# A deep learning approach to the probabilistic numerical solution of path-dependent partial differential equations

Jiang Yu Nguwi[*] Nicolas Privault[†]

Division of Mathematical Sciences

School of Physical and Mathematical Sciences

Nanyang Technological University

21 Nanyang Link, Singapore 637371

June 30, 2022

**Abstract**

Recent work on Path-Dependent Partial Differential Equations (PPDEs) has shown that PPDE solutions can be approximated by a probabilistic representation, implemented in the literature by the estimation of conditional expectations using regression. However, a limitation of this approach is to require the selection of a basis in a function space. In this paper, we overcome this limitation by the use of deep learning methods, and we show that this setting allows for the derivation of error bounds on the approximation of conditional expectations. Numerical examples based on a two-person zero-sum game, as well as Asian and barrier option pricing, are presented. In comparison with other deep learning approaches, our algorithm appears to be more accurate, especially in large dimensions.

*Keywords*: Path-dependent partial differential equations (PPDEs), deep neural networks, numerical methods for PPDEs.

*Mathematics Subject Classification (2020):* 65C05, 60H30.

[*]nguw0003@e.ntu.edu.sg

[†]nprivault@ntu.edu.sg

## 1 Introduction

Fully nonlinear PPDEs of the form

$$\begin{cases}\partial_t u(t,\omega) + b(t,\omega)\cdot\partial_\omega u(t,\omega) + \dfrac{1}{2}\sigma\sigma^\top(t,\omega) : \partial_{\omega\omega}u(t,\omega) + F\left(\cdot, u, \sigma^\top\partial_\omega u, \sigma^\top\partial^2_{\omega\omega}u\sigma\right)(t,\omega) = 0,\\ u(T,\omega) = g(\omega),\end{cases} \tag{1.1}$$

have been introduced in Peng (2011) and their well-posedness in the sense of viscosity solutions have been studied in Ekren et al. (2014; 2016;). Here, $\omega$ is in the set of $\mathbb{R}^d$-valued continuous paths, $b(t,\omega)$ is $\mathbb{R}^d$-valued, $\sigma(t,\omega)$ takes values in the space of invertible $d\times d$ matrices, and $F$ is a real-valued function on $\mathbb{R}_+\times\mathbb{R}^3$ satisfying Assumption 1 below. The precise meaning of the partial derivatives $\partial_t u(t,\omega)$, $\partial_\omega u(t,\omega)$ and $\partial_{\omega\omega}u(t,\omega)$, which are connected to the horizontal and vertical derivatives of functional Itô calculus introduced in Dupire (2009), will be discussed in Section 2.

PPDEs of the type (1.1) have recently been the object of increased attention due to their ability to model control and pricing problems in a non-Markovian setting, see e.g. Tang and Zhang (2015), Jacquier and Oumgari (2019), Viens and Zhang (2019).

Nevertheless, a large class of PPDE is not analytically solvable, and one has to rely on the numerical solution. In Ren and Tan (2017), a probabilistic scheme based on Fahim et al. (2011) has been proposed, and was proved to converge to the viscosity solution of PPDE. However, its practical implementation is far from trivial due to the presence of the conditional expectation. The suggestion of Ren and Tan (2017) to use regression as in Gobet et al. (2005) relies on a careful basis function choice, which may not always be possible, see the discussion at the end of Section 2.

Neural networks methods for PDEs have been introduced independently in Han et al. (2018) and Sirignano and Spiliopoulos (2018) using backward stochastic differential equations and the Galerkin method respectively, see also Beck et al. (2021), Huré et al. (2019) for other variants of deep learning-based numerical solutions.

A deep neural network algorithm for the numerical solution of PPDEs has also been proposed in Saporito and Zhang (2020) by applying Long Short-Term Memory (LSTM) networks in the framework of the deep Galerkin method for PDEs, see Sirignano and Spiliopoulos (2018). On the other hand, Sabate-Vidales et al. (2020) propose to combine the LSTM network and the path signature to solve the linear PPDE. Unlike regression methods, deep learning algorithms do not rely on the choice of a basis.

In this paper, we propose a deep learning approach to the implementation of the probabilistic scheme of Ren and Tan (2017) for the numerical solution of fully nonlinear PPDEs of the form (1.1). The main idea of Algorithm 4.1 is based on the $L^2$ minimality property of the conditional expectations, which allows us to transform the conditional expectation in the probabilistic scheme into an optimization problem (3.4) that can be solved using neural networks.

Additionally, we propose an error bound of the Algorithm 4.1 in Proposition 4.2, which is used to prove its convergence in Theorem 4.4. In Section 5, we detail the implementation of our algorithm, followed by numerical examples of two-person zero-sum game, Asian and barrier options pricing. In the numerical comparisons, Algorithm 4.1 appears more accurate than the deep learning algorithms of Saporito and Zhang (2020) and Sabate-Vidales et al. (2020).

This paper is organized as follows. The necessary preliminaries on PPDEs are stated in Section 2, and the probabilistic schemes introduced in Fahim et al. (2011) for PDEs and in Ren and Tan (2017) for PPDEs are reviewed in Section 3. In Section 4 we present our main deep learning algorithm, and derive its error bound and convergence respectively in Proposition 4.2 and Theorem 4.4. Numerical implementation and examples are presented in Section 5.

## 2 Viscosity solutions of Path-dependent PDEs

Fix $T > 0$, $x_0 \in \mathbb{R}^d$, let $\Omega = \mathcal{C}_{x_0}([0,T];\mathbb{R}^d)$ denote the set of $\mathbb{R}^d$-valued continuous paths $\omega$ started at $\omega_0 = x_0$ and let $\Theta := [0,T] \times \Omega$. We let $B = (B_t)_{t\in[0,T]}$ denote the $\mathbb{R}^d$-valued canonical process on $\Omega$, while $\mathbb{F} = (\mathcal{F}_t)_{0\le t\le T}$ denotes the canonical filtration generated by $(B_t)_{t\in[0,T]}$, and $\mathbb{P}_0$ is the Wiener measure on $(\Omega, \mathbb{F})$. A natural metric $d$ on $\Theta$ is defined as

$$d\left((t,\omega),(t',\omega')\right) = |t-t'| + \|\omega_{t\wedge\cdot} - \omega'_{t'\wedge\cdot}\|, \qquad (t,\omega),(t',\omega') \in \Theta,$$

where $\|\omega\| := \sup_{t\in[0,T]} \|\omega_t\|_d$, $\omega \in \Omega$, and $\|\cdot\|_d$ denotes the Euclidean norm on $\mathbb{R}^d$. Next, we define $\mathcal{P}$ as the set of probability measures $\mathbb{P}$ such that the canonical process

$$B_t = A_t^{\mathbb{P}} + M_t^{\mathbb{P}} = \int_0^t \mu_s^{\mathbb{P}} ds + M_t^{\mathbb{P}}, \qquad t \in [0,T],$$

is a semimartingale, where $(A^{\mathbb{P}})_{t\in[0,T]}$ is a finite variation process and $(M^{\mathbb{P}})_{t\in[0,T]}$ is a continuous martingale, so that the quadratic variation

$$\langle M^{\mathbb{P}}\rangle_t = \int_0^t a_s^{\mathbb{P}} ds, \qquad t \in [0,T],$$

taking values in the space $\mathbb{S}^d$ of $d \times d$ symmetric matrices is absolutely continuous with respect to the Lebesgue measure on $[0,T]$, and

$$\sup_{t\in[0,T]} \|\mu_t^{\mathbb{P}}\|_d \le L, \qquad \frac{1}{2}\sup_{t\in[0,T]} \mathrm{Tr}\big(a_t^{\mathbb{P}}\big) \le L, \quad \mathbb{P}\text{-a.s.},$$

where $L > 0$ is fixed throughout the paper. We denote by $v_1 \cdot v_2$ the dot product of $v_1, v_2 \in \mathbb{R}^d$, let $\boldsymbol{I}_d$ be the identity matrix in $\mathbb{S}^d$, let $A : B = \mathrm{Tr}(AB)$, and let $\mathbf{1}_d = (1, \ldots, 1)$ represent the all-ones vector in $\mathbb{R}^d$. The following definition makes sense of the classical partial derivatives used in the PPDE (1.1) as path derivatives on $\Theta$, see Ekren et al. (2016).

**Definition 2.1** *We say that $u \in C^{1,2}(\Theta)$ if $(t,\omega) \mapsto u_t(\omega) \in \mathbb{R}$ is continuous on $(\Theta, d)$ and there exists continuous processes $(t,\omega) \mapsto \partial_t u \in \mathbb{R}$, $(t,\omega) \mapsto \partial_\omega u_t \in \mathbb{R}^d$, $(t,\omega) \mapsto \partial_{\omega\omega} u_t \in \mathbb{S}^d$, continuous on $(\Theta, d)$, and such that*

$$du_t = \partial_t u dt + \frac{1}{2}\partial_{\omega\omega} u_t : d\langle B\rangle_t + \partial_\omega u_t \cdot dB_t, \quad \mathbb{P}\text{-}a.s., \textit{ for all } \mathbb{P} \in \mathcal{P}. \tag{2.0}$$

As in e.g. Ren and Tan (2017) and Fahim et al. (2011), we assume that the coefficients of (1.1) satisfy the following conditions throughout the paper.

**Assumption 1** *i) $\sigma(t,\omega) \in \mathbb{S}^d$ is invertible for all $(t,\omega) \in \Theta$, and $b(t,\omega)$, $\sigma(t,\omega)$ satisfy*

$$\sup_{(t,\omega)\neq(t',\omega')} \frac{\|b(t,\omega) - b(t',\omega')\|_d}{|t-t'|^{1/2} + \|\omega_{t\wedge\cdot} - \omega'_{t\wedge\cdot}\|} + \sup_{(t,\omega)\neq(t',\omega')} \frac{\|\sigma(t,\omega) - \sigma(t',\omega')\|_{d\times d}}{|t-t'|^{1/2} + \|\omega_{t\wedge\cdot} - \omega'_{t\wedge\cdot}\|} < \infty,$$

*where $\|\cdot\|_{d\times d}$ denotes the Frobenius norm on $\mathbb{S}^d$.*

*ii) $\omega \mapsto g(\omega)$ is bounded Lipschitz on $\Omega$,*

*iii) $F(t,\omega,u,z,\gamma)$ is continuous in $(t,\omega,u,z,\gamma) \in \Theta \times \mathbb{R} \times \mathbb{R}^d \times \mathbb{S}^d$ and non-decreasing in $\gamma \in \mathbb{S}^d$ for the positive semidefinite order $\le_{\mathrm{psd}}$.*

*iv) $F(t,\omega,u,z,\gamma)$ is Lipschitz with respect to $(\omega,u,z,\gamma) \in \Omega \times \mathbb{R} \times \mathbb{R}^d \times \mathbb{S}^d$ uniformly in $t \in [0,T]$, and $\sup_{(t,\omega)\in\Theta} |F(t,\omega,0,0,0)| < \infty$.*

*v)* *$F(t,\omega,u,z,\gamma)$ is elliptic, i.e., $F(t,\omega,u,z,\gamma) \le F(t,\omega,u,z,\gamma')$ for $\gamma \le \gamma'$, and satisfies $\dfrac{\partial F}{\partial \gamma}(t,\omega,u,z,\gamma) \le_{\rm psd} \boldsymbol{I}_d$ for a.e. $(t,\omega,u,z,\gamma) \in \Theta \times \mathbb{R} \times \mathbb{R}^d \times \mathbb{S}^d$.*

*vi)* *$\dfrac{\partial F}{\partial z}(t,\omega,u,z,\gamma) \in \mathrm{Image}\left(\sigma \dfrac{\partial F}{\partial \gamma}\sigma^\top(t,\omega,u,z,\gamma)\right)$ for all $(t,\omega,u,z,\gamma) \in \Theta \times \mathbb{R} \times \mathbb{R}^d \times \mathbb{S}^d$, and*

$$\text{ess sup}_{(t,\omega,u,z,\gamma)} \left| \left(\frac{\partial F}{\partial z}\right)^\top \left(\sigma \frac{\partial F}{\partial \gamma}\sigma^\top\right)^{-1} \frac{\partial F}{\partial z}(t,\omega,u,z,\gamma)\right| < \infty.$$

In general, $u$ may not be smooth enough to ensure the existence of the classical solution for (1.1), hence we rely on the weaker notion of viscosity solution. For this, we will use the shift operations

$$(\omega \otimes_t \omega')_s := \omega_s \mathbb{1}_{[0,t]}(s) + (\omega'_{s-t} - x_0 + \omega_t)\mathbb{1}_{(t,T]}(s), \quad \omega, \omega' \in \Omega,$$

and, for $u : \Theta \to \mathbb{R}$,

$$u^{t,\omega}(s,\omega') := u(t+s, \omega \otimes_t \omega'), \quad \omega, \omega' \in \Omega.$$

Next, for $u$ in the set $\mathrm{BUC}(\Theta)$ of all bounded and uniformly continuous functions $u : \Theta \to \mathbb{R}$ on $(\Theta, d)$ we define the sets of test functions

$$\overline{\mathcal{A}}u(t,\omega) := \left\{ \varphi \in C^{1,2}(\Theta) : (\varphi - u^{t,\omega})_0 = 0 = \sup_{\tau \in \mathcal{T}_{H_\delta}} \overline{\mathcal{E}}\left[(\varphi - u^{t,\omega})_\tau\right] \right\}$$

and

$$\underline{\mathcal{A}}u(t,\omega) := \left\{ \varphi \in C^{1,2}(\Theta) : (\varphi - u^{t,\omega})_0 = 0 = \inf_{\tau \in \mathcal{T}_{H_\delta}} \underline{\mathcal{E}}\left[(\varphi - u^{t,\omega})_\tau\right] \right\},$$

$(t,\omega) \in \Theta$, where $H_\delta(\omega') := \delta \wedge \inf\{s \ge 0 : |\omega'_s| \ge \delta\}$, $\mathcal{T}_{H_\delta}$ is the set of all $\mathbb{F}$-stopping times taking values in $[0, H_\delta]$, $\overline{\mathcal{E}}[\,\cdot\,] := \sup\limits_{\mathbb{P} \in \mathcal{P}} \mathbb{E}^{\mathbb{P}}[\,\cdot\,]$, and $\underline{\mathcal{E}}[\,\cdot\,] := \inf\limits_{\mathbb{P} \in \mathcal{P}} \mathbb{E}^{\mathbb{P}}[\,\cdot\,]$.
The following definition makes sense of the viscosity solution of the PPDE (1.1).

**Definition 2.2** *i) $u \in BUC(\Theta)$ is a viscosity subsolution (resp. supersolution) of the PPDE (1.1) if for any $(t,\omega) \in \Theta$ and for all $\varphi \in \underline{\mathcal{A}}u(t,\omega)$, (resp. $\varphi \in \bar{\mathcal{A}}u(t,\omega)$), we have*

$$\partial_t \varphi(t,\omega) + b(t,\omega)\cdot\partial_\omega\varphi(t,\omega) + \frac{1}{2}\sigma\sigma^\top(t,\omega) : \partial_{\omega\omega}\varphi(t,\omega) + F\left(\,\cdot\,, \varphi, \sigma^\top \partial_\omega \varphi, \sigma^\top \partial^2_{\omega\omega}\varphi\sigma\right)(t,\omega) \ge 0,$$

*resp.* $\le 0$.

*ii) $u$ is viscosity solution of the PPDE (1.1) if it is both a viscosity subsolution and a viscosity supersolution of (1.1).*

## 3 Probabilistic numerical solution

Next, we consider the probabilistic scheme introduced by Fahim et al. (2011) for PDEs, and later generalized to PPDEs by Ren and Tan (2017). For a given $N \geq 1$ and $h = T/N$, define the random variable

$$X_h^{(t,\omega)} := \begin{pmatrix} x_0 \\ x_0 + b(t,\omega)h + \sigma(t,\omega)B_h \end{pmatrix}, \tag{3.1}$$

where $B_h \sim N(0, h\boldsymbol{I}_d)$ is a $d$-dimensional Gaussian vector. For any vectorized matrix $y = (x_0, x_1, \ldots, x_i)^{\mathrm{V}} \in \mathbb{R}^{(i+1)d}$ with $i \leq N$, we consider the linear interpolation $\overline{y} \in \Omega$ of $y$ defined as

$$\overline{y}_s = \begin{cases} \dfrac{s-kh}{h} x_{k+1} + \left(1 - \dfrac{s-kh}{h}\right) x_k, & s \in [kh, (k+1)h),\ k = 0, 1, \ldots, i-1, \\ x_i, & s \in [ih, T]. \end{cases} \tag{3.2}$$

For $\phi : \Theta \to \mathbb{R}$ a given function, we let

$$\mathcal{D}_h \phi(t,\omega) := \mathbb{E}\left[\phi\big(t+h, \omega \otimes_t \overline{X}_h^{(t,\omega)}\big) H_h(t,\omega) \,\middle|\, \mathcal{F}_t\right],$$

where $H_h = \left(H_0^h, H_1^h, H_2^h\right)$ are the weights defined by

$$H_0^h := 1, \quad H_1^h := \frac{B_h}{h}, \quad H_2^h := \frac{B_h B_h^\top - h\boldsymbol{I}_d}{h^2}.$$

As in (2.5) of Fahim et al. (2011) and (4.11) of Ren and Tan (2017), we let the operator $\mathbb{T}^{t,\omega}$ be defined as

$$\mathbb{T}^{t,\omega}\left[u^h(t+h, \cdot)\right] := \mathbb{E}\left[u^h\big(t+h, \omega \otimes_t \overline{X}_h^{(t,\omega)}\big) \,\middle|\, \mathcal{F}_t\right] + hF\left(\cdot, \mathcal{D}_h u_{t+h}^h\right)(t,\omega). \tag{3.3}$$

The approximation $u^h$ of $u$ is then defined inductively as in (2.4) of Fahim et al. (2011) and § 3 of Ren and Tan (2017) as the linear interpolation $u^h(t,\omega)$ of the sequence

$$\begin{cases} u^h(Nh, \omega) = g(\omega), \\ u^h(ih, \omega) = \mathbb{T}^{t,\omega}\left[u^h((i+1)h, \cdot)\right], \qquad i = 0, 1, \ldots, N-1. \end{cases} \tag{3.4}$$

The convergence of $u^h$ to $u$ as $h$ tends to zero is ensured by the following result, see Theorem 3.9 and Proposition 4.9 in Ren and Tan (2017).

**Theorem 3.1** *Under Assumption 1, assume further that the PPDE (1.1) satisfies the comparison principle for viscosity subsolutions and supersolutions, i.e. if $v$ and $w$ are respectively* *viscosity subsolution and supersolution of PPDE* (1.1) *and* $v(T,\,\cdot\,) \le w(T,\,\cdot\,)$*, then* $v \le w$ *on* $\Theta$*. Then, the PPDE* (1.1) *admits a unique viscosity solution* $u$ *given by the limit*

$$u(t,\omega) = \lim_{h\to 0} u^h(t,\omega), \tag{3.5}$$

*locally uniformly in* $(t,\omega) \in \Theta$*.*

We refer to Theorem 4.2 of Ren et al. (2017) for sufficient conditions on PPDE coefficients for the comparison principle of viscosity solutions to be satisfied, see also Section 5.1. Convergence rates of $O(h^{1/10})$ and $O(h)$ have been derived for (3.5) respectively in Fahim et al. (2011) for PDEs of Hamilton-Jacobi-Bellman type and in Zhang and Zhuo (2014) for PPDEs under smoothness conditions, while the convergence rate of PPDE solutions remains unknown without smoothness conditions.

## 4 Deep learning approximation

In Ren and Tan (2017), the numerical estimation of the conditional expectation in (3.3) has been implemented using regression as in Gobet et al. (2005), which requires to choose a basis for the functional space to be projected on. For example, assuming that

$$F(t,\omega,u,z,\gamma) = \widetilde{F}\left(t,\omega_t,\int_0^t \omega_s ds, u, z, \gamma\right)$$

if the function $g$ in (1.1) takes the form

$$g(\omega) = \widetilde{g}\left(\omega_T, \int_0^T \omega_s ds\right),$$

it can be reasonably guessed that the actual solution $u$ will be of the form

$$u(t,\omega) = \widetilde{u}\left(t,\omega_t,\int_0^t \omega_s ds\right),$$

motivating the choice of basis

$$\left(1, \omega_t, \int_0^t \omega_s ds, \omega_t^2, \left(\int_0^t \omega_s ds\right)^2, \omega_t \int_0^t \omega_s ds\right),$$

using second order polynomials. However, when $g$ is not expressed in such form, e.g. when

$$g(\omega) = \widetilde{g}\left(\omega_T, \sup_{s\in[0,T]} \omega_s\right),$$

it is less clear how the actual solution $u$ will look like, making it difficult to pick an appropriate basis for the projection. We overcome this difficulty, by an alternative deep learning approach, which does not rely on the specific form of the actual solution $u$ and has been previously applied with success to various high-dimensional problems, see e.g. Han et al. (2018), Beck et al. (2021), Huré et al. (2019).

Given $\rho : \mathbb{R} \to \mathbb{R}$ denote an activation function such as $\rho_{\text{ReLU}}(x) := \max(0, x)$, $\rho_{\tanh}(x) := \tanh(x)$, $\rho_{\text{Id}}(x) := x$, we define the set of layer functions $\mathbb{L}^{\rho}_{d_1,d_2}$ by

$$\mathbb{L}^{\rho}_{d_1,d_2} := \big\{ L : \mathbb{R}^{d_1} \to \mathbb{R}^{d_2} \ : \ L(x) = \rho(Wx + b),\ x \in \mathbb{R}^{d_1},\ W \in \mathbb{R}^{d_2 \times d_1},\ b \in \mathbb{R}^{d_2} \big\},$$

where $d_1 \geq 1$ is the input dimension, $d_2 \geq 1$ is the output dimension, and the activation function $\rho$ is applied component-wise to $Wx + b$. Then, we denote by

$$\mathbb{NN}^{\rho,l,m}_{d_0,d_1} := \big\{ L_l \circ \cdots \circ L_0 : \mathbb{R}^{d_0} \to \mathbb{R}^{d_1} \ : \ L_0 \in \mathbb{L}^{\rho}_{d_0,m}, L_l \in \mathbb{L}^{\rho_{\text{Id}}}_{m,d_1}, L_i \in \mathbb{L}^{\rho}_{m,m}, 1 \leq i < l \big\}$$

the set of feed-forward neural networks with one output layer, $l \geq 1$ hidden layers each containing $m \geq 1$ neurons, and the activation functions of the output layer and the hidden layers being respectively the identity function $\rho_{\text{Id}}$ and $\rho$. Any $L_l \circ \cdots \circ L_0 \in \mathbb{NN}^{\rho,l,m}_{d_0,d_1}$ is fully determined by the sequence

$$\theta := \big( W_0, b_0, W_1, b_1, \ldots, W_{l-1}, b_{l-1}, W_l, b_l \big),$$

of $((d_0 + 1)m + (l - 1)(m + 1)m + (m + 1)d_1)$ of parameters, such that

$$L_i(x) = \rho(W_l x + b_l), \qquad i = 0, 1, \ldots, l.$$

Building on (3.1)-(3.2), we let $X^{\pi}$ denote the discretization

$$X^{\pi}_0 = x_0, \qquad X^{\pi}_{i+1} = \begin{pmatrix} X^{\pi}_i \\ X^{(ih,\overline{X}^{\pi}_i)}_h(1) - x_0 + X^{\pi}_i(i) \end{pmatrix}, \quad i = 0, 1, \ldots, N-1, \tag{4.1}$$

where $X^{\pi}_i(k) \in \mathbb{R}^d$ is the $k$-*th* entry of the zero-based array $X^{\pi}_i \in \mathbb{R}^{(i+1)d}$ for $0 \leq k \leq i \leq N$. The similar notation is also used on $X^{(ih,\overline{X}^{\pi}_i)}_h$. Next, we introduce the deep learning scheme for the approximation of (3.4).

**Algorithm 4.1** *i) Fix* $(d, N, l, (m_i)_{0 \leq i < N})$*, the activation function* $\rho$*, and a threshold* $\varepsilon_{\text{thres}} > 0$*, initialize* $\widehat{\mathcal{V}}_N : \mathbb{R}^{(N+1)d} \to \mathbb{R}$ *by* $\widehat{\mathcal{V}}_N(x) = g(\overline{x})$.

*ii) For $i = N-1, \dots, 0$, given $\widehat{\mathcal{V}}_{i+1} : \mathbb{R}^{(i+2)d} \to \mathbb{R}$,*

*a) initialize the neural networks*
$(\mathcal{Y}_i(\cdot\,;\theta), \mathcal{Z}_i(\cdot\,;\theta), \gamma_i(\cdot\,;\theta)) \in \mathbb{NN}^{\rho,l,m_i}_{(i+1)d,1} \times \mathbb{NN}^{\rho,l,m_i}_{(i+1)d,d} \times \mathbb{NN}^{\rho,l,m_i}_{(i+1)d,d(d+1)/2}$,

*b) compute the mean square error function*

$$\begin{aligned} E_i(\theta) \;&:=\; \mathbb{E}\big[\big|\widehat{\mathcal{V}}_{i+1}\big(X^\pi_{i+1}\big)H^h_0 - \mathcal{Y}_i\big(X^\pi_i;\theta\big)\big|^2 + \big\|\widehat{\mathcal{V}}_{i+1}\big(X^\pi_{i+1}\big)H^h_1 - \mathcal{Z}_i\big(X^\pi_i;\theta\big)\big\|^2_d \\ &\quad + \big\|\widehat{\mathcal{V}}_{i+1}\big(X^\pi_{i+1}\big)H^h_2 - \mathrm{Sym}\big(\gamma_i\big(X^\pi_i;\theta\big)\big)\big\|^2_{d\times d}\big], \end{aligned} \tag{4.2}$$

*where for any sequence $(a_1, \dots, a_{d(d+1)/2}) \in \mathbb{R}^{d(d+1)/2}$ we let*

$$\mathrm{Sym}\big((a_1, \dots, a_{d(d+1)/2})^\top\big) = \begin{pmatrix} 2a_{d(d-1)/2+1} & a_{d(d+1)/2-1} & \cdots & a_2 & a_1 \\ a_{d(d+1)/2-1} & \ddots & \ddots & \ddots & a_3 \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ a_2 & \ddots & \ddots & \ddots & a_{d(d-1)/2} \\ a_1 & a_3 & \cdots & a_{d(d-1)/2} & 2a_{d(d+1)/2} \end{pmatrix},$$

*c) choose $\theta^*_i$ in the set*

$$\Big\{\theta = \big(W_0, b_0, W_1, b_1, \dots, W_{l-1}, b_{l-1}, W_l, b_l\big) \;:\; E_i(\theta) < \inf_\theta E_i(\theta) + \varepsilon_{\text{thres}}\Big\}.$$

*iii) Update $\big(\widehat{\mathcal{Y}}_i(\cdot), \widehat{\mathcal{Z}}_i(\cdot), \widehat{\gamma}_i(\cdot)\big) = (\mathcal{Y}_i(\cdot\,;\theta^*_i), \mathcal{Z}_i(\cdot\,;\theta^*_i), \gamma_i(\cdot\,;\theta^*_i))$ and $\widehat{\mathcal{V}}_i : \mathbb{R}^{(i+1)d} \to \mathbb{R}$ by*

$$\widehat{\mathcal{V}}_i(x) := \widehat{\mathcal{Y}}_i(x) + hF\big(ih, \overline{x}, \widehat{\mathcal{Y}}_i(x), \widehat{\mathcal{Z}}_i(x), \mathrm{Sym}\,(\widehat{\gamma}_i(x))\,\big). \tag{4.3}$$

We note that minimizing the error function $E_i(\theta)$ is equivalent to minimizing the quantity

$$\begin{aligned} \varepsilon^{l,m,\theta}_i \;&:=\; \mathbb{E}\big[\big|\mathbb{E}_i\big[\widehat{\mathcal{V}}_{i+1}\left(X^\pi_{i+1}\right)H^h_0\big] - \mathcal{Y}_i\left(X^\pi_i;\theta\right)\big|^2 + \big\|\mathbb{E}_i\big[\widehat{\mathcal{V}}_{i+1}\left(X^\pi_{i+1}\right)H^h_1\big] - \mathcal{Z}_i\left(X^\pi_i;\theta\right)\big\|^2_d \\ &\quad + \big\|\mathbb{E}_i\big[\widehat{\mathcal{V}}_{i+1}\left(X^\pi_{i+1}\right)H^h_2\big] - \mathrm{Sym}\left(\gamma_i\left(X^\pi_i;\theta\right)\right)\big\|^2_{d\times d}\big], \end{aligned} \tag{4.4}$$

from the relationship

$$\begin{aligned} E_i(\theta) = \mathbb{E}\big[&\big|\widehat{\mathcal{V}}_{i+1}\big(X^\pi_{i+1}\big)H^h_0 - \mathbb{E}_i\big[\widehat{\mathcal{V}}_{i+1}\big(X^\pi_{i+1}\big)H^h_0\big] + \mathbb{E}_i\big[\widehat{\mathcal{V}}_{i+1}\big(X^\pi_{i+1}\big)H^h_0\big] - \mathcal{Y}_i\big(X^\pi_i;\theta\big)\big|^2 \\ &+ \big\|\widehat{\mathcal{V}}_{i+1}\big(X^\pi_{i+1}\big)H^h_1 - \mathbb{E}_i\big[\widehat{\mathcal{V}}_{i+1}\big(X^\pi_{i+1}\big)H^h_1\big] + \mathbb{E}_i\big[\widehat{\mathcal{V}}_{i+1}\big(X^\pi_{i+1}\big)H^h_1\big] - \mathcal{Z}_i\big(X^\pi_i;\theta\big)\big\|^2_d \\ &+ \big\|\widehat{\mathcal{V}}_{i+1}\big(X^\pi_{i+1}\big)H^h_2 - \mathbb{E}_i\big[\widehat{\mathcal{V}}_{i+1}\big(X^\pi_{i+1}\big)H^h_2\big] + \mathbb{E}_i\big[\widehat{\mathcal{V}}_{i+1}\big(X^\pi_{i+1}\big)H^h_2\big] - \mathrm{Sym}\big(\gamma_i\big(X^\pi_i;\theta\big)\big)\big\|^2_{d\times d}\big] \\ = \mathbb{E}\big[&\big|\widehat{\mathcal{V}}_{i+1}\left(X^\pi_{i+1}\right)H^h_0 - \mathbb{E}_i\big[\widehat{\mathcal{V}}_{i+1}\left(X^\pi_{i+1}\right)H^h_0\big]\big|^2 + \big\|\widehat{\mathcal{V}}_{i+1}\left(X^\pi_{i+1}\right)H^h_1 - \mathbb{E}_i\big[\widehat{\mathcal{V}}_{i+1}\left(X^\pi_{i+1}\right)H^h_1\big]\big\|^2_d \\ &+ \big\|\widehat{\mathcal{V}}_{i+1}\left(X^\pi_{i+1}\right)H^h_2 - \mathbb{E}_i\big[\widehat{\mathcal{V}}_{i+1}\left(X^\pi_{i+1}\right)H^h_2\big]\big\|^2_{d\times d}\big] + \varepsilon^{l,m,\theta}_i, \end{aligned}$$

where in the second equality, we have used the fact that for any square-integrable $\mathcal{F}_i$-measurable random variable $Y$,

$$\mathbb{E}[(\mathbb{E}_i[X]-Y)(X-\mathbb{E}_i[X])]=\mathbb{E}[(\mathbb{E}_i[X]-Y)\mathbb{E}_i[X-\mathbb{E}_i[X]]]=0.$$

The next result is an error bound of the Algorithm 4.1.

**Proposition 4.2** *Using (3.4) and the notation of Algorithm 4.1, and assuming $F(t,\omega,u,z,\gamma)$ is Lipschitz with respect to $(\omega,u,z,\gamma)\in\Omega\times\mathbb{R}\times\mathbb{R}^d\times\mathbb{S}^d$ uniformly in $t\in[0,T]$ as in part (iv) of Assumption 1, we have*

$$\max_{i=0,\dots,N-1}\mathbb{E}\left[\left|\widehat{\mathcal{V}}_i\left(X_i^\pi\right)-u^h\left(ih,\overline{X}_{ih}^\pi\right)\right|^2\right]\leq M\frac{L^N-1}{L-1}\varepsilon^{l,m},$$

*where we let $L:=32\left(1+K^2h^2+K^2hd+K^2d(d+1)\right)$ and $M:=32\left(1+K^2h^2\right)$, and*

$$\varepsilon^{l,m}:=\sum_{i=0}^{N-1}\varepsilon_i^{l,m,\theta_i^*}. \tag{4.5}$$

*Proof.* Let $\delta_i:=\widehat{\mathcal{V}}_i\left(X_i^\pi\right)-u^h\left(ih,\overline{X}_{ih}^\pi\right)$, $i=0,\dots,N-1$. By (3.4), (4.3), Assumption 1 and the conditional Hölder inequality, we have

$$\begin{aligned}
\mathbb{E}\left[|\delta_i|^2\right] &= \mathbb{E}\Big[\Big|\widehat{\mathcal{Y}}_i\left(X_i^\pi\right)+hF\big(ih,\overline{X}_{ih}^\pi,\widehat{\mathcal{Y}}_i\left(X_i^\pi\right),\widehat{\mathcal{Z}}_i\left(X_i^\pi\right),\mathrm{Sym}\left(\widehat{\gamma}_i\left(X_i^\pi\right)\right)\big)-\mathbb{E}_i\left[u^h\big((i+1)h,\overline{X}_{(i+1)h}^\pi\big)\right] \\
&\quad+hF\big(ih,\overline{X}_{ih}^\pi,\mathbb{E}_i\left[u^h\big((i+1)h,\overline{X}_{(i+1)h}^\pi\big)H_0^h\right],\mathbb{E}_i\left[u^h\big((i+1)h,\overline{X}_{(i+1)h}^\pi\big)H_1^h\right], \\
&\qquad\quad \mathbb{E}_i\left[u^h\big((i+1)h,\overline{X}_{(i+1)h}^\pi H_2^h\big)\right]\big)\Big|^2\Big] \\
&\leq 16\big(1+K^2h^2\big)\mathbb{E}\Big[\Big|\widehat{\mathcal{Y}}_i\left(X_i^\pi\right)-\mathbb{E}_i\left[u^h\big((i+1)h,\overline{X}_{(i+1)h}^\pi\big)H_0^h\right]\Big|^2\Big] \\
&\quad+16K^2h^2\mathbb{E}\Big[\Big\|\widehat{\mathcal{Z}}_i\left(X_i^\pi\right)-\mathbb{E}_i\left[u^h\big((i+1)h,\overline{X}_{(i+1)h}^\pi\big)H_1^h\right]\Big\|_d^2 \\
&\qquad\qquad+\Big\|\mathrm{Sym}\left(\widehat{\gamma}_i\left(X_i^\pi\right)\right)-\mathbb{E}_i\left[u^h\big((i+1)h,\overline{X}_{(i+1)h}^\pi\big)H_2^h\right]\Big\|_{d\times d}^2\Big] \\
&= 16\big(1+K^2h^2\big)\mathbb{E}\Big[\Big|\widehat{\mathcal{Y}}_i\left(X_i^\pi\right)-\mathbb{E}_i\big[\widehat{\mathcal{V}}_{i+1}\big(X_{i+1}^\pi\big)H_0^h\big] \\
&\qquad\qquad+\mathbb{E}_i\big[\widehat{\mathcal{V}}_{i+1}\left(X_{i+1}^\pi\right)H_0^h\big]-\mathbb{E}_i\left[u^h\big((i+1)h,\overline{X}_{(i+1)h}^\pi\big)H_0^h\right]\Big|^2\Big] \\
&\quad+16K^2h^2\mathbb{E}\Big[\Big\|\widehat{\mathcal{Z}}_i\left(X_i^\pi\right)-\mathbb{E}_i\big[\widehat{\mathcal{V}}_{i+1}\left(X_{i+1}^\pi\right)H_1^h\big]+\mathbb{E}_i\big[\widehat{\mathcal{V}}_{i+1}\left(X_{i+1}^\pi\right)H_1^h\big] \\
&\qquad\qquad-\mathbb{E}_i\left[u^h\big((i+1)h,\overline{X}_{(i+1)h}^\pi\big)H_1^h\right]\Big\|_d^2+\Big\|\mathrm{Sym}\left(\widehat{\gamma}_i\left(X_i^\pi\right)\right)-\mathbb{E}_i\big[\widehat{\mathcal{V}}_{i+1}\left(X_{i+1}^\pi\right)H_2^h\big] \\
&\qquad\qquad+\mathbb{E}_i\big[\widehat{\mathcal{V}}_{i+1}\left(X_{i+1}^\pi\right)H_2^h\big]-\mathbb{E}_i\big[u^h\big((i+1)h,\overline{X}_{(i+1)h}^\pi\big)H_2^h\big]\Big\|_{d\times d}^2\Big] \\
&\leq 32\big(1+K^2h^2\big)\varepsilon_i^{l,m,\theta^*}+32\mathbb{E}\left[\mathbb{E}_i\left[\delta_{i+1}\right]\mathbb{E}_i\left[\left(1+K^2h^2\right)\left|H_0^h\right|^2+K^2h^2\left\|H_1^h\right\|_d^2+K^2h^2\left\|H_2^h\right\|_{d\times d}^2\right]\right] \\
&\leq 32\big(1+K^2h^2\big)\varepsilon_i^{l,m,\theta^*}+32\left(1+K^2h^2+K^2hd+K^2d(d+1)\right)\mathbb{E}\left[\delta_{i+1}\right] \\
&= M\varepsilon_i^{l,m,\theta^*}+L\mathbb{E}\left[\delta_{i+1}\right],
\end{aligned}$$

$i = 0, \ldots, N-2$. By backward induction and the fact that $\widehat{\mathcal{V}}_N(x) = u^h(t_N, \overline{x}) = g(\overline{x})$, we obtain

$$\max_{i=0,\ldots,N-1} \mathbb{E}[|\delta_i|^2] \leq \sum_{i=0}^{N-1} L^{i-1} M \varepsilon_i^{l,m,\theta^*} \leq M\varepsilon^{l,m} \sum_{i=0}^{N-1} L^{i-1} = M\frac{L^N - 1}{L-1}\varepsilon^{l,m}.$$

□

The proof of Proposition 4.2 uses only the Lipschitz continuity of $F$ in Assumption 1, while the rest of the conditions in Assumption 1 are required in Theorem 3.1 as in Fahim et al. (2011) and Ren and Tan (2017). Next, we recall the following universal approximation theorem.

**Theorem 4.3** *(Theorem 1 in Hornik (1991)). Fix $l \geq 1$, if the activation function $\rho$ is unbounded and nonconstant, then for any finite measure $\mu$ the set $\bigcup_{m=1}^{\infty} \mathrm{NN}_{d_0,1}^{\rho,l,m}$ is dense in $L^q(\mu)$ for all $q \geq 1$.*

The next corollary shows that the neural network approximation can be made arbitrarily close to the PPDE solution $u\left(0, (x_0)_{s\in[0,T]}\right)$.

**Theorem 4.4** *Under the assumptions of Theorems 3.1 and 4.3, assume additionally that the activation function $\rho$ is Lipschitz. Then, for any $\varepsilon > 0$ there exists $(m_i)_{0\leq i<N}$ and $(\theta_i^*)_{0\leq i<N}$ such that $(\widehat{\mathcal{V}}_i)_{0\leq i\leq N}$ constructed from $(m_i)_{0\leq i<N}$ and $(\theta_i^*)_{0\leq i<N}$ in (4.3) satisfies*

$$\left|u(0, (x_0)_{s\in[0,T]}) - \widehat{\mathcal{V}}_0(x_0)\right| < \varepsilon.$$

*Proof.* Let $\varepsilon > 0$. By Theorem 3.1, we can find $h > 0$ small enough such that

$$\left|u\left(0, (x_0)_{s\in[0,T]}\right) - u^h\left(0, (x_0)_{s\in[0,T]}\right)\right| < \frac{\varepsilon}{2}.$$

First, we note that by Proposition 4.2, the proof is complete by the triangle inequality if we can choose $(m_i)_{0\leq i<N}$ and $(\theta_i^*)_{0\leq i<N}$ such that $\varepsilon^{l,m}$ defined by (4.4) and (4.5) satisfies

$$\begin{aligned}\varepsilon^{l,m} = \sum_{i=0}^{N-1} \mathbb{E}\Big[&\left|\mathbb{E}_i\left[\widehat{\mathcal{V}}_{i+1}\left(X_{i+1}^\pi\right) H_0^h\right] - \mathcal{Y}_i\left(X_i^\pi; \theta_i^*\right)\right|^2 + \left\|\mathbb{E}_i\left[\widehat{\mathcal{V}}_{i+1}\left(X_{i+1}^\pi\right) H_1^h\right] - \mathcal{Z}_i\left(X_i^\pi; \theta_i^*\right)\right\|_d^2 \\ &+ \left\|\mathbb{E}_i\left[\widehat{\mathcal{V}}_{i+1}\left(X_{i+1}^\pi\right) H_2^h\right] - \mathrm{Sym}\left(\gamma_i\left(X_i^\pi; \theta_i^*\right)\right)\right\|_{d\times d}^2\Big] < \frac{L-1}{2M(L^N-1)}\varepsilon, \end{aligned} \tag{4.6}$$

Next, we note that (4.6) holds if we show that

$$\mathbb{E}\left[\left\|\mathbb{E}_i\left[\widehat{\mathcal{V}}_{i+1}\left(X_{i+1}^\pi\right) H_2^h\right] - \mathrm{Sym}\left(\gamma_i\left(X_i^\pi; \theta_i^*\right)\right)\right\|_{d\times d}^2\right] < \frac{L-1}{6NM(L^N-1)}\varepsilon,$$

$i=0,\dots,N-1$, as the argument for the other terms similar. For this, we rely on the universal approximation Theorem 4.3, which requires us to show that the function $\mathbb{E}_i\big[\widehat{\mathcal{V}}_{i+1}\big(X^\pi_{i+1}\big)H^h_2\big]:\mathbb{R}^{(i+1)d}\to\mathbb{S}^d$ is in $L^2(\mu)$, where $\mu$ is the joint distribution of $X^\pi_i$. By the Lipschitz condition on $b(t,\omega)$ and $\sigma(t,\omega)$ in Assumption 1, it is straightforward to show that

$$\mathbb{E}\Big[\max_{0\le k\le i}\|X^\pi_i(k)\|^q_d\Big]<\infty \qquad \text{and} \qquad \mathbb{E}\left[\|\varphi(X^\pi_i)\|^q_k\right]<\infty, \tag{4.7}$$

for any Lipschitz continuous function $\varphi:\mathbb{R}^{(i+1)d}\to\mathbb{R}^k$ and all $q\ge 1$, $0\le i\le N$, see Appendix A. Hence, by the Lipschitz Assumption 1-(ii) and Hölder's inequality, we have $\mathbb{E}_i\big[\widehat{\mathcal{V}}_{i+1}\big(X^\pi_{i+1}\big)H^h_2\big]=\mathbb{E}_i\big[g\big(\overline{X}^\pi_{(i+1)h}\big)H^h_2\big]\in L^2(\mu)$ at the level $i=N-1$. For $0\le i<N$, using Assumption 1-(iv) we have

$$\begin{aligned}
&\mathbb{E}\big[\big\|\mathbb{E}_i\big[\widehat{\mathcal{V}}_{i+1}\big(X^\pi_{i+1}\big)H^h_2\big]\big\|^2_{d\times d}\big]\\
&\le\ \mathbb{E}\big[\|H^h_2\|^2_{d\times d}\times\big|\widehat{\mathcal{Y}}_{i+1}(X^\pi_{i+1})+hF\big((i+1)h,\overline{X}^\pi_{(i+1)h},\widehat{\mathcal{Y}}_{i+1}(X^\pi_{i+1}),\\
&\qquad\qquad\qquad\qquad\widehat{\mathcal{Z}}_{i+1}(X^\pi_{i+1}),\mathrm{Sym}\left(\widehat{\gamma}_{i+1}(X^\pi_{i+1})\right)\big)\big|^2\big]\\
&\le\ 6^2\Big(\mathbb{E}\big[\|H^h_2\|^4_{d\times d}\big]\mathbb{E}\big[|\widehat{\mathcal{Y}}_{i+1}(X^\pi_{i+1})|^4+h^4|F((i+1)h,(0)_{0\le s\le T},0,0,0)|^4\\
&\qquad+h^4K^4\big(\|\overline{X}^\pi_{(i+1)h}\|^4+|\widehat{\mathcal{Y}}_{i+1}(X^\pi_{i+1})|^4+\|\widehat{\mathcal{Z}}_{i+1}(X^\pi_{i+1})\|^4_d+\|\mathrm{Sym}\big(\widehat{\gamma}_{i+1}(X^\pi_{i+1})\big)\|^4_{d\times d}\big)\big]\Big)^{1/2}\\
&<\ \infty,
\end{aligned}$$

which shows that $\mathbb{E}_i\big[\widehat{\mathcal{V}}_{i+1}\big(X^\pi_{i+1}\big)H^h_2\big]\in L^2(\mu)$. In the last inequality we have used (4.7), the fact that $\varphi\in\mathbb{NN}^{\rho,l,m}_{d_0,d_1}$ is Lipschitz when the activation function $\rho$ is Lipschitz, and $\mathbb{E}\left[\|H^h_2\|^4_{d\times d}\right]<\infty$ with $\|\overline{X}^\pi_{ih}\|:=\max_{0\le k\le i}\|X^\pi_i(k)\|_d$. □

Using additionally that $u\in\mathrm{BUC}(\Theta)$ is uniformly continuous, Proposition 4.2 and Theorem 4.4 can be extended from $(0,(x_0)_{s\in[0,T]})$ to any $(t,\omega)\in\Theta$ by changing (4.1) to start from $X^\pi_{kh}=(\omega^\pi_{s\wedge kh})_{s\in[0,T]}$, where $\omega^\pi$ is the linear interpolation of the discretization of $\omega$.

## 5 Numerical examples

The optimization in (4.2) is implemented using Monte Carlo simulation and the Adam gradient descent algorithm, see Kingma and Ba (2014). Precisely, fix the batch size $O$ and the training steps $P$, let $\big(X^{\pi,j}_{(i+1)}\big)_{1\le j\le O}$ be an i.i.d. sample of $(i+1)d$-dimensional random vector with batch size $O$ generated by (4.1), and let

$$L^O_i(\theta):=\frac{1}{O}\sum_{j=1}^{O}\big[\big|\widehat{\mathcal{V}}_{i+1}\big(X^{\pi,j}_{i+1}\big)H^h_0-\mathcal{Y}_i\big(X^{\pi,j}_i;\theta\big)\big|^2+\big\|\widehat{\mathcal{V}}_{i+1}\big(X^{\pi,j}_{i+1}\big)H^h_1-\mathcal{Z}_i\big(X^{\pi,j}_i;\theta\big)\big\|^2_d$$

$$+ \left\| \widehat{\mathcal{V}}_{i+1}\big(X_{i+1}^{\pi,j}\big) H_2^h - \mathrm{Sym}\big(\gamma_i\big(X_i^{\pi,j};\theta\big)\big) \right\|_{d\times d}^2 \Big].$$

Then, we initialize the parameter $\theta_0$ using Xavier initialization, see Glorot and Bengio (2010), and update it using the following rule

$$\begin{cases} v_p &= \beta_1 v_{p-1} + (1-\beta_1)\dfrac{\partial L_i^O}{\partial\theta}(\theta_{p-1}) \\ w_p &= \beta_2 w_{p-1} + (1-\beta_2)\left(\dfrac{\partial L_i^O}{\partial\theta}(\theta_{p-1})\right)^2 \\ \theta_p &= \theta_{p-1} - \eta_p \left(\dfrac{v_p}{1-\beta_1}\right) \Big/ \left(\varepsilon_{\text{Adam}} + \sqrt{\dfrac{w_p}{1-\beta_1}}\right), \end{cases}$$

where $1 \le p \le P$, $(\eta_p)_{1\le p\le P} \in \mathbb{R}^P$ is the learning rate, $(\varepsilon_{\text{Adam}}, \beta_1, \beta_2) \in \mathbb{R}^3$ are the parameters of the Adam algorithm, and $(v_0, w_0)$ is initialized at $(0,0)$. Empirically we have $\theta_P \approx \theta^*$ when $O$ and $P$ are large enough, see e.g. Kingma and Ba (2014). In addition, we use the batch normalization technique, see Ioffe and Szegedy (2015), to stabilize the training process. Define $BN_{\gamma,\beta,\varepsilon_{BN}}$ a transformation over a set of $d_1$-dimensional $\big(x_j^{(i)}\big)_{1\le i\le O, 1\le j\le d_1}$ with batch size $O$ by

$$BN_{\gamma,\beta,\varepsilon_{BN}}\big(x^{(i)}\big) = \left(\beta_j + \gamma_j\big(x_j^{(i)} - \mu_j\big)\Big/\sqrt{\sigma_j^2 + \varepsilon_{BN}}\ \right)_{1\le j\le d_1}, \tag{5.1}$$

where $\gamma, \beta \in \mathbb{R}^{d_1}$, $\varepsilon_{BN} \in \mathbb{R}$, and

$$\mu_j = \frac{1}{O}\sum_{i=1}^{O} x_j^{(i)} \quad \text{and} \quad \sigma_j^2 = \frac{1}{O}\sum_{i=1}^{O}(x_j^{(i)} - \mu_j)^2.$$

Fix $\varepsilon_{BN} \in \mathbb{R}$, a neural network $\varphi(\cdot\ ;\theta) \in \mathbb{NN}_{d_0,d_1}^{\rho,l,m}$ is modified such that each of the layer functions $L_i \in \mathbb{L}_{d_2,d_3}^{\rho}$ is changed to $L_i \in \mathbb{L}_{d_2,d_3}^{\rho,BN}$, where

$$\mathbb{L}_{d_2,d_3}^{\rho,BN} := \Big\{ L : \mathbb{R}^{d_2} \to \mathbb{R}^{d_3}\ :\ L(x) = BN_{\gamma,\beta,\varepsilon_{BN}}\left(\rho(Wx+b)\right), W \in \mathbb{R}^{d_3\times d_2}, b, \gamma, \beta \in \mathbb{R}^{d_3} \Big\},$$

and a transformation from $x \in \mathbb{R}^{d_0}$ to $BN_{\gamma,\beta,\varepsilon_{BN}}(x)$ is added before passing to the first layer. Then, the neural network parameter $\theta$ is changed to

$$\theta^{BN} = \big(\gamma_{-1}, \beta_{-1} W_0, b_0, \gamma_0, \beta_0, W_1, b_1, \gamma_1, \beta_1, \dots, W_{l-1}, b_{l-1}, \gamma_{l-1}, \beta_{l-1}, W_l, b_l, \gamma_l, \beta_l\big).$$

In the following subsections, we provide three examples of implementation of the numerical scheme of Proposition 4.2. In our numerical examples, the activation function $\rho = \rho_{\text{ReLU}}$, the Adam parameters $(\beta_1, \beta_2, \varepsilon_{\text{Adam}}) = (0.9, 0.999, 10^{-8})$, the batch normalization's parameter

$\varepsilon_{BN} = 10^{-6}$, and the learning rate

$$\eta_p = \begin{cases} 10^{-1}, & 1 \le p < 2P/3, \\ 10^{-2}, & 2P/3 \le p < 5P/6, \\ 10^{-3}, & 5P/6 \le p < P, \end{cases}$$

are fixed. We note that in the case of $d = 100$, the numerical simulations of Saporito and Zhang (2020) and Sabate-Vidales et al. (2020) are not presented because they require more than the 12 GB RAM provided by Google Colab.

As in Alanko and Avellaneda (2013), for better convergence we implement the modification

$$\begin{cases} Y_h\phi(t,\omega) := \mathbb{E}\left[\phi\left(t+h, \omega\otimes_t \overline{X}_h^{(t,\omega)}\right) H_0^h \,\middle|\, \mathcal{F}_t\right], \\ Z_h\phi(t,\omega) := \mathbb{E}\left[\left(\phi\left(t+h, \omega\otimes_t \overline{X}_h^{(t,\omega)}\right) - Y_h\phi(t,\omega)\right) H_1^h \,\middle|\, \mathcal{F}_t\right], \\ \Gamma_h\phi(t,\omega) := \mathbb{E}\left[\left(\phi\left(t+h, \omega\otimes_t \overline{X}_h^{(t,\omega)}\right) - Y_h\phi(t,\omega) - Z_h\phi(t,\omega)\cdot W_h\right) H_2^h \,\middle|\, \mathcal{F}_t\right], \end{cases}$$

of (3.4) where the operator $\mathbb{T}^{t,\omega}$ in (3.3) is replaced by

$$\mathbb{T}^{t,\omega}\left[u^h(t+h,\,\cdot\,)\right] = Y_h\phi(t,\omega) + hF\left(\,\cdot\,, Y_h\phi, Z_h\phi, \Gamma_h\phi\right)(t,\omega), \tag{5.2}$$

and the error function $E_i(\theta)$ in (4.2) is replaced with

$$\begin{aligned} E_i(\theta) = \mathbb{E}\big[\big|\widehat{\mathcal{V}}_{i+1}(X_{i+1}^\pi)H_0^h - \mathcal{Y}_i(X_i^\pi;\theta)\big|^2 + \big\|\big(\widehat{\mathcal{V}}_{i+1}(X_{i+1}^\pi) - \mathcal{Y}_i(X_i^\pi;\theta)\big)H_1^h - \mathcal{Z}_i(X_i^\pi;\theta)\big\|_d^2 \\ + \big\|\big(\widehat{\mathcal{V}}_{i+1}(X_{i+1}^\pi) - \mathcal{Y}_i(X_i^\pi;\theta) - \mathcal{Z}_i(X_i^\pi;\theta)\cdot W_h\big)H_2^h - \gamma_i(X_i^\pi;\theta)\big\|_{d\times d}^2\big]. \end{aligned} \tag{5.3}$$

Although the following examples do not satisfy all conditions stated in Assumption 1, we will use them as in e.g. Ren and Tan (2017) to assess the performance of Algorithm 4.1. In the sequel, we let $(B_t)_{0\le t\le T} = (B_t^1,\ldots,B_t^d)_{0\le t\le T}$ denote a $d$-dimensional Brownian motion.

## 5.1 Path-dependent two-person zero-sum game

In this section we consider the higher dimensional extension

$$u(0,x_0) = \inf_{\mu_t\in[\underline{\mu},\overline{\mu}]}\ \sup_{a_t\in[\underline{a},\overline{a}]} \mathbb{E}\left[g\left(X_T^{\mu,a}, \int_0^T X_s^{\mu,a}ds\right) + \int_0^T f\left(t, X_t^{\mu,a}, \int_0^t X_s^{\mu,a}ds\right)dt\right],$$

of the path-dependent two-person zero-sum game in (5.1) of Ren and Tan (2017), where $(X_t)_{0\le t\le T} = (X_t^1,\ldots,X_t^d)_{0\le t\le T}$ follows the SDE

$$\begin{cases} dX_t^{\mu,a} = \mu_t\mathbf{1}_d dt + \sqrt{a_t}\ \boldsymbol{I}_d dB_t, \\ X_0 = x_0, \end{cases} \tag{5.4}$$

where $x_0 \in \mathbb{R}^d$, $\underline{\mu}$, $\overline{\mu}$, $\underline{a}$, $\overline{a} \in \mathbb{R}$. The solution of this control problem is the solution of the following PPDE $u : [0,T] \times C([0,T];\mathbb{R}^d) \to \mathbb{R}$ evaluated at $(0, x_0)$

$$\partial_t u + \min_{\mu \in [\underline{\mu},\overline{\mu}]} \mu\big(\mathbf{1}_d \cdot \partial_\omega u\big) + \frac{1}{2} \max_{a \in [\underline{a},\overline{a}]} \big(a \,\mathrm{tr}\,(\partial^2_{\omega\omega} u)\big) + f\left(t, \omega_t, \int_0^t \omega_s ds\right) = 0, \quad u(T,\omega) = g\left(\omega_T, \int_0^T \omega_s ds\right),$$

and for the purpose of our deep algorithm we rewrite the above PPDE as

$$\partial_t u + \frac{\underline{a}}{2} \mathrm{tr}\,(\partial^2_{\omega\omega} u) + \min_{\mu \in [\underline{\mu},\overline{\mu}]} \mu\,(\mathbf{1}_d \cdot \partial_\omega u) + \frac{1}{2} \max_{a \in [\underline{a},\overline{a}]} a \,\mathrm{tr}\,(\partial^2_{\omega\omega} u) + f\left(t, \omega_t, \int_0^t \omega_s ds\right) - \frac{\underline{a}}{2} \mathrm{tr}\,(\partial^2_{\omega\omega} u) = 0, \tag{5.5}$$

with

$$F(t,\omega,u,z,\gamma) = \frac{1}{\sqrt{\underline{a}}} \min_{\mu \in [\underline{\mu},\overline{\mu}]} \mu\,(\mathbf{1}_d \cdot z) + \frac{1}{2\underline{a}} \max_{a \in [\underline{a},\overline{a}]} (a \,\mathrm{tr}\,\gamma) + f\left(t, \omega_t, \int_0^t \omega_s ds\right) - \frac{1}{2} \mathrm{tr}\,\gamma.$$

Denoting by $x = (x^1, \ldots, x^d)$ and $y = (y^1, \ldots, y^d)$ we put $g(x,y) = \cos\left(\frac{1}{d}\sum_{i=1}^d (x^i + y^i)\right)$ and

$$f(t,x,y) = \left(\frac{1}{d}\sum_{i=1}^d x^i + \overline{\mu}\right)\left(\sin\left(\frac{1}{d}\sum_{i=1}^d (x^i+y^i)\right)\right)^+ - \left(\frac{1}{d}\sum_{i=1}^d x^i + \underline{\mu}\right)\left(\sin\left(\frac{1}{d}\sum_{i=1}^d (x^i+y^i)\right)\right)^-$$
$$+ \frac{\underline{a}}{2d}\left(\cos\left(\frac{1}{d}\sum_{i=1}^d (x^i+y^i)\right)\right)^+ - \frac{\overline{a}}{2d}\left(\cos\left(\frac{1}{d}\sum_{i=1}^d (x^i+y^i)\right)\right)^-.$$

Although this choice of $f(t,x,y)$ does not satisfy part (iv) of Assumption 1, it makes the PPDE (5.5) explicitly solvable as $u(t,\omega) = \cos\left(\frac{1}{d}\sum_{i=1}^d \left(\omega^i_t + \int_0^t \omega^i_s ds\right)\right)$, which can be used to evaluate the precision of Algorithm 4.1. Source codes are available on request.

| Method | $d$ | Regr./Deep | Mean | Stdev | Ref. value | Rel. $L^1$-error | Runtime (s) |
|---|---|---|---|---|---|---|---|
| Deep PPDE using (5.3) | 1 | Deep | 1.000805 | 9.61E-05 | 1.0 | 8.05E-04 | 62 |
| Deep PPDE using (4.2) | 1 | Deep | 0.999331 | 1.52E-03 | 1.0 | 1.37E-03 | 61 |
| Saporito and Zhang (2020) | 1 | Deep | 1.000852 | 1.12E-02 | 1.0 | 9.42E-03 | 26 |
| Ren and Tan (2017) using (5.2) | 1 | Regr. | 0.9999463 | 4.72E-05 | 1.0 | 5.47E-05 | 1 |
| Ren and Tan (2017) using (3.4) | 1 | Regr. | 1.075509 | 2.59E-02 | 1.0 | 7.55E-02 | 1 |
| Deep PPDE using (5.3) | 10 | Deep | 1.000914 | 2.19E-04 | 1.0 | 9.14E-04 | 63 |
| Deep PPDE using (4.2) | 10 | Deep | 0.9939934 | 2.99E-03 | 1.0 | 6.01E-03 | 62 |
| Saporito and Zhang (2020) | 10 | Deep | 0.9537241 | 1.63E-01 | 1.0 | 1.06E-01 | 517 |
| Ren and Tan (2017) using (5.2) | 10 | Regr. | 1.000166 | 6.00E-06 | 1.0 | 1.66E-04 | 2 |
| Ren and Tan (2017) using (3.4) | 10 | Regr. | 2.348812 | 4.31E-01 | 1.0 | Diverges | 2 |
| Deep PPDE using (5.3) | 100 | Deep | 1.002474 | 5.01E-04 | 1.0 | 2.47E-03 | 83 |
| Deep PPDE using (4.2) | 100 | Deep | 0.970783 | 1.89E-02 | 1.0 | 3.01E-02 | 81 |
| Ren and Tan (2017) using (5.2) | 100 | Regr. | 26.12039 | 7.36E+00 | 1.0 | Diverges | 104 |
| Ren and Tan (2017) using (3.4) | 100 | Regr. | 328.2853 | 8.81E+01 | 1.0 | Diverges | 102 |

Table 1: Comparison between *i)* PPDE with training parameters $m = d + 10$, $l = 2$, $O = 256$, $h = 0.01$, and $P = 900$; *ii)* Ren and Tan (2017) with $O = 10000$ and $h = 0.01$; *iii)* Saporito and Zhang (2020) with training parameters $m = d + 10$, $l = 2$, $O = 256$, $h = 0.01$, and $P = 1000$.

In Table 1, our PPDE algorithm is compared to Ren and Tan (2017) and Saporito and Zhang (2020) with ten Monte Carlo runs, with $\underline{\mu} = -0.2$, $\overline{\mu} = 0.2$, $\underline{a} = 0.04$, $\overline{a} = 0.09$, $T = 0.1$, $x_0 = (0, \ldots, 0)$, and runtimes are measured in seconds.

## 5.2 Asian options

The second example is the following pricing problem of Asian basket call option:

$$u(0, x_0) = \mathbb{E}\left[e^{-r_0 T}\left(\frac{1}{Td}\sum_{i=1}^{d}\int_0^T X_s^i ds - K\right)^+\right],$$

with strike price $K \in \mathbb{R}$, where $(X_t)_{0\le t\le T} = (X_t^1, \ldots, X_t^d)_{0\le t\le T}$ is a $d$-dimensional asset price process following the geometric Brownian motions

$$X_t^i = X_0^i \mathrm{e}^{\sigma_i B_t^i + r_i - \sigma_i^2 t/2}, \qquad t \in \mathbb{R}_+, \quad i = 1, \ldots, d, \tag{5.6}$$

where $x_0 \in \mathbb{R}^d$, and $r_1, \ldots, r_d, \sigma_1, \ldots, \sigma_d \in \mathbb{R}$. The solution of this pricing problem is given by evaluating at $(t, x) = (0, x_0)$ the solution $u : [0, T] \times C\left([0, T]; \mathbb{R}^d\right) \to \mathbb{R}$ of the following PPDE:

$$\partial_t u + r(\omega_t) \cdot \partial_\omega u + \frac{1}{2}\left(\sigma\sigma^\top(\omega_t) : \partial^2_{\omega\omega} u\right) - r_0 u = 0, \quad u(T, \omega) = \left(\frac{1}{Td}\sum_{i=1}^{d}\int_0^T \omega_s^i ds - K\right)^+,$$

where $r(\omega_t) = \left(r_1\omega_t^1, \ldots, r_d\omega_t^d\right)$ and $\sigma(\omega_t) = \mathrm{Diag}\left(\sigma_1\omega_t^1, \ldots, \sigma_d\omega_t^d\right)$. Here, $F(t, \omega, u, z, \gamma) = -r_0 u$ does not depend on $z$ and $\gamma$, therefore the neural networks $\mathcal{Z}_i(\cdot\,; \theta)$ and $\gamma_i(\cdot\,; \theta)$ are not needed, which improves the efficiency of Algorithm 4.1.

When $d = 1$ we compare our deep PPDE algorithm with other deep PDE algorithms such as Han et al. (2018) and Beck et al. (2021). For this, we write

$$u(t, (X_s)_{s\in[0,t]}) = g\left(t, \frac{1}{X_t}\left(\frac{1}{T}\int_0^t X_u dt - K\right)\right),$$

where $g : [0, T] \times \mathbb{R} \to \mathbb{R}$ is the solution of the Rogers and Shi (1995) PDE

$$\partial_t g + (1/T - rz)\frac{\partial g}{\partial x} + \frac{1}{2}\sigma^2 z^2 \partial^2_{zz} g = 0, \quad g(T, z) = z^+, \tag{5.7}$$

see Proposition 10.7 in Privault (2014).

We use Monte Carlo simulations with $O = 1{,}000{,}000$ and $h = 0.01$ as the reference solution. To compare our PPDE algorithm with Ren and Tan (2017), Saporito and Zhang

(2020), Sabate-Vidales et al. (2020), Han et al. (2018), and Beck et al. (2021) under the setting of $r_0 = r_1 = \cdots = r_d = 0.01$, $\sigma_1 = \cdots = \sigma_d = 0.1$, $K = 0.7$, $T = 0.1$, $x_0 = (1, \ldots, 1)$. The statistics of 10 independent runs are summarized in Table 2.

| Method | $d$ | Regr./Deep | Mean | Stdev | Ref. value | Rel. $L^1$-error | Runtime (s) |
|---|---|---|---|---|---|---|---|
| Han et al. (2018) | 1 | Deep | 0.3002467 | 2.31E-06 | 0.3002021 | 1.49E-04 | 32 |
| Beck et al. (2021) | 1 | Deep | 0.3002827 | 4.21E-04 | 0.3002021 | 1.12E-03 | 20 |
| Sabate-Vidales et al. (2020) | 1 | Deep | 0.3002722 | 1.24E-03 | 0.3002021 | 3.51E-03 | 10 |
| Deep PPDE | 1 | Deep | 0.3008159 | 1.29E-03 | 0.3002021 | 3.83E-03 | 31 |
| Saporito and Zhang (2020) | 1 | Deep | 0.3002544 | 2.43E-03 | 0.3002021 | 6.01E-03 | 25 |
| Ren and Tan (2017) | 1 | Regr. | 0.3002768 | 2.23E-04 | 0.3002021 | 4.77E-04 | 1 |
| Deep PPDE | 10 | Deep | 0.3010345 | 4.08E-04 | 0.3002024 | 2.77E-03 | 31 |
| Sabate-Vidales et al. (2020) | 10 | Deep | 0.3002251 | 2.11E-03 | 0.3002024 | 5.80E-03 | 411 |
| Saporito and Zhang (2020) | 10 | Deep | 0.304033 | 1.05E-02 | 0.3002024 | 2.97E-02 | 522 |
| Ren and Tan (2017) | 10 | Regr. | 0.3002137 | 6.07E-05 | 0.3002024 | 1.70E-04 | 2 |
| Deep PPDE | 100 | Deep | 0.3006346 | 1.28E-04 | 0.3001993 | 1.45E-03 | 35 |
| Ren and Tan (2017) | 100 | Regr. | 0.3001923 | 2.65E-05 | 0.3001993 | 7.50E-05 | 23 |

Table 2: Comparison between *i*) PPDE with training parameters $m = d + 10$, $l = 2$, $O = 256$, $h = 0.01$, and $P = 900$; *ii*) Ren and Tan (2017) with $O = 10000$ and $h = 0.01$; *iii*) Saporito and Zhang (2020) with training parameters $m = d + 10$, $l = 2$, $O = 256$, $h = 0.01$, and iterations 1000; *iv*) Sabate-Vidales et al. (2020) with training parameters $m = d + 10$, $l = 2$, $O = 256$, $h = 0.01$, and $P = 600$; *v*) Han et al. (2018) with training parameters $m = d + 10$, $l = 2$, $O = 64$, $h = 0.01$, and $P = 4000$; *vi*) Beck et al. (2021) with training parameters $m = d + 10$, $l = 2$, $O = 256$, $h = 0.01$, and $P = 600$.

## 5.3 Barrier options

The third example is the following pricing problem of barrier basket call option:

$$u(0, x_0) = \mathbb{E}\left[ e^{-r_0 T} \mathbb{1}_{\left\{ \max_{0\le s\le T} \left( \frac{1}{d}\sum_{i=1}^d X_s^i \right) < B \right\}} \left( \frac{1}{d}\sum_{i=1}^d X_T^i - K \right)^+ \right],$$

where the strike price $K \in \mathbb{R}$, the barrier $B \in \mathbb{R}$, and $(X_t)_{0\le t\le T} = (X_t^1, \ldots, X_t^d)_{0\le t\le T}$ is a $d$-dimensional stock processes that follows the geometric Brownian motions (5.6) The solution of this pricing problem is given by evaluating at $(t, x) = (0, x_0)$ the solution $u : [0, T] \times C\left([0, T]; \mathbb{R}^d\right) \to \mathbb{R}$ of the following PPDE:

$$\partial_t u + r(\omega_t) \cdot \partial_\omega u + \frac{1}{2}\left( \sigma\sigma^\top(\omega_t) : \partial^2_{\omega\omega} u \right) - r_0 u = 0,$$
$$u(T, \omega) = \mathbb{1}_{\left\{ \max_{0\le s\le T} \left( \frac{1}{d}\sum_{i=1}^d \omega_s^i \right) < B \right\}} \left( \frac{1}{d}\sum_{i=1}^d \omega_T^i - K \right)^+.$$

As in Section 5.2, $F(t, \omega, u, z, \gamma) = -r_0 u$ does not depend on $z$ and $\gamma$ and the neural networks $\mathcal{Z}_i(\cdot\,;\theta)$, and $\gamma_i(\cdot\,;\theta)$ are not needed.

We use Monte Carlo simulations with $O = 1000000$ and $h = 0.01$ as the reference solution to compare our PPDE algorithm with Ren and Tan (2017), Saporito and Zhang (2020), and Sabate-Vidales et al. (2020) under the setting of $r_0 = r_1 = \cdots = r_d = 0.01$, $\sigma_1 = \cdots = \sigma_d = 0.1$, $K = 0.7$, $B = 1.2$, $T = 0.1$, $x_0 = (1, \dots, 1)$. The statistics of 10 independent runs are summarized in Table 3.

| Method | $d$ | Regr./Deep | Mean | Stdev | Ref. value | Rel. $L^1$-error | Runtime (s) |
|---|---|---|---|---|---|---|---|
| Sabate-Vidales et al. (2020) | 1 | Deep | 0.3009402 | 2.18E-03 | 0.3007008 | 5.75E-03 | 8 |
| Deep PPDE | 1 | Deep | 0.3019161 | 1.97E-03 | 0.3007008 | 5.92E-03 | 31 |
| Saporito and Zhang (2020) | 1 | Deep | 0.3019159 | 2.24E-03 | 0.3007008 | 6.98E-03 | 26 |
| Ren and Tan (2017) | 1 | Regr. | 0.3006738 | 2.81E-04 | 0.3007008 | 7.72E-04 | 1 |
| Deep PPDE | 10 | Deep | 0.3017532 | 5.44E-04 | 0.3006973 | 3.51E-03 | 31 |
| Sabate-Vidales et al. (2020) | 10 | Deep | 0.301107 | 3.15E-03 | 0.3006973 | 7.35E-03 | 225 |
| Saporito and Zhang (2020) | 10 | Deep | 0.3030515 | 1.03E-02 | 0.3006973 | 2.71E-02 | 519 |
| Ren and Tan (2017) | 10 | Regr. | 0.3007255 | 1.34E-04 | 0.3006973 | 3.56E-04 | 2 |
| Deep PPDE | 100 | Deep | 0.3016375 | 1.98E-04 | 0.3007003 | 3.12E-03 | 35 |
| Ren and Tan (2017) | 100 | Regr. | 0.3035602 | 3.74E-03 | 0.3007003 | 1.05E-02 | 23 |

Table 3: Comparison between *i)* PPDE with training parameters $m = d + 10$, $l = 2$, $O = 256$, $h = 0.01$, and $P = 900$; *ii)* Ren and Tan (2017) with $O = 10000$ and $h = 0.01$; *iii)* Saporito and Zhang (2020) with training parameters $m = d + 10$, $l = 2$, $O = 256$, $h = 0.01$, and $P = 1000$; *iv)* Sabate-Vidales et al. (2020) with training parameters $m = d + 10$, $l = 2$, $O = 256$, $h = 0.01$, and $P = 600$.

## A Appendix

Proof of (4.7). We first show the finiteness of the first term in (4.7) by induction, where $i = 0$ obviously holds. Assume that it holds at level $i$, by Assumption 1, Hölder's inequality, the independence between $B_h$ and $\overline{X}^\pi_{ih}$, and (4.1), we have

$$\begin{aligned}
&\mathbb{E}\Big[\max_{0\le k\le i+1}\|X^\pi_{i+1}(k)\|^q_d\Big] \le \mathbb{E}\Big[\max\Big(\max_{0\le k\le i}\|X^\pi_i(k)\|^q_d, \|X^\pi_{i+1}((i+1)h)\|^q_d\Big)\Big] \\
&= \mathbb{E}\Big[\max\Big(\max_{0\le k\le i}\|X^\pi_i(k)\|^q_d,\ \|X^\pi_i(i) + b\big(ih, \overline{X}^\pi_{ih}\big)h + \sigma\big(ih, \overline{X}^\pi_{ih}\big)B_h\|^q_d\Big)\Big] \\
&\le \mathbb{E}\Big[\max_{0\le k\le i}\|X^\pi_i(k)\|^q_d\Big] + \mathbb{E}\left[\|X^\pi_i(i) + b\big(ih, \overline{X}^\pi_{ih}\big)h + \sigma\big(ih, \overline{X}^\pi_{ih}\big)B_h\|^q_d\right] \\
&\le \mathbb{E}\Big[\max_{0\le k\le i}\|X^\pi_i(k)\|^q_d\Big] + 3^q\mathbb{E}\Big[\|X^\pi_i(i)\|^q_d + h^q\big(K\big(|ih|^{1/2} + \|\overline{X}^\pi_{ih}\|\big) + \|b(0,(0)_{0\le s\le T})\|_d\big)^q \\
&\qquad + \|B_h\|^q_d\big(K\big(|ih|^{1/2} + \|\overline{X}^\pi_{ih}\|\big) + \|\sigma(0,(0)_{0\le s\le T})\|_{d\times d}\big)^q\Big] \\
&\le \mathbb{E}\Big[\max_{0\le k\le i}\|X^\pi_i(k)\|^q_d\Big] + 9^q\mathbb{E}\left[\|X^\pi_i(i)\|^q_d\right] + h^q\left(K^q\left(T^{q/2} + \mathbb{E}\left[\|\overline{X}^\pi_{ih}\|^q\right]\right) + \|b(0,(0)_{0\le s\le T})\|^q_d\right) \\
&\qquad + \mathbb{E}\left[\|B_h\|^q_d\right]\left(K^q\left(T^{q/2} + \mathbb{E}\left[\|\overline{X}^\pi_{ih}\|^q\right]\right) + \|\sigma(0,(0)_{0\le s\le T})\|^q_{d\times d}\right)
\end{aligned}$$

$$\leq \quad C_0 + C_1 \mathbb{E}\Big[\max_{0\leq k\leq i} \|X_i^\pi(k)\|_d^q\Big] < \infty,$$

where $C_0 \geq 0$ and $C_1 \geq 1$. In the second last inequality we used the fact that $\|\overline{X}_{ih}^\pi\|^q = \max\limits_{0\leq k\leq i}\|X_i^\pi(k)\|_d^q$, and the centered Gaussian random variable $B_h$ has finite $\mathbb{E}\left[|B_h|^q\right]$ for any choice of $q$. The finiteness of the second term in (4.7) is obvious since

$$\mathbb{E}\left[\|\varphi(X_i^\pi)\|_k^q\right] \leq K^q \mathbb{E}\big[\|\overline{X}_{ih}^\pi\|_{(i+1)d}^q\big] + \|\varphi(0,(0)_{0\leq s\leq T})\|_k^q < \infty.$$

## References

[1] S. Alanko and M. Avellaneda. Reducing variance in the numerical solution of BSDEs. *C. R. Math. Acad. Sci. Paris*, 351(3-4):135–138, 2013.

[2] C. Beck, S. Becker, P. Cheridito, A. Jentzen, and A. Neufeld. Deep splitting method for parabolic PDEs. *SIAM J. Sci. Comput.*, 43(5):A3135–A3154, 2021.

[3] B. Dupire. Functional Itô calculus. *Preprint SSRN*, 2009.

[4] I. Ekren, C. Keller, N. Touzi, and J. Zhang. On viscosity solutions of path dependent PDEs. *Ann. Probab.*, 42(1):204–236, 2014.

[5] I. Ekren, N. Touzi, and J. Zhang. Viscosity solutions of fully nonlinear parabolic path dependent PDEs: Part I. *Ann. Probab.*, 44(2):1212–1253, 2016.

[6] I. Ekren, N. Touzi, and J. Zhang. Viscosity solutions of fully nonlinear parabolic path dependent PDEs: Part II. *Ann. Probab.*, 44(4):2507–2553, 2016.

[7] A. Fahim, N. Touzi, and X. Warin. A probabilistic numerical method for fully nonlinear parabolic PDEs. *Ann. Appl. Probab.*, 21(4):1322–1364, 2011.

[8] X. Glorot and Y. Bengio. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pages 249–256, 2010.

[9] E. Gobet, J.P. Lemor, and X. Warin. A regression-based Monte Carlo method to solve backward stochastic differential equations. *Ann. Appl. Probab.*, 15(3):2172–2202, 2005.

[10] J. Han, A. Jentzen, and W. E. Solving high-dimensional partial differential equations using deep learning. *Proc. Natl. Acad. Sci. USA*, 115(34):8505–8510, 2018.

[11] K. Hornik. Approximation capabilities of multilayer feedforward networks. *Neural Netw.*, 4(2):251–257, 1991.

[12] C. Huré, H. Pham, and X. Warin. Some machine learning schemes for high-dimensional nonlinear PDEs. *arXiv preprint arXiv:1902.01599*, 2019.

[13] S. Ioffe and C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*, 2015.

[14] A. Jacquier and M. Oumgari. Deep curve-dependent PDEs for affine rough volatility. *arXiv preprint arXiv:1906.02551*, 2019.

[15] D.P. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[16] S. Peng. Note on viscosity solution of path-dependent PDE and G-martingales. *arXiv preprint arXiv:1106.1144*, 2011.

[17] N. Privault. *Stochastic finance: An introduction with market examples.* Financial Mathematics Series. Chapman & Hall/CRC, 2014.

[18] Z. Ren and X. Tan. On the convergence of monotone schemes for path-dependent PDEs. *Stochastic Process. Appl.*, 127(6):1738–1762, 2017.

[19] Z. Ren, N. Touzi, and J. Zhang. Comparison of viscosity solutions of fully nonlinear degenerate parabolic path-dependent PDEs. *SIAM J. Math. Anal.*, 49(5):4093–4116, 2017.

[20] L.C.G. Rogers and Z. Shi. The value of an Asian option. *J. Appl. Probab.*, 32(4):1077–1088, 1995.

[21] M. Sabate-Vidales, D. Šiška, and L. Szpruch. Solving path dependent PDEs with LSTM networks and path signatures. *arXiv preprint arXiv:2011.10630*, 2020.

[22] Y.F. Saporito and Z. Zhang. PDGM: A neural network approach to solve path-dependent partial differential equations. *arXiv preprint arXiv:2003.02035*, 2020.

[23] J. Sirignano and K. Spiliopoulos. DGM: A deep learning algorithm for solving partial differential equations. *J. Comput. Phys.*, 375:1339–1364, 2018.

[24] S. Tang and F. Zhang. Path-dependent optimal stochastic control and viscosity solution of associated Bellman equations. *Discrete Contin. Dyn. Syst.*, 35(11):5521–5553, 2015. ISSN 1078-0947.

[25] F. Viens and J. Zhang. A martingale approach for fractional Brownian motions and related path dependent PDEs. *Ann. Appl. Probab.*, 29(6):3489–3540, 2019.

[26] J. Zhang and J. Zhuo. Monotone schemes for fully nonlinear parabolic path dependent PDEs. *Int. J. Financ. Eng.*, 1(01):1450005, 2014.