{R}$ we define

$$\begin{aligned} \frac{\partial v}{\partial t} + \alpha \Delta v + 2\beta \|\mathrm{D}_x v\|^2 - \gamma v &= 0, \quad (t, x) \in [0, T) \times \mathbb{R}^d, \\ v(T, x) &= g(x), \quad x \in \mathbb{R}^d. \end{aligned} \tag{22}$$

The above PDE is structurally similar to a Hamilton–Jacobi–Bellman (HJB) equation. However, since $\beta > 0$, it cannot be written in the form (13), does not have an associate stochastic control problem, and hence does not represent an HJB equation. We consider the equivalent FBSDE

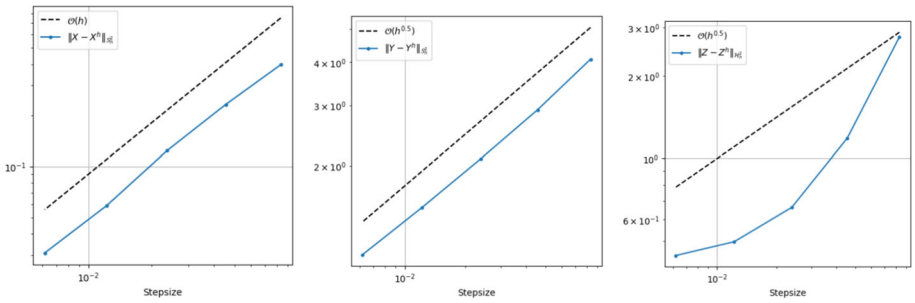

Fig. 9 Empirical convergence plots for approximations of (X, Y, Z) using the deep multi-FBSDE method phase II

$$\begin{aligned} X_t =& x_0 + \int_0^t \beta Z_t \mathrm{d}s + \sqrt{2\alpha} W_t, \\ Y_t =& g(X_T) + \int_t^T \left(\beta \|Z_s\|^2 - \gamma Y_s\right) \mathrm{d}s - \sqrt{2\alpha} \int_t^T \langle Z_s, \mathrm{d}W_s \rangle, \quad t \in [0, T]. \end{aligned} \tag{23}$$

We again employ the Cole–Hopf transformation to derive a reference solution. Specifically, defining $\rho(t, x) = \log\left(\frac{\alpha}{2\beta} v(t, x)\right)$ leads directly to the simplified PDE

$$\begin{aligned} \frac{\partial \rho}{\partial t}(t, x) + \alpha \Delta \rho - \gamma \rho \log \rho =& 0, \quad (t, x) \in [0, T) \times \mathbb{R}^d, \\ \rho(T, x) =& \exp\left(\frac{2\beta}{\alpha} g(x)\right), \quad x \in \mathbb{R}^d. \end{aligned} \tag{24}$$

Back transformation yields $\rho(t, x) = \exp(\frac{2\beta}{\alpha} v(t, x))$. In the special case $\gamma = 0$, we have for $t \in [0, T]$

$$Y_t = -\frac{\alpha}{2\beta} \log\left(E\left[\mathrm{e}^{-\frac{2\beta}{\alpha} g(X_t + \sqrt{2\alpha(T-t)}\,\xi)} \middle| X_t\right]\right), \tag{25}$$

where ξ is a standard d-dimensional normally distributed random vector.

Problem specific settings:
For the deep multi-FBSDE method, we opt for $K = 2$, setting $\psi_1(t, x, y, z) = 0$ and $\psi_2(t, x, y, z) = -\frac{\beta}{2} z$. We employ a terminal condition $g(x) = -|x_1 - x_2|$, where $x = (x_1, x_2)^\top$. The discretization of the problem is set with $N = 40$. In our first experiment, we set $\alpha = 0.0315$, $\beta = 0.6$, $x_0 = (-0.1, 0.1)^\top$, and $\gamma = 0$ and use (25) to compute a reference solution. In our second experiment, we set $\alpha = 0.0315$, $x_0 = (-0.1, 0.1)^\top$, $\gamma = 1$, and let β vary between 0.00125 and 5.

For the first experiment, the reference solution is computed with a Monte Carlo approximation of (25). For the second experiment, we use a finite difference scheme to approximate (24). More specifically, we approximate the solutions on a spatial domain $[-2, 2] \times [-2, 2]$ with artificial Dirichlet boundary conditions.

Results and discussion:
Figure 10 displays the approximate initial condition during training for both the deep FBSDE and deep multi-FBSDE methods. The figure clearly illustrates that the deep FBSDE method yields inaccurate approximations, whereas the deep multi-FBSDE method produces highly

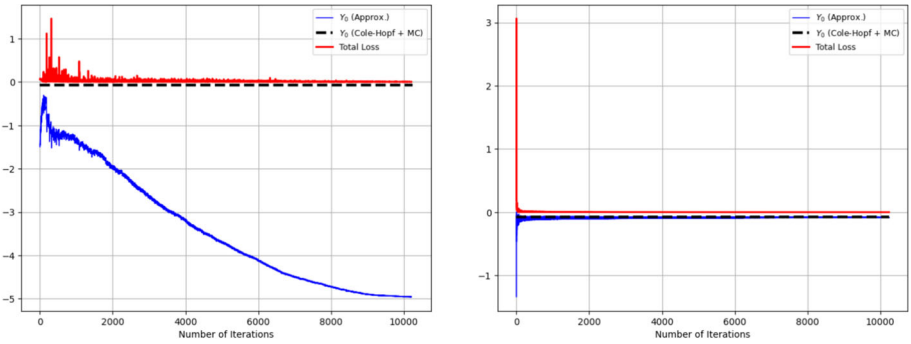

Fig. 10 Loss and approximation of Y_0 during training for the deep FBSDE method (left) and the deep multi-FBSDE method (right)

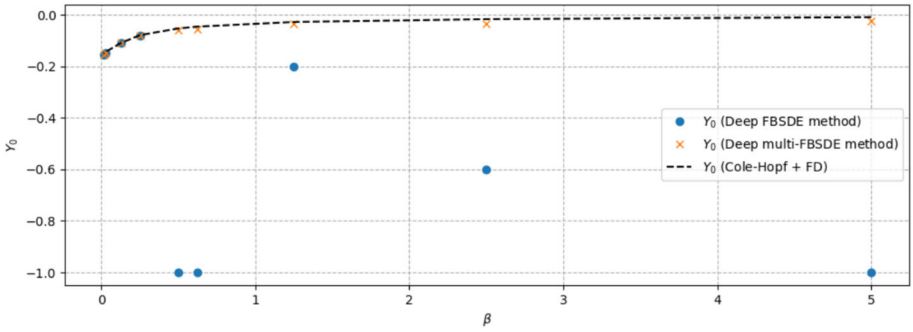

Fig. 11 Approximate initial values for different values of β. Values of Y_0 smaller than -1 have been set to -1 for the sake of visibility

accurate results. For the second experiment, we explore the accuracy of the deep FBSDE method and the deep multi-FBSDE method for different values of β. In Figure 11, we see that for $\beta > 0.25$, the deep FBSDE fails to provide accurate approximations. The deep multi-FBSDE method provides accurate approximations for all β tested.

5 Conclusions and Potential Future Research Directions

In this paper, we introduced a two-phase deep-learning-based method, the deep multi-FBSDE method, for approximating coupled forward–backward stochastic differential equations (FBSDEs). Our approach focuses on scenarios where the standard deep FBSDE method [21] does not converge. The main characteristic of our method is that its loss function is composed of the sum of multiple deep FBSDE loss functions, each originating from a FBSDE satisfying the same partial differential equation (PDE) as the original FBSDE, but with transformed drift and driver. The method was demonstrated on two stochastic control problems and two FBSDEs related to advection diffusion reaction PDEs, unrelated to stochastic control. For neither of the problems, the deep FBSDE method worked, but the deep multi-FBSDE method did. Overall, our results suggest that focusing on a family of equivalent FBSDEs offers a promising path to overcome convergence challenges in deep FBSDE methods, thus broadening the scope of problems in finance, control theory, and related areas where these methods can be applied effectively.

Although the method seems to work very well on challenging FBSDEs, we still do not understand the reason for this and why the deep FBSDE method fails. There is thus a need to both understand theoretically for what problems the deep FBSDE does not work and why, and why the deep multi-FBSDE method does work for these problems. From a practical perspective, the choice of K and $\psi_1, \ldots, \psi_K$ in our algorithm is currently somewhat *ad hoc*. While this works for moderately complex problems, a more systematic approach to selecting K and $\psi_1, \ldots, \psi_K$ would be especially beneficial when dealing with high-dimensional systems or models where it is difficult to anticipate the impact of different choices. Developing such a framework constitutes a natural next step toward enhancing both the reliability and the applicability of the proposed method. Furthermore, extending the methodology to broader classes of PDE-driven problems and exploring theoretical guarantees for large-scale or high-dimensional settings would deepen our understanding of its robustness and versatility.

One possible direction toward a more systematic selection of $\psi_1, \ldots, \psi_K$ is to replace manual specification by an adaptive procedure based on adversarial training. Inspired by recent work such as SOC-MartNet [13], one could parametrize ψ by a neural network that is updated during training to expose directions in which the terminal condition residual is relatively large, while the main network parameters are optimized to reduce this residual. Such an approach would provide an automatic mechanism for exploring families of equivalent FBSDEs and could be particularly useful in high-dimensional settings. A detailed investigation of adversarially selecting ψ and its impact on stability and convergence is left for future work.

Acknowledgements Part of this work was carried out during a two-month postdoctoral visit of Kristoffer Andersson (KA) at the University of Utrecht, whose hospitality and support are gratefully acknowledged. KA also acknowledges financial support from RiBa 2022 Research Fund provided by the University of Verona. We very much thank Per Ljung att Saab for his careful reading and many comments that helped us improve the manuscript, to Karl Hammar at Saab and Chalmers for pointing out an error, and to Benjamin Svedung Wettervik at Saab for some interesting reflections.

Funding This research received no external funding.

Data Availability No external datasets were generated or analysed in this study.

Declarations

Conflict of interest The authors declare that they have no competing interests.

Ethical approval Not applicable.

Consent to participate Not applicable.

Consent for publication Not applicable.

Code availability The code that supports the findings is available from the corresponding author upon reasonable request.

References

1. Andersson, K., Andersson, A., Oosterlee, C.W.: Convergence of a robust deep FBSDE method for stochastic control. SIAM J. Sci. Comput. **45**(1), A226–A255 (2023)
2. Andersson, K., Gnoatto, A.: Multi-layer Deep xVA: Structural Credit Models, Measure Changes and Convergence Analysis. arXiv preprint arXiv:2502.14766 (2025)
3. Andersson, K., Gnoatto, A., Patacca, M., Picarelli, A.: A deep solver for BSDEs with jumps. SIAM J. Financ. Math. **16**(3), 875–911 (2025)
4. Åström, K.J.: Introduction to stochastic control theory. Courier Corporation (2012)
5. Beck, C., Becker, S., Cheridito, P., Jentzen, A., Neufeld, A.: Deep splitting method for parabolic PDEs. SIAM J. Sci. Comput. **43**(5), A3135–A3154 (2021)
6. Beck, C., Becker, S., Grohs, P., Jaafari, N., Jentzen, A.: Solving the Kolmogorov PDE by means of deep learning. J. Sci. Comput. **88**(3), 1–28 (2021)
7. Beck, C., Weinan, E., Jentzen, A.: Machine learning approximation algorithms for high-dimensional fully nonlinear partial differential equations and second-order backward stochastic differential equations. J. Nonlinear Sci. **29**, 1563–1619 (2019)
8. Beck, C., Hutzenthaler, M., Jentzen, A., Kuckuck, B.: An overview on deep learning-based approximation methods for partial differential equations. Discrete Continuous Dyn. Syst. – Series B **28**(6), 3697–3746 (2023). https://doi.org/10.3934/dcdsb.2022238
9. Bender, C., Zhang, J.: Time discretization and markovian iteration for coupled fbsdes. Annal. Appl. Probab **18**(1), 143–177 (2008)
10. Berner, J., Grohs, P., Jentzen, A.: Analysis of the generalization error: Empirical risk minimization over deep artificial neural networks overcomes the curse of dimensionality in the numerical approximation of Black-Scholes partial differential equations. SIAM J. Math. Data Sci. **2**(3), 631–657 (2020)
11. Bussell, D., García-Trillos, C.A.: Deep multi-step mixed algorithm for high dimensional non-linear PDEs and associated BSDEs. arXiv preprint arXiv:2308.14487 (2023)
12. Cai, W., Fang, S., Zhou, T.: Deep random difference method for high dimensional quasilinear parabolic partial differential equations. arXiv preprint arXiv:2506.20308 (2025)
13. Cai, W., Fang, S., Zhou, T.: SOC-MartNet: A Martingale Neural Network for the Hamilton-Jacobi-Bellman Equation Without Explicit inf H in Stochastic Optimal Controls. SIAM J. Sci. Comput. **47**(4), C795–C819 (2025)
14. Chan-Wai-Nam, Q., Mikael, J., Warin, X.: Machine learning for semi linear PDEs. J. Sci. Comput. **79**(3), 1667–1712 (2019)
15. Chessari, J., Kawai, R., Shinozaki, Y., Yamada, T.: Numerical methods for backward stochastic differential equations: A survey. Probab. Surv. **20**, 486–567 (2023)
16. Elbrächter, D., Grohs, P., Jentzen, A., Schwab, C.: DNN expression rate analysis of high-dimensional PDEs: Application to option pricing. Constructive Approximation pp. 1–69 (2021)
17. Fang, F., Oosterlee, C.W.: A novel pricing method for european options based on fourier-cosine series expansions. SIAM J. Sci. Comput. **31**(2), 826–848 (2009)

18. Fujii, M., Takahashi, A., Takahashi, M.: Asymptotic expansion as prior knowledge in deep learning method for high dimensional BSDEs. Asia-Pacific Finan. Markets. **26**(3), 391–408 (2019)
19. Germain, M., Pham, H., Warin, X.: Approximation error analysis of some deep backward schemes for nonlinear PDEs. SIAM J. Sci. Comput. **44**(1), A28–A56 (2022)
20. Grohs, P., Hornung, F., Jentzen, A., von Wurstemberger, P.: A proof that artificial neural networks overcome the curse of dimensionality in the numerical approximation of Black–Scholes partial differential equations, Memoirs of the American Mathematical Society, vol. 284 (1410). American Mathematical Society, Providence, RI (2023). https://doi.org/10.1090/memo/1410
21. Han, J., Jentzen, A., Weinan, E.: Solving high-dimensional partial differential equations using deep learning. Proc. Natl. Acad. Sci. **115**(34), 8505–8510 (2018)
22. Han, J., Long, J.: Convergence of the deep BSDE method for coupled FBSDEs. Probab. Uncertain. Quant. Risk **5**(1), 1–33 (2020)
23. Henry-Labordere, P.: Deep primal-dual algorithm for BSDEs: Applications of machine learning to CVA and IM. Available at SSRN 3071506 (2017)
24. Huang, Z., Negyesi, B., Oosterlee, C.W.: Convergence of the deep BSDE method for stochastic control problems formulated through the stochastic maximum principle. Math. Comput. Simul. **227**, 553–568 (2025)
25. Huang, Z., Oosterlee, C.W.: Convergence of the Markovian iteration for coupled FBSDEs via a differentiation approach. arXiv preprint arXiv:2504.02814 (2025)
26. Huré, C., Pham, H., Warin, X.: Deep backward schemes for high-dimensional nonlinear PDEs. Math. Comput. **89**(324), 1547–1579 (2020)
27. Hutzenthaler, M., Jentzen, A., Kruse, T., Nguyen, T.A.: A proof that rectified deep neural networks overcome the curse of dimensionality in the numerical approximation of semilinear heat equations. SN Partial Differ. Equ. Appl **1**(2), 1–34 (2020)
28. Jentzen, A., Salimova, D., Welti, T.: A proof that deep artificial neural networks overcome the curse of dimensionality in the numerical approximation of Kolmogorov partial differential equations with constant diffusion and nonlinear drift coefficients. Commun. Math. Sci. **19**(5), 1167–1205 (2021)
29. Ji, S., Peng, S., Peng, Y., Zhang, X.: Three algorithms for solving high-dimensional fully coupled FBSDEs through deep learning. IEEE Intell. Syst. **35**(3), 71–84 (2020)
30. Ji, S., Peng, S., Peng, Y., Zhang, X.: A novel control method for solving high-dimensional Hamiltonian systems through deep neural networks. SIAM J. Sci. Comput. **47**(4), C873–C898 (2025)
31. Jiang, Y., Li, J.: Convergence of the deep BSDE method for FBSDEs with non-Lipschitz coefficients. Probab. Uncertain. Quant. Risk **6**(4), 391–408 (2021)
32. Kapllani, L., Teng, L.: A backward differential deep learning-based algorithm for solving high-dimensional nonlinear backward stochastic differential equations. IMA J. Num. Anal. pp. draf022 (2025)
33. Negyesi, B., Andersson, K., Oosterlee, C.W.: The One Step Malliavin scheme: new discretization of BSDEs implemented with deep learning regressions. IMA J. Numer. Anal. **44**(6), 3595–3647 (2024)
34. Negyesi, B., Huang, Z., Oosterlee, C.W.: Generalized convergence of the deep BSDE method: a step towards fully-coupled FBSDEs and applications in stochastic control. arXiv preprint arXiv:2403.18552 (2024)
35. Raissi, M.: Forward–backward stochastic neural networks: deep learning of high-dimensional partial differential equations. In: Peter Carr Gedenkschrift: Research Advances in Mathematical Finance, pp. 637–655. World Scientific (2024)
36. Reisinger, C., Stockinger, W., Zhang, Y.: A posteriori error estimates for fully coupled Mckean-Vlasov forward-backward SDEs. IMA J. Numer. Anal. **44**(4), 2323–2369 (2024)
37. Wang, Y., Ni, Y.H.: Deep BSDE-ML Learning and Its Application to Model-Free Optimal Control. Preprint arXiv:2201.01318 (2022)
38. Zhang, J.: Backward stochastic differential equations. Springer, Berlin/Heidelberg, Germany (2017)