XNet-Enhanced Deep BSDE Method and Numerical Analysis

Xiaotao Zheng*[1], Xingye Yue†[1], Zhihong Xia‡[2,3], and Xin Li§[2]

[1]Center for Financial Engineering, Soochow University, Suzhou 215008, Jiangsu, China
[2]Institute of Advanced Research, Great Bay University, Dongguan 523808, Guangdong, China
[3]Department of Mathematics, Northwestern University, Evanston 60208, IL, USA

Abstract

Semilinear parabolic partial differential equations (PDEs) are fundamental to modeling complex dynamical systems across scientific domains. The Deep Backward Stochastic Differential Equation (BSDE) method is a promising approach for high-dimensional PDEs; however, existing convergence results apply only to globally Lipschitz generators, excluding important cases such as Allen–Cahn and Hamilton–Jacobi–Bellman (HJB) equations.

This paper presents both a theoretical and a computational advance for Deep BSDE methods. Theoretically, we establish the convergence theory for non–Lipschitz generators–covering Allen–Cahn equations with cubic nonlinearity and HJB equations with quadratic gradient growth–based on a bounded double–well lemma and a truncated-BSDE analysis within the Bouchard–Touzi–Zhang theory. Computationally, we instantiate the framework with XNet, a shallow architecture with $\mathcal{O}(L)$ parameters that preserves strong approximation while substantially reducing optimization and computational cost. Numerical experiments on 100–dimensional PDEs corroborate the predicted convergence behavior and demonstrate significant efficiency gains over standard feedforward implementations.
Keyword: Deep BSDE method, XNet, high–dimensional PDEs, neural networks, approximation errors.

1 Introduction

Semilinear parabolic partial differential equations (PDEs) play a crucial role in modeling complex dynamic systems across various scientific domains, from financial mathematics to biological processes. Consider the general form:

$$\begin{cases} \partial_t u(t,x) + \frac{1}{2}\,\mathrm{Tr}\big(\sigma\sigma^T(t,x)\,D_x^2 u(t,x)\big) + \mu(t,x)\cdot\nabla_x u(t,x) \\ \qquad + f\big(t,x,u(t,x),\sigma^T(t,x)\nabla_x u(t,x)\big) = 0, \quad (t,x)\in[0,T)\times\mathbb{R}^d, \\ u(T,x) = g(x), \quad x\in\mathbb{R}^d. \end{cases} \tag{1}$$

Although traditional numerical methods such as the Finite Difference Method (FDM) and Finite Element Method (FEM) perform well in handling low-dimensional cases ($d \le 3$), they struggle to solve high-dimensional problems due to the "curse of dimensionality" [12].

Recent deep–learning approaches–PINNs [8, 27, 30, 33, 37], Deep Galerkin [19, 34], and Deep Ritz [23, 39]–have shown promise for high–dimensional PDEs. However, sampling strategies become challenging in very high dimensions (e.g., $d \ge 100$). This motivates alternative deep learning approaches based on stochastic processes (Monte Carlo sampling), which include the Deep Backward Stochastic Differential Equation (BSDE) method [14], Deep Splitting [1], Deep Backward Dynamic Programming (DBDP) method [21] and recent

*Email: 20234013002@stu.suda.edu.cn
†Email: xyyue@suda.edu.cn
‡Co-Corresponding author: xia@math.northwestern.edu
§Co-corresponding author: xinli2023@u.northwestern.edu

martingale-based approaches [5, 6]. Among these, the Deep BSDE method is a widely adopted technique for handling such highly high–dimensional PDEs (1). This method leverages neural networks to approximate the gradient of the solutions to high–dimensional semilinear parabolic PDEs, utilizing the capacity of these networks to manage the stochastic nature of the solutions.

Despite its innovative approach and empirical success, the Deep BSDE method faces fundamental theoretical and computational limitations. Convergence analysis requires the generator function f to satisfy global Lipschitz conditions [2, 15]. This excludes important PDE classes arising in applications, notably Allen-Cahn equations with cubic nonlinearity ($f = y - y^3$) and Hamilton-Jacobi-Bellman (HJB) equations with quadratic gradient growth ($f = -\frac{1}{2}|z|^2$). While numerical convergence has been observed for such equations [13, 14], theoretical justification remained absent. From a practical perspective, the method requires careful balance of approximation, generalization, and optimization errors through appropriate network architecture design. Standard feedforward networks, despite universal approximation properties [20], face scalability limitations in the multi-network Deep BSDE framework.

In this context, we introduce two major contributions to address these theoretical and practical challenges. First, we provide the rigorous convergence analysis for Deep BSDE method applied to non-Lipschitz PDEs, specifically Allen-Cahn type equations and HJB type equations. For Allen-Cahn equations, we exploit boundedness properties of double-well potential dynamics to establish local Lipschitz conditions. For HJB equations, we prove convergence through systematic analysis within the Bouchard-Touzi-Zhang framework [41], demonstrating that the Deep BSDE scheme converges to truncated BSDE systems. This theoretical advancement expands the class of PDEs for which Deep BSDE methods have guaranteed convergence. Second, we introduce XNet [25, 26], a parameter-efficient neural architecture designed based on Cauchy's approximation theorem. XNet achieves $\mathcal{O}(L)$ parameter complexity compared to $\mathcal{O}(HL^2)$ for traditional feedforward networks, dramatically reducing computational burden while maintaining superior approximation capabilities. This addresses the approximation-optimization trade-off critical in multi-network Deep BSDE implementations. We provide comprehensive numerical validation through 100-dimensional Allen-Cahn equations and nonlinear financial derivative pricing problems. The XNet-enhanced Deep BSDE method demonstrates superior computational efficiency and accuracy, with clearer convergence behavior observable across both test cases. Notably, for Allen-Cahn equations in continuous-time implementation, XNet enables observation of convergence rates approaching 1.6, which remain obscured when using standard feedforward architectures.

Main contributions. This work delivers both a theoretical and a computational advance for Deep BSDE. Theoretically, we develop a convergence framework for non-Lipschitz generators (Allen–Cahn with cubic nonlinearity; HJB with quadratic gradient growth). For Allen–Cahn, we exploit a bounded double-well lemma to recover local Lipschitz control; for HJB, we combine truncation with the Bouchard–Touzi–Zhang (BTZ) reformulation and identify a martingale target that the learner must approximate. The resulting error bounds are architecture-agnostic: they depend only on the accuracy of the target approximation at each time layer (time-averaged Z under Lipschitz generators vs. a martingale target for quadratic HJB), not on any particular network. Computationally, we instantiate the framework with XNet [25, 26], a compact architecture motivated by Cauchy-type approximation that achieves strong accuracy with substantially fewer parameters, thereby reducing the optimization burden inherent in the multi-network Deep BSDE setting. Experiments on 100-dimensional benchmarks corroborate the predicted convergence behavior and demonstrate significant efficiency gains.

1. **Convergence analysis beyond Lipschitz.** A rigorous framework covering Allen–Cahn (cubic nonlinearity) and HJB (quadratic gradient growth), built on a bounded double-well lemma and a truncated–BTZ analysis.

2. **Efficient instantiation and verification.** XNet serves as an efficient instantiation that reduces the target-approximation error under fixed compute; 100D Allen–Cahn and nonlinear pricing tests show lower errors, faster runtimes, and clearer convergence behavior than standard feedforward implementations.

Outline of the article The remainder is organized as follows. Section 2 reviews the Deep BSDE method. Section 3 establishes convergence theory for Deep BSDE methods, extending the analysis to non-Lipschitz generators including Allen-Cahn and HJB equations. Sections 4 and 5 present discrete-time and

continuous-time implementations, respectively, demonstrating XNet's superior approximation capabilities and convergence properties. Section 6 concludes the paper.

2 Deep BSDE Method

In this section, we begin by introducing the BSDE system related to semilinear parabolic partial differential equations (PDEs). Subsequently, we will present the Deep BSDE (DBSDE) method proposed by E et al. [14].

2.1 Forward Backward Stochastic Differential Equation (FBSDE)

Following the seminal work of Peng [28, 29], there exists a well-established connection between the semilinear parabolic PDE (1) and backward stochastic differential equations (BSDEs). This probabilistic representation forms the theoretical foundation for the Deep BSDE method.

Consider a filtered probability space $(\Omega, \mathcal{F}, \mathbb{P})$ equipped with a d-dimensional standard Brownian motion $W = \left(W^{(1)}, \ldots, W^{(d)}\right) : [0,T] \times \Omega \to \mathbb{R}^d$, where $\{\mathcal{F}_t\}_{t\in[0,T]}$ denotes the natural filtration generated by W. Let $X = \left(X^{(1)}, \ldots, X^{(d)}\right) : [0,T] \times \Omega \to \mathbb{R}^d$, $Y : [0,T] \times \Omega \to \mathbb{R}$, and $Z : [0,T] \times \Omega \to \mathbb{R}^d$ be $\mathcal{F}$-adapted stochastic processes satisfying the following forward-backward stochastic differential equation (FBSDE) system:

$$\begin{cases} X_t = \xi + \displaystyle\int_0^t \mu\left(s, X_s\right) \mathrm{d}s + \int_0^t \sigma\left(s, X_s\right) \mathrm{d}W_s, & (2) \\ Y_t = g\left(X_T\right) + \displaystyle\int_t^T f\left(s, X_s, Y_s, Z_s\right) \mathrm{d}s - \int_t^T \left(Z_s\right)^T \ \mathrm{d}W_s. & (3) \end{cases}$$

Under certain conditions, this FBSDE system admits a unique solution and provides the probabilistic representation of the PDE solution. Specifically, for any $t \in [0,T]$, the following equation holds almost surely in probability,

$$\begin{cases} Y_t = u\left(t, X_t\right), \\ Z_t = \sigma^T\left(t, X_t\right) \nabla u\left(t, X_t\right). \end{cases} \tag{4}$$

This connection enables the reformulation of the deterministic PDE-solving problem into a stochastic framework. Specifically, if the d-dimensional process $\{X_t\}_{t\in[0,T]}$ satisfies the forward SDE (2), then the PDE solution satisfies the following integral representation:

$$\begin{aligned} u(t, X_t) - u(0, \xi) = &- \int_0^t f\left(s, X_s, u(s, X_s), \sigma^T(s, X_s) \nabla u(s, X_s)\right) ds \\ &+ \int_0^t \left[\nabla u(s, X_s)\right]^T \sigma(s, X_s) dW_s. \end{aligned} \tag{5}$$

2.2 Implementation of the Deep BSDE Method

After adapting the solution of PDE (1) to the SDE (5) by BSDE theory, given a partitioning of the interval $[0,T]$, $\pi : 0 = t_0 < t_1 < \cdots < t_N = T$ with $\Delta t_n = t_{n+1} - t_n$, the solution at each time step can be approximated with the Euler-Maruyama scheme,

$$\begin{aligned} u\left(t_{n+1}, X_{t_{n+1}}\right) - u\left(t_n, X_{t_n}\right) \approx &- f\left(t_n, X_{t_n}, u\left(t_n, X_{t_n}\right), \sigma^T\left(t_n, X_{t_n}\right) \nabla u\left(t_n, X_{t_n}\right)\right) \Delta t_n \\ &+ \left[\nabla u\left(t_n, X_{t_n}\right)\right]^{\mathrm{T}} \sigma\left(t_n, X_{t_n}\right) \Delta W_n, n = 0, 1, ..., N-1, \end{aligned} \tag{6}$$

where $\Delta W_n = W_{t_{n+1}} - W_{t_n}$ represents the Brownian motion increment over the time interval $[t_n, t_{n+1}]$. To achieve a globally approximate scheme, neural networks can be incorporated into the forward discretization process (6). The first step towards this is to obtain training data by sampling M independent paths $\left\{X_{t_n}^m\right\}_{0\le n\le N}^{m=1,2,\ldots,M}$, where $\left\{X_{t_0}^m\right\}^{m=1,\ldots,M} = \xi$. The critical step next is employing the neural network parameters θ_{u_0}, $\theta_{\nabla u_0}$ to approximate the solution and the gradient function at $t = t_0$ respectively. Let θ_n represent all network parameters approximating the gradient function $\nabla u(t,x)$ by the neural network at

time $t = t_n$, where $n = 1, 2, ..., N-1$. With the above approximations, the total set of parameters is $\theta = \{\theta_{u_0}, \theta_{\nabla u_0}, \theta_1, \theta_2, \ldots, \theta_{N-1}\}$. The equation (6) can be rewritten as follows.

$$\begin{aligned}\hat{u}\left(t_{n+1}, X^m_{t_{n+1}}\right) - \hat{u}\left(t_n, X^m_{t_n}\right) = -f\left(t_n, X^m_{t_n}, \hat{u}\left(t_n, X^m_{t_n}\right), \sigma^T\left(t_n, X^m_{t_n}\right) \nabla u^\theta_{t_n}\left(t_n, X^m_{t_n}\right)\right) \Delta t_n \\ + \left[\nabla u^\theta_{t_n}\left(t_n, X^m_{t_n}\right)\right]^{\mathrm{T}} \sigma\left(t_n, X^m_{t_n}\right) \Delta W^m_{t_n}, m = 1, 2, ..., M, n = 0, 1, ..., N-1.\end{aligned} \tag{7}$$

In particular, when $t = t_0$, we have

$$\hat{u}\left(t_1, X^m_{t_1}\right) = \theta_{u_0} - f\left(t_0, \xi, \theta_{u_0}, \sigma^T\left(t_0, \xi\right) \theta_{\nabla u_0}\right) \Delta t_0 + \left[\theta_{\nabla u_0}\right]^{\mathrm{T}} \sigma\left(t_0, \xi\right) \Delta W^m_{t_0}, m = 1, 2, ..., M. \tag{8}$$

By applying the globally approximate scheme (7), an approximate value of $u\left(t_N, X^m_{t_N}\right)$, denoted as $\hat{u}\left(t_N, X^m_{t_N}\right)$, can be output, where $m = 1, 2, \ldots, M$. The matching of a given terminal condition defines the expected loss function,

$$l(\theta) = \frac{1}{M} \sum_{m=1}^{M} \left| g\left(X^m_T\right) - \hat{u}\left(t_N, X^m_{t_N}\right) \right|^2. \tag{9}$$

Through optimizing the network parameters, it becomes evident that an approximate solution θ_{u_0} for $u(0, \xi)$ can be obtained.

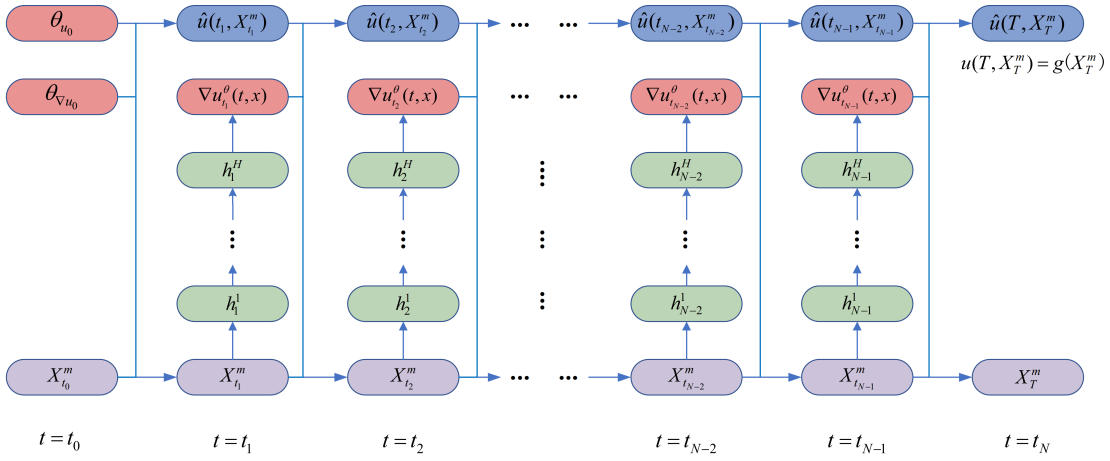

Figure 1: The neural network architecture for Deep BSDE method. The network consists of multiple $(N-1)$ sub-networks, with each sub-network corresponding to a time interval. Each sub-network has H hidden layers. It should be noted that in addition to these, θ_{u_0} and $\theta_{\nabla u_0}$ are also network parameters that need to be optimized.

The Deep BSDE method involves three error types: approximation, generalization, and optimization errors. Proper network architecture design minimizes approximation errors and optimization cost. Figure 1 illustrates the multi-network architecture, highlighting parameter optimization complexity in discrete-time settings.

2.3 Network Architecture for the Deep BSDE Method: Feedforward Neural Networks and XNet

The original Deep BSDE method [14] employed feedforward neural networks (FNNs) for gradient approximation. As demonstrated in Section 3, the convergence properties of the Deep BSDE method critically depend on the approximation capabilities of the underlying neural network architecture. Traditional FNNs struggle to balance approximation capability with optimization cost. We propose replacing FNNs with shallow XNet architectures, demonstrating that XNet provides enhanced approximation efficiency while maintaining computational tractability.

The mathematical framework of feedforward neural networks (FNNs) is:

$$Y^{\theta}_{\mathrm{FNN}}(X;W,b) = \sigma_H(W_H\sigma_{H-1}(\cdots\sigma_1(W_1X+b_1)+\cdots+b_{H-1})+b_H)\,, \tag{10}$$

with parameter sets $W = \{W_1, W_2, \ldots, W_H\}$, $b = \{b_1, b_2, \ldots, b_H\}$. Here, $X \in \mathbb{R}^d$ is the input, $W_i \in \mathbb{R}^{n_i \times n_{i-1}}$ and $b_i \in \mathbb{R}^{n_i}$ denote the weight matrices and bias vectors for the i-th layer, respectively, with n_i representing the number of neurons in layer i. The activation functions σ_i (commonly sigmoid, tanh, or ReLU) introduce the requisite nonlinearity for function approximation.

The feedforward networks employed in the original Deep BSDE implementation typically consist of H hidden layers with L neurons each, yielding a total parameter complexity of $\mathcal{O}(HL^2)$. While the universal approximation theorem [20] establishes that sufficiently large FNNs guarantee approximation capability, the parameter growth presents significant computational challenges, particularly in the multi-network Deep BSDE framework where optimization complexity scales dramatically with network size.

Theorem 1 (Cauchy Approximation Theorem; Theorem 1 in Xia [25])**.** *Let $f(z^1, \ldots, z^d)$ be analytic on an open set $U \subset \mathbb{C}^d$, and let $M \subset U$ be compact. Then for any $\varepsilon > 0$, there exist points $(\xi^1_k, \ldots, \xi^d_k) \in U$, $k = 1, \ldots, L$, and coefficients $\lambda_1, \ldots, \lambda_L \in \mathbb{C}$ such that*

$$\sup_{z\in M}\left|f(z) - \sum_{k=1}^{L}\frac{\lambda_k}{\prod_{j=1}^{d}(\xi^j_k - z^j)}\right| < \varepsilon. \tag{11}$$

Moreover, the approximation error admits algebraic convergence of arbitrarily high order in L.

Remark 1 (Precise meaning of the convergence order)**.** ***The statement in Theorem 1 that the approximation admits algebraic convergence of arbitrarily high order*** *means that for any $p > 0$, there exists a constant $C_p > 0$, depending on p and on f but independent of L, such that*

$$\sup_{z\in M}\left|f(z) - \sum_{k=1}^{L}\frac{\lambda_k}{\prod_{j=1}^{d}(\xi^j_k - z^j)}\right| \le C_p\, L^{-p}.$$

In fact, when f is analytic in a neighborhood of M, the approximation error decays geometrically (i.e., exponentially) with respect to L, from which the above algebraic bounds follow immediately. Here p is an arbitrarily prescribed convergence order. For each fixed $p > 0$, the approximation error admits an $O(L^{-p})$ bound with a constant depending on p and the target function f, but independent of L. In the analytic case, this follows from the underlying exponential convergence.

Motivated by the Cauchy approximation theorem and validated through extensive empirical studies across function approximation, PDE solving, and reinforcement learning applications, Xia et al. [25, 26] developed the XNet architecture with the following mathematical formulation:

$$\begin{aligned} Y^{\theta}_{\mathrm{XNet}}(X) &= \mathrm{Re}\left(\sum_{k=1}^{L}\frac{\alpha_k + i\beta_k}{\sum_{j=1}^{d}a^j_k x_j + c_k + ie_k}\right) \\ &= \sum_{k=1}^{L}\left(\alpha_k\frac{\sum_{j=1}^{d}a^j_k x_j + c_k}{\left(\sum_{j=1}^{d}a^j_k x_j + c_k\right)^2 + e_k^2} + \beta_k\frac{e_k}{\left(\sum_{j=1}^{d}a^j_k x_j + c_k\right)^2 + e_k^2}\right), \end{aligned} \tag{12}$$

where $X = (x_1, x_2, ..., x_d)$ represents the d-dimensional input, $Y^{\theta}_{\mathrm{XNet}}$ denotes the output, and $\alpha_k, \beta_k, a^j_k, c_k$, and e_k are trainable parameters.

The architectural simplicity of shallow XNet achieves an better balance between approximation capability and optimization cost. With $\mathcal{O}(L)$ parameters versus $\mathcal{O}(HL^2)$ for traditional FNNs, XNet dramatically reduces computational burden while maintaining superior approximation power. This efficiency is particularly valuable in Deep BSDE implementations requiring simultaneous training of multiple networks across temporal layers. Combined with dimension-independent convergence properties, XNet enables the extension of Deep BSDE methods to high-dimensional problems previously considered computationally intractable.

3 Convergence of Deep BSDE method

The Deep BSDE method transforms the PDE solving problem into a stochastic control framework. Consider the Euler discretization scheme:

$$\begin{cases} X_0^\pi = \xi, \quad Y_0^\pi = \theta_{u_0}, \quad Z_0^{\theta,\pi} = \sigma^{\mathrm{T}}(t_0,\xi)\theta_{\nabla u_0}, \\ X_{t_{n+1}}^\pi = X_{t_n}^\pi + \mu\left(t_n, X_{t_n}^\pi\right)\Delta t_n + \sigma\left(t_n, X_{t_n}^\pi\right)\Delta W_n, \\ Y_{t_{n+1}}^\pi = Y_{t_n}^\pi - f\left(t_n, X_{t_n}^\pi, Y_{t_n}^\pi, Z_{t_n}^{\theta,\pi}\right)\Delta t_n + \left[Z_{t_n}^{\theta,\pi}\right]^{\mathrm{T}}\Delta W_n, \\ Z_{t_n}^{\theta,\pi} = \sigma^{\mathrm{T}}\left(t_n, X_{t_n}^\pi\right)\nabla u_{t_n}^\theta\left(t_n, X_{t_n}^\pi\right), \end{cases} \tag{13}$$

where $\nabla u_{t_n}^\theta$ represents the neural network approximation at time step n, which serves as the control strategy. The objective is to solve the optimization problem:

$$\inf_{\theta_{u_0},\theta_{\nabla u_0}\in\mathcal{N}_0,\theta_i\in\mathcal{N}_i} F(\theta) = \mathbb{E}\left[\left|g\left(X_T^\pi\right) - Y_{t_N}^\pi\right|^2\right], \tag{14}$$

where $\mathcal{N}_0$ and $\mathcal{N}_i (0 \le i \le N-1)$ are parametric function spaces generated by neural networks.

In this section, we establish a comprehensive convergence theory for the Deep BSDE method, extending beyond the classical Lipschitz framework. The theoretical framework addresses three primary sources of computational error: approximation errors arising from temporal discretization and neural network representation, generalization errors from Monte Carlo sampling, and optimization errors due to network complexity and scale. Under the assumption that neural network approximation errors and generalization errors are sufficiently small relative to discretization errors, we establish convergence results for three fundamental categories of generators.

3.1 Convergence Analysis for Lipschitz Generators

The foundational convergence theory for the Deep BSDE method was established by Han et al. [15], which requires the generator function f to satisfy a global Lipschitz condition. Under standard regularity Assumptions 1 and 2 (detailed in Appendix A), this framework provides rigorous convergence guarantees for the Deep BSDE approximation scheme.

Lemma 1. *Under Assumptions 1 and 2, there exists a constant C_1, independent of $h = \max_n \Delta t_n$, d, and M, such that*

$$\begin{aligned} &\sup_{t_n\in[0,T]}\left(\mathbb{E}\left|X_{t_n} - X_{t_n}^\pi\right|^2 + \mathbb{E}\left|Y_{t_n} - Y_{t_n}^\pi\right|^2\right) + \sum_{n=0}^{N-1}\int_{t_n}^{t_{n+1}} \mathbb{E}\left|Z_t - Z_{t_n}^\pi\right|^2 dt \\ &\qquad \le C_1\left[h + \mathbb{E}\left|g\left(X_T^\pi\right) - Y_T^\pi\right|^2\right]. \end{aligned} \tag{15}$$

Lemma 2. *Under Assumptions 1 and 2, there exists a constant C_2, independent of h, d and M, such that*

$$\begin{aligned} &\inf_{\theta_{u_0},\theta_{\nabla u_0}\in\mathcal{N}_0,\phi_n\in\mathcal{N}_n} \mathbb{E}\left|g\left(X_T^\pi\right) - Y_T^\pi\right|^2 \\ &\le C_2\left\{h + \inf_{\theta_{u_0},\theta_{\nabla u_0}\in\mathcal{N}_0} \mathbb{E}\left|Y_0^\pi - \theta_{u_0}\right|^2 + \mathbb{E}\left|Z_0^\pi - \theta_{\nabla u_0}\right|^2 h\right. \\ &\qquad \left. + \inf_{\phi_n^\theta\in\mathcal{N}_n}\sum_{n=1}^{N-1}\mathbb{E}\left|\mathbb{E}\left[\tilde{Z}_{t_n} \mid X_{t_n}^\pi\right] - \phi_n^\theta\left(t_n, X_{t_n}^\pi\right)\right|^2 h\right\}, \end{aligned} \tag{16}$$

where $\tilde{Z}_{t_n} = h^{-1}\mathbb{E}\left[\int_{t_n}^{t_{n+1}} Z_t \mathrm{d}t \mid \mathcal{F}_{t_n}\right]$.

Lemmas 1 and 2 establish bounds on the absolute error of the Deep BSDE method. The detailed proofs can be found in Han et al. [15]. Given our primary interest in relative error calculations, we further derive bounds on the relative error:

$$\begin{aligned} \mathbb{E}\left|\frac{u(0,\xi) - \theta_{u_0}}{u(0,\xi)}\right|^2 \le & C\left\{\frac{h}{u(0,\xi)^2} + \inf_{\theta_{u_0},\theta_{\nabla u_0}\in\mathcal{N}_0}\mathbb{E}\left|Z_0^\pi - \theta_{\nabla u_0}\right|^2\frac{h}{u(0,\xi)^2}\right. \\ &\left. + \inf_{\phi_n^\theta\in\mathcal{N}_n}\sum_{n=1}^{N-1}\mathbb{E}\left|\mathbb{E}\left[\tilde{Z}_{t_n} \mid X_{t_n}^\pi\right] - \phi_n^\theta\left(t_n, X_{t_n}^\pi\right)\right|^2\frac{h}{u(0,\xi)^2}\right\}, \end{aligned} \tag{17}$$

where constant C is independent of h, d and M.

Remark 2. *This analysis reveals that the approximation errors of the Deep BSDE method consist of two primary components: time discretization errors (standard in numerical methods) and neural network approximation errors. When $u(0,\xi)$ is sufficiently large, the time discretization error becomes negligible, and the overall approximation error is dominated by the network's approximation capability.*

While this theoretical framework is well-established for Lipschitz generators, many practically important PDEs violate the global Lipschitz condition. Notable examples include Allen-Cahn equations with cubic nonlinearity ($f(t,x,y,z) = y - y^3$) and Hamilton-Jacobi-Bellman equations with quadratic gradient growth ($f(t,x,y,z) = -\frac{1}{2}|z|^2$) from the original Deep BSDE work [14]. Although these equations demonstrated numerical convergence in empirical studies, their theoretical convergence remained unexplained within the existing Lipschitz framework. To address this theoretical gap, we extend the convergence analysis to these two important classes of non-Lipschitz equations. Through novel analytical techniques that exploit the special structural properties of these generators, we establish convergence results that provide the first rigorous theoretical justification for the empirical success observed in practice.

3.2 Convergence Analysis for Allen-Cahn Type Equations

Allen-Cahn equations with cubic nonlinearity represent a fundamental class of reaction-diffusion systems arising in phase field theory and materials science. Consider the d-dimensional Allen-Cahn equation:

$$\begin{cases} \frac{\partial u}{\partial t}(t,x) + \frac{1}{2}\mathrm{Tr}[\sigma\sigma^T(t,x)D^2u(t,x)] + \mu(t,x)\cdot\nabla u(t,x) + u(t,x) - [u(t,x)]^3 = 0, \\ \quad (t,x) \in [0,T)\times\mathbb{R}^d, \\ u(T,x) = g(x), \quad x\in\mathbb{R}^d, \end{cases} \tag{18}$$

where the generator function $f(t,x,y,z) = y - y^3$ exhibits the characteristic double-well potential structure. The cubic nonlinearity formally violates the global Lipschitz condition required in standard convergence theory.

The key insight for establishing convergence lies in exploiting the intrinsic boundedness properties induced by the double-well potential. The critical observation is that solutions to the BSDE (19) exhibit natural bounds that prevent explosive growth, despite the super-linear nonlinearity.

Lemma 3 (Boundedness Properties of Double-Well Dynamics)**.** *Let $(Y_t, Z_t)_{t\in[0,T]}$ be the adapted solution to the BSDE*

$$Y_t = Y_T + \int_t^T \left(Y_s - Y_s^3\right) ds - \int_t^T Z_s\, dW_s, \tag{19}$$

where $Y_T \in L^2(\Omega)$. Then there exists a constant $C > 0$, depending only on T and $\mathbb{E}[Y_T^2]$, such that

$$\sup_{t\in[0,T]} \mathbb{E}\left[Y_t^2\right] \le C.$$

In particular,

$$\sup_{t\in[0,T]} \mathbb{E}\left[|Y_t|\right] < \infty.$$

The proof follows from explicit integration using separation of variables (detailed in Appendix B). This boundedness result is crucial for controlling the cubic nonlinearity in the BSDE framework.

Theorem 2 (Convergence for Allen-Cahn Equations)**.** *Consider the Allen-Cahn equation* (18) *with generator function $f(t,x,y,z) = y - y^3$. Under Assumptions 1, 2, the Deep BSDE method* (13) *converges. Specifically, there exists a constant $C_1, C_2 > 0$, independent of h, d, and M, such that*

$$\begin{aligned} \sup_{0\le n\le N} \mathbb{E}\left|Y_{t_n} - Y_n^\pi\right|^2 &\le C_1\left[h + \mathbb{E}\left|g\left(X_T^\pi\right) - Y_T^\pi\right|^2\right] \\ \inf_{\theta_{u_0},\theta_{\nabla u_0}\in\mathcal{N}_0,\phi_n\in\mathcal{N}_n} \mathbb{E}\left|g\left(X_T^\pi\right) - Y_T^\pi\right|^2 &\le C_2\left\{h + \inf_{\theta_{u_0},\theta_{\nabla u_0}\in\mathcal{N}_0} \mathbb{E}\left|Y_0^\pi - \theta_{u_0}\right|^2 + \mathbb{E}\left|Z_0^\pi - Z_0^{\theta,\pi}\right|^2 h\right. \\ &\quad \left. + \inf_{\phi_n^\theta\in\mathcal{N}_n} \sum_{n=1}^{N-1} \mathbb{E}\left|\mathbb{E}\left[\tilde{Z}_{t_n} \mid X_{t_n}^\pi\right] - Z_n^{\theta,\pi}\right|^2 h\right\}. \end{aligned} \tag{20}$$

Proof. From the BSDE representation (equation 3) and Lemma 3, we know that $\mathbb{E}|Y_t|$ is bounded. Specifically, for any two solutions $Y_{1,t}$ and $Y_{2,t}$, we have:

$$\begin{aligned} &\mathbb{E}|f(t,X_t,Y_{1,t},Z_t)-f(t,X_t,Y_{2,t},Z_t)| \\ =&\mathbb{E}|Y_{1,t}-Y_{2,t}||1-Y_{1,t}^2-Y_{1,t}Y_{2,t}-Y_{2,t}^2| \\ \leq& K\mathbb{E}|Y_{1,t}-Y_{2,t}|, \end{aligned} \tag{21}$$

where the constant K depends on the uniform bounds of Y_t. This local Lipschitz property enables application of the standard convergence framework. □

Remark 3. *This result resolves the theoretical gap for Allen-Cahn equations, providing rigorous justification for the numerical convergence observed in the original Deep BSDE work [14] and our numerical experiments in Section 5.*

3.3 Convergence Analysis for HJB Type Equations

Hamilton-Jacobi-Bellman (HJB) equations represent a fundamental class of nonlinear PDEs arising in stochastic optimal control and mathematical finance. These equations present significant theoretical challenges due to their quadratic gradient dependence, which violates standard Lipschitz conditions required for classical convergence analysis. Consider the d-dimensional HJB equation:

$$\begin{cases} \frac{\partial u}{\partial t}(t,x)+\frac{1}{2}\mathrm{Tr}[\sigma\sigma^T(t,x)D^2u(t,x)]+\mu(t,x)\cdot\nabla u(t,x)-\frac{1}{2}|\sigma^T(t,x)\nabla u(t,x)|^2=0, \\ u(T,x)=g(x), \quad x\in\mathbb{R}^d, \end{cases} \tag{22}$$

where the generator function is $f(t,x,y,z)=-\frac{1}{2}|z|^2$. The quadratic dependence on z violates the standard Lipschitz condition in H_2 in Assumption 1, presenting a significant theoretical challenge.

The convergence analysis for HJB equations requires a fundamentally different approach than the Allen-Cahn case. The quadratic growth of the generator with respect to the control variable z can lead to explosive behavior, necessitating sophisticated truncation and regularization techniques. Our proof strategy establishes convergence through a systematic analysis of four interconnected stochastic systems.

We begin by introducing a truncated BSDE system where the quadratic generator is regularized through projection onto a ball of radius B:

$$\begin{cases} X_t=\xi+\displaystyle\int_0^t \mu(s,X_s)\,ds+\int_0^t \sigma(s,X_s)\,dW_s, \\ Y_t^B=g(X_T)+\displaystyle\int_t^T f^B(s,X_s,Y_s^B,Z_s^B)\,ds-\int_t^T Z_s^B\,dW_s, \end{cases} \tag{23}$$

where $f^B(\cdot,\cdot,\cdot,z)=f(\cdot,\cdot,\cdot,\varphi^B(z))$ and φ^B is the projection on the centered Euclidean ball of radius ρB with $\rho>0$ chosen such that f^B is B-Lipschitz-continuous with respect to z.

The truncation error between the original and truncated systems is controlled by the following result from the BSDE literature:

Lemma 4 (Truncation Error Control; Theorem 6.2 in [22], Remark 5.5 in [31])**.** *Under the H_1, H_3 in Assumption 1 and Assumption 3, for any $p\geq 1$ and $\beta\geq 1$, there exist positive constants C_p and D_β such that*

$$\mathbb{E}\left[\sup_{t\in[0,T]}|Y_t^B-Y_t|^{2p}\right]+\mathbb{E}\left[\Big(\int_0^T|Z_s^B-Z_s|^2\,ds\Big)^p\right]\leq C_pD_\beta B^{-\frac{\beta}{2\bar{q}}}.$$

This lemma establishes that the truncated system (23) converges to the original BSDE system (2) as $B\to\infty$. The constants C_p and D_β, as well as the Hölder conjugate $\bar{q}$, are characterized in Theorem 6.2 of Imkeller [22] and Remark 5.5 of Richou [31].

Having established truncation error control, our primary task reduces to proving convergence of the Deep BSDE system (13) to the truncated system (23). This requires a reformulation within the Bouchard–Touzi–Zhang

(BTZ) framework. Right multiplying $(\Delta W_i)^T$ on both sides of (13), and taking the expectation conditional expectation $\mathbb{E}[\cdot \,|\, \mathcal{F}_n]$ again, we obtain

$$\mathbb{E}\big[Y_{n+1}^{\pi}(\Delta W_n)^T\big|\mathcal{F}_n\big] = h\, Z_n^{\theta,\pi}.$$

The above observation motivates us to consider the following BTZ scheme [4, 40]:

$$\begin{aligned}
\overline{X}_0^{\pi} &= \xi, \\
\overline{X}_{n+1}^{\pi} &= \overline{X}_n^{\pi} + \mu(t_n, \overline{X}_n^{\pi})h + \sigma(t_n, \overline{X}_n^{\pi})\Delta W_n, \\
\overline{Y}_n^{\pi} &= \mathbb{E}_n[\overline{Y}_{n+1}^{\pi} + f^B(t_n, \overline{X}_n^{\pi}, \overline{Y}_n^{\pi}, \overline{Z}_n^{\pi})h], \\
\overline{Z}_n^{\pi} &= \frac{1}{h}\mathbb{E}_n[\overline{Y}_{n+1}^{\pi}\Delta W_n].
\end{aligned} \tag{24}$$

The truncated system (23) also admits a perturbed system (24) representation.

$$\begin{aligned}
X_{n+1}^{\pi} &= X_n^{\pi} + \mu(t_n, X_n^{\pi})h + \sigma(t_n, X_n^{\pi})\Delta W_n + \Upsilon_n^X, \\
\widetilde{Y}_n^{\pi} &= \mathbb{E}_n[\widetilde{Y}_{n+1}^{\pi} + hf^B(t_n, X_n^{\pi}, \widetilde{Y}_n^{\pi}, \widetilde{Z}_n^{\pi})] + \Upsilon_n^Y, \\
\widetilde{Z}_n^{\pi} &= \frac{1}{h}\mathbb{E}_n[\widetilde{Y}_{n+1}^{\pi}\Delta W_n],
\end{aligned} \tag{25}$$

where $\widetilde{Y}_n^{\pi} = Y_{t_n}^B$, and the perturbation term satisfies:

$$\begin{aligned}
\Upsilon_n^X &= \int_{t_n}^{t_{n+1}} \mu(s, X_s) - \mu(t_n, X_n^{\pi})\, ds + \int_{t_n}^{t_{n+1}} \sigma(s, \widetilde{X}_s) - \sigma(t_n, X_n^{\pi})\, dW_s, \\
\Upsilon_n^Y &= \mathbb{E}_n\left[\int_{t_n}^{t_{n+1}} \big(f^B(s, X_s, Y_s^B, Z_s^B) - f^B(t_n, X_n^{\pi}, Y_{t_n}^B, \widetilde{Z}_n^{\pi})\big)ds\right].
\end{aligned} \tag{26}$$

We prove that the Deep BSDE system converges to the truncated BSDE system within the BTZ framework by Theorem 3 and Theorem 4.

Theorem 3. *Let $\big(X_n^{\pi}, \widetilde{Y}_n^{\pi}, \widetilde{Z}_n^{\pi}\big)$ denote the solution to the truncated BSDE system (25), $\big(X_n^{\pi}, Y_n^{\pi}, Z_{t_n}^{\theta,\pi}\big)$ denote the solution of the Deep BSDE (DBSDE) scheme (13). Under the H_1, H_3 in Assumption 1, Assumption 2 and Assumption 3, there exists a constant C, such that*

$$\sup_{0\le n\le N} \mathbb{E}|\widetilde{Y}_n^{\pi} - Y_n^{\pi}|^2 \le C(h^{\frac{1}{2}} + \mathbb{E}|\widetilde{Y}_N^{\pi} - Y_N^{\pi}|^2).$$

Theorem 4. *Under the H_1, H_3 in Assumption 1, Assumption 2 and Assumption 3, the Deep BSDE system converges to the truncated BSDE system with rate*

$$\mathbb{E}|\widetilde{Y}_N^{\pi} - Y_N^{\pi}| \le e^{C_f T}(\mathbb{E}_0|Y_0 - \theta_{u_0}| + \sum_{n=0}^{N-1} \mathbb{E}_n\Big|\Big(Z_n^{\theta,\pi} - \widetilde{Z}_n^{\pi}\Big)^2\Big|h) + O(h^{\frac{1}{2}}), \tag{27}$$

where $\widetilde{Z}_n^{\pi} = \frac{1}{h}\mathbb{E}_n[Y_{t_{n+1}}^B \Delta W_n]$ represents the target approximation for the neural networks.

The proof of Theorem 4 requires careful analysis of the perturbation terms arising from the continuous-to-discrete approximation. The key technical challenge lies in controlling the quadratic nonlinearity while maintaining the martingale structure. The complete proof is provided in Appendix C. Combining the results of Lemma 4 and Theorem 4 yields our main convergence theorem:

Theorem 5 (Convergence for HJB Equations)**.** *Under the H_1, H_3 in Assumption 1, Assumption 2 and Assumption 3, the Deep BSDE method converges for HJB equations with quadratic generators. Specifically, there exists a constant $C_1', C_2' > 0$ such that*

$$\begin{aligned}
\sup_{0\le n\le N} \mathbb{E}|Y_{t_n} - Y_n^{\pi}|^2 &\le C_1'(h^{1/2} + \mathbb{E}|g(X_T^{\pi}) - Y_N^{\pi}|^2 + C_p D_\beta B^{-\frac{\beta}{2\bar{q}}}) \\
\inf_{\theta_{u_0},\theta_{\nabla u_0}\in\mathcal{N}_0,\phi_n\in\mathcal{N}_n} \mathbb{E}\left|g(X_T^{\pi}) - Y_N^{\pi}\right| &\le C_2'\Bigg(h^{\frac{1}{2}} + \inf_{\theta_{u_0},\theta_{\nabla u_0}\in\mathcal{N}_0} \mathbb{E}\left|Y_0 - \theta_{u_0}\right|^2 + \mathbb{E}\left|\widetilde{Z}_0^{\pi} - Z_0^{\theta,\pi}\right|^2 h \\
&\qquad + \inf_{\phi_n^{\theta}\in\mathcal{N}_n} \sum_{n=1}^{N-1} \mathbb{E}\left|\widetilde{Z}_n^{\pi} - Z_n^{\theta,\pi}\right|^2 h\Bigg) + \sqrt{C_p D_\beta} B^{-\frac{\beta}{4q}}.
\end{aligned} \tag{28}$$

Remark 4. *The convergence analysis reveals a fundamental difference in the target approximation requirements between Lipschitz and quadratic generators. For Lipschitz generators, neural networks must approximate* $h^{-1}\mathbb{E}\left[\int_{t_n}^{t_{n+1}} Z_t\,dt \,\middle|\, \mathcal{F}_{t_n}\right]$*, which represents the time-averaged* Z *process. For quadratic generators, the target becomes* $\frac{1}{h}\mathbb{E}_n[Y^B_{t_{n+1}}\Delta W_n]$*, which captures the martingale structure of the truncated system.*

Remark 5. *This theoretical advancement significantly extends the applicability of Deep BSDE methods to nonlinear PDEs arising in portfolio optimization, risk management, and optimal control problems. The rigorous convergence guarantees provide theoretical foundation for the empirical success observed in computational finance applications [9, 13, 14].*

3.4 Generalization Errors and Optimization Errors for Deep BSDE method

In the Deep BSDE method, the loss function is defined as the expectation of matching the terminal condition (14). However, in practical computations, the exact expectation is not directly computed. Instead, we approximate it using the loss function (9). This approximation results in a generalization error,

$$\begin{aligned}\text{Generalization Error} &= \mathbb{E}|g\left(X_T\right) - Y_{t_N}|^2 - \frac{1}{M}\sum_{m=1}^{M}\left|g\left(X_T^m\right) - Y_{t_N}^m\right|^2 \\ &= \mathcal{L}\left(g\right) - \mathcal{L}\left(g_M\right).\end{aligned} \tag{29}$$

Lemma 5. *For Monte Carlo methods* (13) *that use i.i.d. samples, the convergence rate of the mean generalization error is*

$$\mathbb{E}\left[\mathcal{L}(g) - \mathcal{L}(g_M)\right] = O(M^{-1/2+\varepsilon}), \tag{30}$$

where ϵ *is arbitrarily small constant, function* g *satisfes the boundary growth condition* (31) *for some small constants* $(B_i)_{i=1}^d$*. This result follows directly from the work of Xiao et al. [38].*

Definition 1. *(Boundary growth condition). Let* $d \in \mathbb{N}$*, suppose* $\mathcal{G}$ *is a class of realvalued functions defined on* $(0,1)^d$*. We say that* $\mathcal{G}$ *satisfes the boundary growth condition with constants* $(B_i)_i = 1^d$ *if there exists* $B \in (0,\infty)$ *such that for every* $g \in \mathcal{G}$*, every subset* $v \subseteq \{1,\cdots,d\}$ *and every* $u = (u_1,\ldots,u_d) \in (0,1)^d$ *it holds that*

$$\left|\left(\prod_{i\in v}\partial/\partial x_i\right) g(u)\right| \le B\prod_{i=1}^{d}[\min(u_i, 1-u_i)]^{-B_i - \mathbf{1}\{i\in v\}}, \tag{31}$$

where $\mathbf{1}\{\cdot\}$ *is an indicator function.*

When using general Monte Carlo methods to sample trajectories (2), a significant number of samples is required to reduce the generalization error to an adequately small level. Therefore, it is necessary to introduce techniques such as importance sampling [36], quasi-Monte Carlo methods [35], Gibbs sampling [11], and other advanced sampling techniques. These methods enable the reduction of generalization errors with a smaller sampling cost.

The optimization problem aims to minimize the loss function (9) by adjusting network parameters. Smaller network architectures define more restricted function spaces, which reduce the parameter search space and promote faster convergence to high-quality optima. The resulting lower-dimensional optimization landscape enhances algorithmic efficiency, leading to reduced optimization error [24, 32]. This advantage is particularly important in the Deep BSDE framework, where multiple networks are trained simultaneously across time discretization steps.

Remark 6. *While the theoretical framework provides convergence guarantees, practical implementation depends heavily on controlling approximation and optimization errors. Generalization errors can be minimized by increasing the number of samples, but the choice of network architecture plays a pivotal role in controlling both approximation and optimization errors, which ultimately determine whether the theoretically predicted convergence behavior can be observed in computational experiments. The limitations of standard feedforward neural networks (FNNs) in achieving the required target approximation accuracy are evident in the experimental results. By incorporating the XNet architecture into the Deep BSDE framework, these shortcomings are effectively mitigated, allowing the method to achieve significantly smaller approximation and*

optimization errors under a constrained computational budget. This enhancement bridges the gap between theoretical convergence guarantees and practical implementation, revealing a more discernible convergence rate, as demonstrated in the numerical studies.

4 Discrete time models

In the Deep BSDE method, we fully connect the $N-1$ temporal steps of the neural networks and train them jointly. At each time step, two options are available for the neural network architecture. For the feedforward neural networks (FNNs), the architecture comprises one input layer (d-dimensional), two hidden layers (each with d+10 dimensions), and one output layer (d-dimensional). The XNet, on the other hand, consists of three components, including one input layer (d-dimensional), a hidden layer comprising d basis functions, and one output layer (d-dimensional). Through the following two numerical examples, we demonstrate how the Deep BSDE method achieves enhanced accuracy and computational efficiency by replacing feedforward neural networks with XNet. Note that the numerical results obtained by the Deep BSDE method using feedforward neural networks are based on the code provided by E et al. [14].

4.1 Allen-Cahn Equation

In this subsection, the Deep BSDE method is tested for solving the 100-dimensional Allen-Cahn partial differential equation (PDE) (32) using both the FNNs and XNet. With reference to the general form of the semilinear parabolic equation (1), we set $\alpha = 1$, $f(y,z) = y - y^3$, and $g(x) = \left[2 + \frac{2}{5}|x|^2_{\mathbb{R}^d}\right]^{-1}$. The PDE is represented as follows,

$$\begin{cases} \frac{\partial u}{\partial t}(t,x) + (\Delta_x u)(t,x) + u(t,x) - [u(t,x)]^3 = 0, (t,x) \in [0,T) \times \mathbb{R}^d, \\ u(T,x) = g(x), x \in \mathbb{R}^d, \end{cases} \tag{32}$$

where the spatial dimension is $d = 100$ and the terminal time $T = \frac{3}{10}$. Using the branching diffusion method [16, 17, 18], a reference value for the exact solution is obtained, $u(0,\xi) = u(0,0,\ldots,0) \approx 0.052802$. The Deep BSDE method is implemented by the FNNs and the XNet with setting the time step number to $N = 20, 30, 40, 80$, and conducting five independent runs for each configuration. During the training process, the numerical solution tends to stabilize after approximately 5000 iterations. Therefore, the average of the results from iterations 5000 to 10000 is taken as the computed value function. The results are presented in Table 1. Note that in the following figures, we use the term "Two-Layer Net" to specifically refer to the feedforward neural networks (FNNs) with two hidden layers as described above.

Under a 20-time-step discretization, as shown in Figure 2, it is observed that switching to the XNet results in a faster decrease of the loss function and an increase in computational speed. However, accuracy improvement is not significant. This can be explained by Equation (33), where the small value function indicates that the approximation error is dominated by the time discretization rather than the network approximation error,

$$\begin{aligned} \mathbb{E}\left|\frac{u(0,\xi) - \theta_{u_0}}{u(0,\xi)}\right|^2 \leq & C\left\{\frac{h}{0.052802^2} + \inf_{\theta_{u_0},\theta_{\nabla u_0} \in \mathcal{N}_0} \mathbb{E}\left|Z_0 - \theta_{\nabla u_0}\right|^2 \frac{h}{0.052802^2}\right. \\ & \left. + \inf_{\phi_n \in \mathcal{N}_n} \sum_{i=0}^{N-1} \mathbb{E}\left|\mathbb{E}\left[\tilde{Z}_{t_n} \mid X^\pi_{t_n}, Y^\pi_{t_n}\right] - \phi_n\left(X^\pi_{t_n}, Y^\pi_{t_n}\right)\right|^2 \frac{h}{0.052802^2}\right\}. \end{aligned} \tag{33}$$

As shown in Table 1, as the temporal discretization step size increases and h decreases, the approximation error associated with temporal discretization reduces. In this process, it is observed that the XNet, with its superior approximation capabilities, yields higher accuracy. However, the implementation of the FNNs fails to result in further improvement in the computational results. This phenomenon can be attributed to XNet providing smaller neural network approximation errors and optimization errors in the Deep BSDE method. As shown in Figure 3, when the temporal discretization reaches 80 time steps, the results indicate that the Deep BSDE method implemented with the XNet results in a faster decrease in the loss function, higher computational efficiency, and greater accuracy.

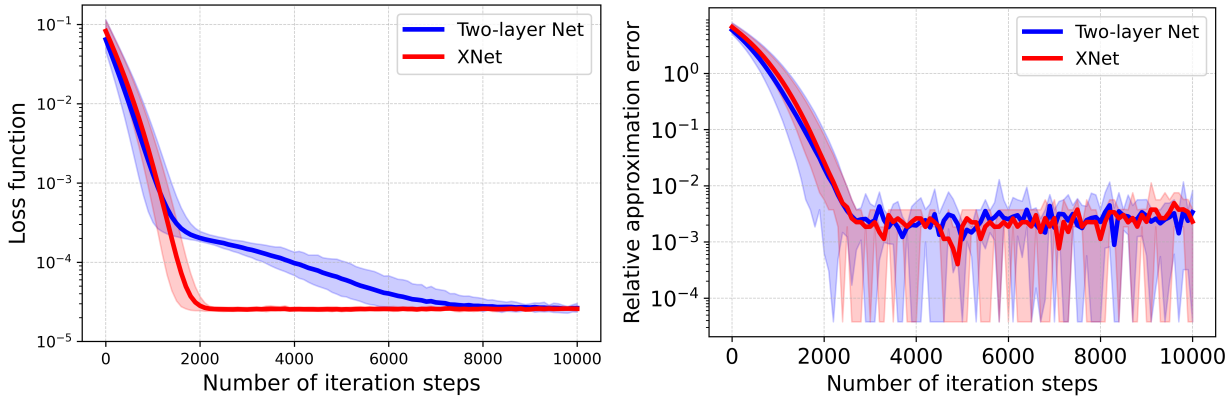

Figure 2: Comparison of Two Network Architectures for Solving the Allen-Cahn Equation under 20-step-time Discretization

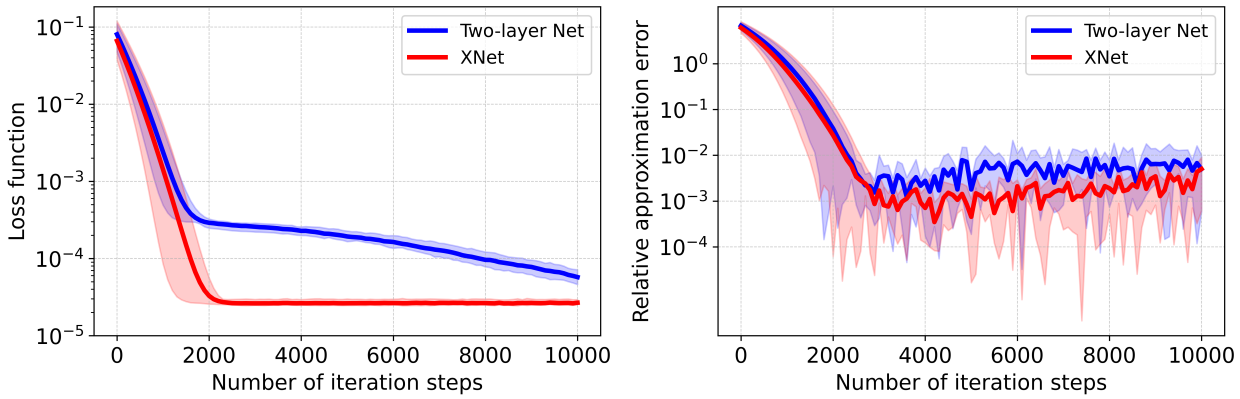

Figure 3: Comparison of Two Network Architectures for Solving the Allen-Cahn Equation under 80-step-time discretization

4.2 Pricing of European financial derivatives with different interest rates for borrowing and lending (PricingDiffrate) equation

In this example, we consider a specialized nonlinear Black-Scholes equation. This equation models the pricing problem of European financial derivatives in a financial market where the risk-free bank account utilized for hedging purposes exhibits differential interest rates for borrowing and lending [3]. Referring to the general form of the semi-linear parabolic PDE (1), we set $\bar{\mu} = 0.06$, $\mu(t,x) = \bar{\mu}x$, $\bar{\sigma} = 0.2$, $\sigma(t,x) = \bar{\sigma}x$. We assume for all $s,t \in [0,T]$, $x = (x_1,\ldots,x_d) \in \mathbb{R}^d$, $y \in \mathbb{R}$, and $z \in \mathbb{R}^d$, with $d = 100$, $T = 1/2$, and $\xi = (100,100,\ldots,100) \in \mathbb{R}^d$. Additionally, a terminal condition $g(x)$ and a non-linear term $f(t,x,y,z)$ are specified for the equation:

$$g(x) = \max\left\{[\max_{1\le i\le 100} x_i] - 120, 0\right\} - 2\max\left\{[\max_{1\le i\le 100} x_i] - 150, 0\right\}, \tag{34}$$

Table 1: Numerical Results for Allen-Cahn Equation

	XNet				Feedforward neural networks			
Time steps	Runtime (s)	Value function	Relative Error	Std. Deviation	Runtime (s)	Value function	Relative Error	Std. Deviation
20	72	5.2899e-02	1.8337e-03	5.7723e-05	83	5.2887e-02	1.6154e-03	6.1286e-05
30	112	5.2877e-02	1.4209e-03	7.4003e-05	138	5.2875e-02	1.3807e-03	9.6353e-05
40	196	5.2846e-02	8.3214e-04	6.5463e-05	260	5.2849e-02	8.8848e-04	1.5200e-04
80	464	5.2820e-02	3.4374e-04	4.8460e-05	691	5.2867e-02	1.2309e-03	1.7797e-04

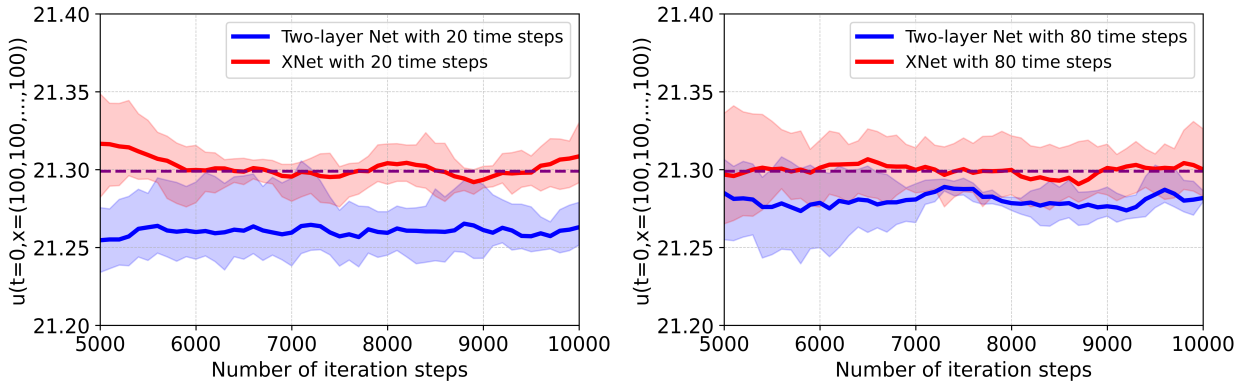

Figure 4: Comparison of Two Network Architectures for Solving the PricingDiffrate Equation under 20-step-time Discretization and 80-step-time Discretization

$$f(t,x,y,z) = -R^l y - \frac{(\bar{\mu}-R^l)}{\bar{\sigma}} \sum_{i=1}^{d} z_i + \left(R^b - R^l\right) \max\left\{0, \left[\tfrac{1}{\bar{\sigma}} \sum_{i=1}^{d} z_i\right] - y\right\}, \tag{35}$$

where $R^l = 0.04$, $R^b = 0.06$. The equation can thus be represented on the domain $t \in [0,T)$ and $x \in \mathbb{R}^d$:

$$\begin{aligned} &\frac{\partial u}{\partial t}(t,x) + \frac{\bar{\sigma}^2}{2} \sum_{i=1}^{d} |x_i|^2 \frac{\partial^2 u}{\partial x_i^2}(t,x) + \bar{\mu} \sum_{i=1}^{d} x_i \frac{\partial u}{\partial x_i}(t,x) \\ &\qquad + f\big(t,x,u(t,x),\bar{\sigma}\,\mathrm{diag}_{\mathbb{R}^{d\times d}}(x_1,\ldots,x_d)(\nabla_x u)(t,x)\big) = 0. \end{aligned} \tag{36}$$

The reference solution to equation (36) is obtained using the Multilevel-Picard approximation method, which yields a value of 21.299. The Deep BSDE method is implemented using both FNNs and XNet architectures with temporal discretization parameters $N = 20, 30, 40, 80$, conducting five independent runs for each configuration. The final computed value function represents the average over iterations 5000 to 10000. The results are presented in Table 2.

From equation (37), it is evident that the error introduced by temporal discretization is minimal, and the approximation error is almost exclusively related to the network's approximation capability and optimization errors.

$$\begin{aligned} \mathbb{E}\left|\frac{u(0,\xi)-\theta_{u_0}}{u(0,\xi)}\right|^2 \leq & C\left\{\frac{h}{21.299^2} + \inf_{\mu_0^\pi \in \mathcal{N}_0} \mathbb{E}\left|Z_0 - \theta_{\nabla u_0}\right|^2 \frac{h}{21.299^2}\right. \\ & \left. + \inf_{\phi_n^\pi \in \mathcal{N}_i} \sum_{i=0}^{N-1} \mathbb{E}\left|\mathbb{E}\left[\tilde{Z}_{t_n} \mid X_{t_n}^\pi, Y_{t_n}^\pi\right] - \phi_n\left(X_{t_n}^\pi, Y_{t_n}^\pi\right)\right|^2 \frac{h}{21.299^2}\right\}. \end{aligned} \tag{37}$$

Consequently, as shown in Table 2, increasing the number of temporal steps does not yield significant improvement in computational accuracy for either architecture. As shown in Table 2 and Figure 4, for the time-step discretization $N = 20, 30, 40, 80$, implementing the Deep BSDE method using the XNet rather than FNNs enhances computational speed and significantly improves accuracy. This improvement can be attributed to XNet providing smaller neural network approximation errors and optimization errors in the Deep BSDE method.

In fact, we have already achieved very good results. The reason we do not further refine the time discretization is due to concerns that a significant increase in network parameters would introduce non-negligible optimization errors. In the following section, we propose a continuous-time network architecture to eliminate the requirement for configuring a network at each temporal step. This approach addresses the challenge of significant parameter growth due to finer temporal discretization, thereby reducing optimization errors.

Table 2: Numerical Results for PricingDiffrate Equation

	XNet				Feedforward neural networks			
Time steps	Runtime (s)	Value function	Relative Error	Std. Deviation	Runtime (s)	value function	Relative Error	Std. Deviation
20	69	2.1306e+01	3.3219e-04	4.2208e-03	96	2.1260e+01	1.8144e-03	2.4084e-03
30	116	2.1302e+01	1.5146e-04	5.5465e-03	198	2.1279e+01	9.4557e-04	4.0696e-03
40	176	2.1303e+01	1.7159e-04	5.1489e-03	257	2.1278e+01	9.8029e-04	4.8573e-03
80	476	2.1304e+01	2.2257e-04	3.9802e-03	729	2.1280e+01	8.9073e-04	3.7753e-03

5 Continuous time models

In this paper, the computational error in deep learning algorithms is categorized into four primary components: the approximation error induced by temporal time discretization, the approximation error arising from neural network representation, the generalization error governed by the number of training samples, and the optimization error associated with the number of network parameters. When the computational error is dominated by the approximation error induced by temporal discretization, increasing the temporal steps is necessary to achieve higher computational accuracy. However, in discrete-time network architectures, this inevitably increases the number of network parameters, reducing computational efficiency and increasing optimization error. To address this issue, in this section, we apply the Deep BSDE method using both XNet and feedforward neural networks (FNNs) within continuous-time network architectures. In these continuous-time network architectures, the input layer includes an additional temporal dimension, resulting in a $d+1$-dimensional input. The output is the gradient function at each time step, which is d-dimensional. The difference lies in the fact that XNet has only one hidden layer (d-dimensional), while the FNNs comprise two hidden layers (each with $d+10$ dimensions).

5.1 Allen-Cahn Equation

In the previous section, when solving the Allen-Cahn equation (32) using discrete-time network architectures, it was observed that the accuracy of the algorithm improved with finer temporal discretization (Table 1). However, the increase in network parameters resulted in higher optimization errors. Here, continuous-time network architectures are adopted. First, XNet possesses sufficient approximation capability, suggesting that the approximation error arising from the neural network is minimal. Second, since XNet has relatively few parameters, we assume that the optimization error is also minimal. Third, we sampled $640,000$ independent trajectories, indicating that the generalization error is small. Under the assumption that the error in the Deep BSDE method is primarily dominated by the approximation error induced by temporal discretization, we can analyze the convergence rate of the Deep BSDE method with XNet.

Table 3: Numerical Results for Allen-Cahn Equation

	XNet					Feedforward neural networks				
Time steps	Runtime (s)	Value function	Relative Error	Error order	Std. Deviation	Runtime (s)	Value function	Relative Error	Error order	Std. Deviation
10	47	5.3020e-02	4.1212e-03		4.8446e-05	42	5.3020e-02	4.1269e-03		4.5878e-05
20	121	5.2906e-02	1.9607e-03	1.07	4.1509e-05	138	5.2907e-02	1.9917e-03	1.05	9.0286e-05
40	261	5.2842e-02	7.6389e-04	1.36	4.2562e-05	367	5.2833e-02	5.9284e-04	1.75	7.4642e-05
80	843	5.2810e-02	1.5496e-04	2.30	3.3253e-05	1452	5.2828e-02	4.9686e-04	0.26	7.4642e-05
160	2004	5.2805e-02	4.8824e-05	1.67	5.1628e-05	3728	5.2815e-02	2.4993e-04	0.99	1.4407e-04

Table 3 and Figure 5 demonstrate that the convergence order of the Deep BSDE method implemented using XNet is slightly less than second order. Notably, when the number of temporal steps reaches 40, FNNs no longer exhibit a discernible convergence order, whereas XNet maintains consistent convergence behavior. This can be attributed to the differential approximation capabilities: XNet possesses superior approximation capabilities, resulting in minimal neural network approximation errors and optimization errors, whereas FNNs lack this property.

With finer temporal discretization, the accuracy improved by both network implementations. The XNet achieves higher accuracy within shorter computational time and demonstrates greater robustness.

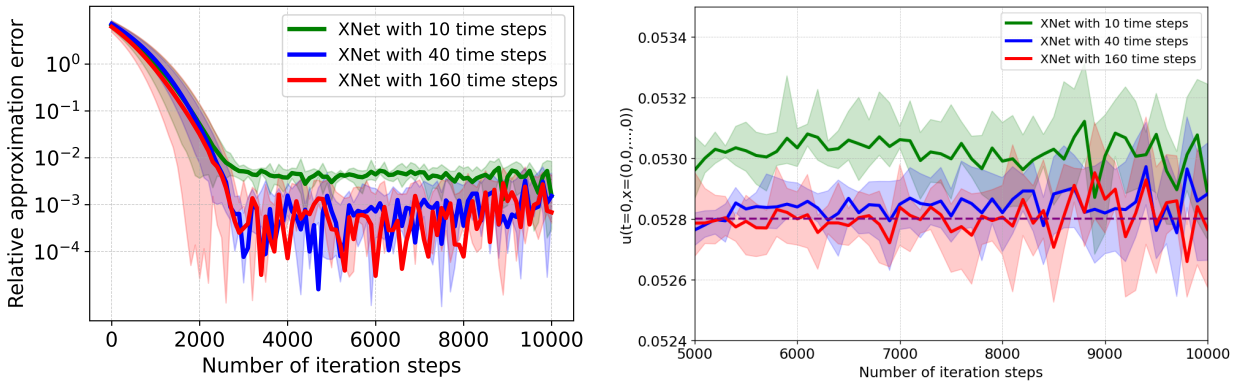

Figure 5: Results of solving the Allen-Cahn Equation using the Deep BSDE method by XNet under N-time-step discretization, with $N = 10$, 40, and 160.

5.2 Pricing of European financial derivatives with different interest rates for borrowing and lending (PricingDiffrate) equation

Table 4: Numerical Results for solving PricingDiffrate Equation

	XNet					Feedforward neural networks				
Time steps	Runtime (s)	Value function	Relative Error	Error order	Std. Deviation	Runtime (s)	Value function	Relative Error	Error order	Std. Deviation
10	55	2.1305e+01	2.6617e-04		2.7172e-03	96	2.1219e+01	3.7457e-03		3.2896e-03
20	70	2.1296e+01	1.5515e-04	0.78	2.5540e-03	142	2.1228e+01	3.3429e-03	0.16	2.9154e-03
40	151	2.1302e+01	1.2577e-04	0.30	3.5567e-03	273	2.1238e+01	2.8748e-03	0.22	2.5155e-03
80	506	2.1301e+01	9.7900e-05	0.36	1.4253e-03	729	2.1224e+01	3.5309e-03	-0.30	2.0813e-03
160	1558	2.1297e+01	9.0625e-05	0.11	2.3293e-03	2736	2.1215e+01	3.9331e-03	-0.16	2.0182e-03

The Deep BSDE method is also applied to solve the PricingDiffrate equation (36) with continuous-time implementations of the XNet and the FNNs. Table 4 and Figure 6 demonstrate that, regardless of the temporal discretization used ($N = 10, 20, 40, 80, 160$), the Deep BSDE method implemented with XNet consistently outperforms in terms of both computational speed and accuracy. These results align with the findings from the discrete-time models discussed in Section 4.

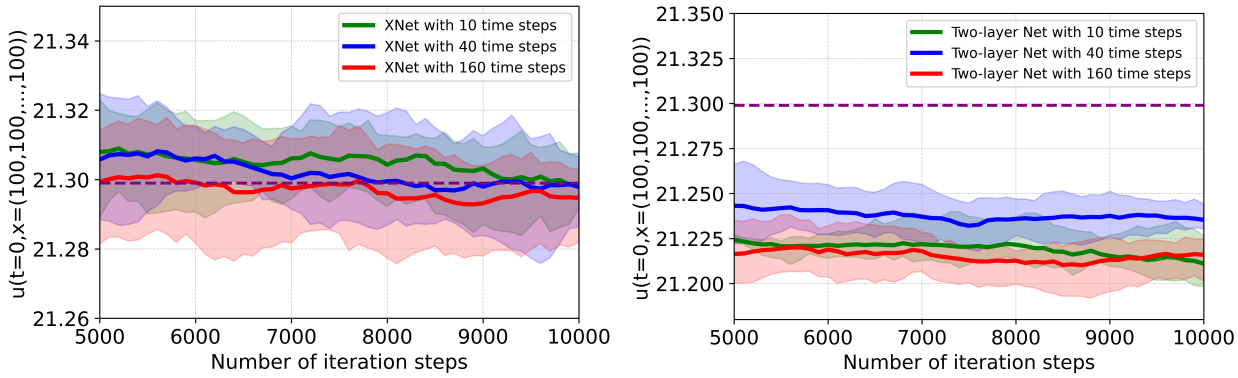

Figure 6: Comparison of Two Network Architectures for Solving the PricingDiffrate under 10-time-step and 160-time-step Discretization

Although introducing XNet allows the Deep BSDE method to achieve an accuracy approaching 9×10^{-5}, no clear error order is observed in this example. We speculate that the approximation error induced by temporal discretization is minimal, and that the computational error is likely dominated by the neural network's approximation capability or training error. To analyze the convergence rate, it is crucial to ensure that the approximation errors, optimization errors, and training errors are all minimized. To this end, we

increase the batch size to reduce the generalization errors, and increase the number of basis functions in XNet to enhance the neural network's approximation capability.

Table 5: Numerical Results for solving PricingDiffrate Equation by XNet

Steps	Batch size	Basis Functions	Runtime (s)	value function	Relative Error	Error order	Std. Deviation
20	64	100	70	2.1296e+01	1.5515e-04		2.5540e-03
20	256	100	98	2.1304e+01	2.4130e-04		4.4412e-03
10	64	200	44	2.1295e+01	2.0725e-04		3.6821e-03
20	64	200	98	2.1301e+01	8.9936e-05	1.20	3.0920e-03
40	64	200	207	2.1298e+01	6.4311e-05	0.48	1.2050e-03
80	64	200	670	2.1300e+01	3.2679e-05	0.98	2.5948e-03

As shown in Table 5 and Figure 7, with a time-step discretization of 20 or 80, we observe the following: On one hand, when the batch size reaches 256, the generalization error arising from the training samples no longer significantly contributes to the overall computational error. On the other hand, by increasing the number of basis functions in XNet, the accuracy improves further.

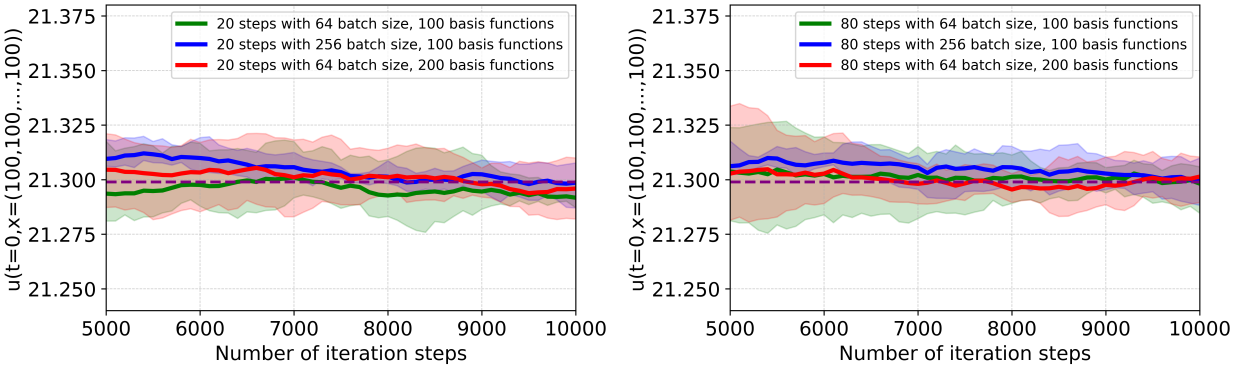

Figure 7: Solving the PricingDiffrate Equation by XNet under various settings with 20-step-time Discretization and 80-step-time Discretization.

As shown in Table 5, by increasing the number of basis functions in XNet to 200, we can clearly observe that the computational accuracy improves consistently as the time-step discretization increases. When the temporal discretization reaches 80, the relative error of the Deep BSDE method with XNet achieves 3.2679×10^{-5}, which significantly exceeds the accuracy achieved with feedforward neural networks (3.9331×10^{-3}).

6 Conclusion

This work establishes a comprehensive theoretical and computational framework for Deep BSDE methods applied to high-dimensional semilinear parabolic PDEs with non-Lipschitz generators. We have rigorously extended convergence theory beyond the classical Lipschitz framework to encompass Allen-Cahn equations with cubic nonlinearity and Hamilton-Jacobi-Bellman equations with quadratic gradient growth, providing the theoretical justification for the empirical success observed in these applications. The introduction of XNet architecture represents a significant computational advancement, achieving $\mathcal{O}(L)$ parameter complexity while maintaining superior approximation capabilities compared to traditional $\mathcal{O}(HL^2)$ feedforward networks. In discrete-time implementations, XNet demonstrates substantial improvements in both computational efficiency and solution accuracy across all tested configurations, with faster convergence and reduced relative errors for both Allen-Cahn and financial derivative pricing problems. The continuous-time framework reveals even more pronounced advantages: XNet significantly reduces neural network approximation errors and optimization errors compared to feedforward networks, enabling clearer observation of convergence behavior with rates approaching 1.6 for Allen-Cahn equations, while feedforward networks fail to maintain consistent convergence beyond 40 time steps. For the financial derivative pricing problem, XNet achieves relative errors of 3.27×10^{-5}, demonstrating markedly superior convergence orders compared to feedforward

implementations. The theoretical convergence guarantees, combined with the demonstrated computational advantages of XNet in both discrete and continuous-time settings, establish a robust foundation for tackling high-dimensional PDE problems in scientific computing, with immediate applications in mathematical finance, materials science, and optimal control theory.

References

[1] C. Beck, S. Becker, P. Cheridito, A. Jentzen, and A. Neufeld, Deep splitting method for parabolic pdes, SIAM Journal on Scientific Computing, 43 (2021), pp. A3135–A3154.

[2] C. BENDER and J. ZHANG, Time discretization and markovian iteration for coupled fbsdes, The Annals of Applied Probability, 18 (2008), pp. 143–177.

[3] Y. Z. Bergman, Option pricing with differential interest rates, The Review of Financial Studies, 8 (1995), pp. 475–500.

[4] B. Bouchard and N. Touzi, Discrete-time approximation and monte-carlo simulation of backward stochastic differential equations, Stochastic Processes and their applications, 111 (2004), pp. 175–206.

[5] W. Cai, S. Fang, W. Zhang, and T. Zhou, Martingale deep learning for very high dimensional quasi-linear partial differential equations and stochastic optimal controls, arXiv preprint arXiv:2408.14395, (2024).

[6] W. Cai, S. Fang, and T. Zhou, Soc-martnet: A martingale neural network for the hamilton-jacobi-bellman equation without explicit inf h in stochastic optimal controls, arXiv preprint arXiv:2405.03169, (2024).

[7] J.-F. Chassagneux and A. Richou, Numerical simulation of quadratic bsdes, The Annals of Applied Probability, (2016), pp. 262–304.

[8] Z. Chen, S.-K. Lai, and Z. Yang, At-pinn: Advanced time-marching physics-informed neural network for structural vibration analysis, Thin-Walled Structures, 196 (2024), p. 111423.

[9] A. Davey and H. Zheng, Deep neural network solver for hjb equations, in APCA International Conference on Automatic Control and Soft Computing, Springer, 2024, pp. 488–502.

[10] M. Gelbrich and W. Römisch, Numerical solution of stochastic differential equations (peter e. kloeden and eckhard platen), SIAM Review, 37 (1995), pp. 272–275.

[11] A. E. Gelfand, Gibbs sampling, Journal of the American statistical Association, 95 (2000), pp. 1300–1304.

[12] P. Grohs, F. Hornung, A. Jentzen, and P. Von Wurstemberger, A proof that artificial neural networks overcome the curse of dimensionality in the numerical approximation of Black–Scholes partial differential equations, vol. 284, American Mathematical Society, 2023.

[13] J. Han, A. Jentzen, and W. E, Solving high-dimensional partial differential equations using deep learning, Proceedings of the National Academy of Sciences, 115 (2018), pp. 8505–8510.

[14] J. Han, A. Jentzen, et al., Deep learning-based numerical methods for high-dimensional parabolic partial differential equations and backward stochastic differential equations, Communications in mathematics and statistics, 5 (2017), pp. 349–380.

[15] J. Han and J. Long, Convergence of the deep bsde method for coupled fbsdes, Probability, Uncertainty and Quantitative Risk, 5 (2020), p. 5.

[16] P. Henry-Labordere, Counterparty risk valuation: A marked branching diffusion approach, arXiv preprint arXiv:1203.2369, (2012).

[17] P. Henry-Labordère, N. Oudjane, X. Tan, N. Touzi, and X. Warin, Branching diffusion representation of semilinear pdes and monte carlo approximation, 55 1 ANNALES DE L'INSTITUT HENRI POINCARÉ PROBABILITÉS ET STATISTIQUES Vol. 55, No. 1 (February, 2019) 1–607, 55 (2019), pp. 184–210.

[18] P. Henry-Labordere, X. Tan, and N. Touzi, A numerical algorithm for a class of bsdes via the branching process, Stochastic Processes and their Applications, 124 (2014), pp. 1112–1140.

[19] W. Hofgard, J. Sun, and A. Cohen, Convergence of the deep galerkin method for mean field control problems, arXiv preprint arXiv:2405.13346, (2024).

[20] K. Hornik, M. Stinchcombe, and H. White, Multilayer feedforward networks are universal approximators, Neural networks, 2 (1989), pp. 359–366.

[21] C. Huré, H. Pham, and X. Warin, Deep backward schemes for high-dimensional nonlinear pdes, Mathematics of Computation, 89 (2020), pp. 1547–1579.

[22] P. Imkeller and G. Dos Reis, Path regularity and explicit convergence rate for bsde with truncated quadratic growth, Stochastic Processes and their Applications, 120 (2010), pp. 348–379.

[23] X. Ji, Y. Jiao, X. Lu, P. Song, and F. Wang, Deep ritz method for elliptical multiple eigenvalue problems, Journal of Scientific Computing, 98 (2024), p. 48.

[24] K. Katanforoosh, D. Kunin, and J. Ma, Parameter optimization in neural networks, 2019.

[25] X. Li, Z. Xia, and H. Zhang, Cauchy activation function and xnet, arXiv preprint arXiv:2409.19221, (2024).

[26] X. Li, X. Zheng, and Z. Xia, Enhancing neural function approximation: The xnet outperforming kan, arXiv preprint arXiv:2501.18959, (2025).

[27] S. Mishra and R. Molinaro, Estimates on the generalization error of physics-informed neural networks for approximating a class of inverse problems for pdes, IMA Journal of Numerical Analysis, 42 (2022), pp. 981–1022.

[28] E. Pardoux and S. Peng, Backward stochastic differential equations and quasilinear parabolic partial differential equations, in Stochastic Partial Differential Equations and Their Applications: Proceedings of IFIP WG 7/1 International Conference University of North Carolina at Charlotte, NC June 6–8, 1991, Springer, 2005, pp. 200–217.

[29] E. Pardoux and S. Tang, Forward-backward stochastic differential equations and quasilinear parabolic pdes, Probability theory and related fields, 114 (1999), pp. 123–150.

[30] M. Raissi, P. Perdikaris, and G. E. Karniadakis, Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations, Journal of Computational physics, 378 (2019), pp. 686–707.

[31] A. Richou, Markovian quadratic and superquadratic bsdes with an unbounded terminal condition, Stochastic Processes and their Applications, 122 (2012), pp. 3173–3208.

[32] Z. Shen, H. Yang, and S. Zhang, Neural network approximation: Three hidden layers are enough, Neural Networks, 141 (2021), pp. 160–173.

[33] Y. Shin, J. Darbon, and G. E. Karniadakis, On the convergence of physics informed neural networks for linear second-order elliptic and parabolic type pdes, arXiv preprint arXiv:2004.01806, (2020).

[34] J. Sirignano and K. Spiliopoulos, Dgm: A deep learning algorithm for solving partial differential equations, Journal of computational physics, 375 (2018), pp. 1339–1364.

[35] I. Soboĺ, Quasi-monte carlo methods, Progress in Nuclear Energy, 24 (1990), pp. 55–61.

[36] S. T. Tokdar and R. E. Kass, Importance sampling: a review, Wiley Interdisciplinary Reviews: Computational Statistics, 2 (2010), pp. 54–60.

[37] Y. Wang and L. Zhong, Nas-pinn: neural architecture search-guided physics-informed neural network for solving pdes, Journal of Computational Physics, 496 (2024), p. 112603.

[38] J. Xiao, F. Fu, and X. Wang, Deep learning based on randomized quasi-monte carlo method for solving linear kolmogorov partial differential equation, Journal of Computational and Applied Mathematics, (2024), p. 116088.

[39] B. Yu et al., The deep ritz method: a deep learning-based numerical algorithm for solving variational problems, Communications in Mathematics and Statistics, 6 (2018), pp. 1–12.

[40] J. Zhang, A numerical scheme for bsdes, The annals of applied probability, 14 (2004), pp. 459–488.

[41] J. Zhang, Backward stochastic differential equations, in Backward Stochastic Differential Equations: From Linear to Fully Nonlinear Theory, Springer, 2017, pp. 79–99.

Appendix A: Assumptions for Deep BSDE Method Convergence Analysis

This appendix provides the mathematical assumptions required for the convergence analysis of Deep BSDE methods under different generator conditions. For Lipschitz generators, the convergence analysis in Section 3.1 requires Assumptions 1 and 2. For Allen-Cahn type equations, the convergence analysis in Section 3.2 uses Assumptions 1 and 2, where the cubic nonlinearity is handled through boundedness properties established in Lemma 3. For HJB type equations, the convergence analysis in Section 3.3 requires Assumption 1 (with H_2 replaced by the conditions in Assumption 3) and Assumption 2.

We adopt the notation $\Delta x = x_1 - x_2$, $\Delta y = y_1 - y_2$, and $\Delta z = z_1 - z_2$. The constants μ_0, σ_0, and g_0 are non-negative real numbers representing the bounds at the origin or the intercept terms.

Assumption 1.

H_1. The coefficients μ, σ, g satisfy standard regularity and bounded conditions. There exist (possibly negative) constants k_μ such that

$$[\mu(t,x_1) - \mu(t,x_2)]^{\mathrm{T}} \Delta x \le k_\mu |\Delta x|^2.$$

there are non-negative constants K, σ_x, f_x, f_z, and g_x such that

$$\begin{aligned} |\mu(t,x_1) - \mu(t,x_2)|^2 \le K|\Delta x|^2, \quad |\mu(t,x)|^2 \le \mu_0 + K|x|^2, \\ |\sigma(t,x_1) - \sigma(t,x_2)|^2 \le \sigma_x|\Delta x|^2, \quad |\sigma(t,x)|^2 \le \sigma_0 + \sigma_x|x|^2 \\ |g(x_1) - g(x_2)|^2 \le g_x|\Delta x|^2, \quad |g(x)|^2 \le g_0 + g_x|x|^2. \end{aligned}$$

H_2. f is uniformly lipschitz continuous with respect to (x,y,z). In particular, There exist (possibly negative) constants k_f such that

$$[f(t,x,y_1,z) - f(t,x,y_2,z)]\,\Delta y \le k_f |\Delta y|^2.$$

There are non-negative constants $K, \mu_y, \sigma_x, \sigma_y$, f_x, f_z, and g_x such that

$$\begin{aligned} |f(t,x_1,y_1,z_1) - f(t,x_2,y_2,z_2)|^2 \le f_x|\Delta x|^2 + K|\Delta y|^2 + f_z|\Delta z|^2, \\ |f(t,x,y,z)|^2 \le f_0 + f_x|x|^2 + K|y|^2 + f_z|z|^2. \end{aligned}$$

H_3. μ, σ, f are uniformly Hölder-$\frac{1}{2}$ continuous with respect to t.

Assumption 2. *One of the following five cases holds:*

H_1. Small time duration, that is, T is small.

H_2. Weak coupling of Y into the forward SDE (2), that is, μ_y and σ_y are small. In particular, if $\mu_y = \sigma_y = 0$, then the forward equation does not depend on the backward one and, thus, Eqs. (2) and (3) are decoupled.

H_3. Weak coupling of X into the backward SDE (3), that is, f_x and g_x are small. In particular, if $f_x = g_x = 0$, then the backward equation does not depend on the forward one and, thus, Eqs. (2) and (3) are also decoupled. In fact, in this case, $Z = 0$ and (3) reduces to an ODE.

H_4. f is strongly decreasing in y, that is, k_f is very negative.

H_5. μ is strongly decreasing in x, that is, k_μ is very negative.

Assumption 3 (Quadratic Gradient Growth Framework)**.**

H_1^*. *For any $0 \le t \le T$, the functions $\mu(t,\cdot)$, $\sigma(t,\cdot)$ are differentiable and their derivatives are uniformly Lipschitz with Lipschitz constant K independent of t. In other words, $\sigma \in B_m^{m\times d}$ and $\mu \in B_m^{m\times 1}$. There exists a positive constant c such that*

$$y^T \sigma(t,x)\sigma^T(t,x) y \ge c|y|^2, \quad x, y \in \mathbb{R}^m,\ t \in [0,T]. \tag{38}$$

H_2^*. *For the generator $f(t,x,y,z)$ with quadratic growth, there exists $C_f \in R_+$ such that:*

$$|f(t,x,y_1,z_1) - f(t,x,y_2,z_2)| \le C_f|y_1 - y_2| + C_f(1 + |z_1| + |z_2|)|z_1 - z_2|, \tag{39}$$

$$|f(t,x_1,y,z) - f(t,x_2,y,z)| \le C_f(1 + |y| + |z|^2)|x_1 - x_2|, \tag{40}$$

$$|f(t,x,y,z)| \le C_f(1 + |y| + |z|^2). \tag{41}$$

Hypothesis above holds. f is differentiable in (x, y, z), and

$$\nabla_x |f(t, x, y, z)| \leq C_f(1 + |y| + |z|^2), \tag{42}$$
$$\nabla_y |f(t, x, y, z)| \leq C_f, \tag{43}$$
$$\nabla_z |f(t, x, y, z)| \leq C_f(1 + |z|). \tag{44}$$

H_3^*. *Moment boundedness: There exists $p > 2$ such that*

$$\sup_{0 \leq t \leq T} \mathbb{E}[|Y_t|^p + |Z_t|^p] < \infty.$$

H_4^*. *Neural network projection property: The approximation $Z_n^{\theta,\pi} = \varphi_N^\theta(Z_n^\pi)$ satisfies uniform bounds and projection regularity.*

Appendix B: Proof of Boundedness Properties for Double-Well Dynamics

This appendix provides the detailed proof of Lemma 3, which establishes the crucial boundedness properties for the double-well dynamics. The boundedness result is essential for proving convergence of the Deep BSDE method for Allen-Cahn equations in Section 3.2.

Proof. The BSDE (19) can be written in differential form as

$$dY_t = (Y_t^3 - Y_t)\, dt + Z_t\, dW_t.$$

Applying Itô's formula to Y_t^2, we obtain

$$d(Y_t^2) = 2Y_t\, dY_t + |Z_t|^2\, dt = \left(-2(Y_t^2 - Y_t^4) + |Z_t|^2\right) dt + 2Y_t Z_t\, dW_t.$$

Integrating from t to T yields

$$Y_T^2 - Y_t^2 = \int_t^T \left(-2(Y_s^2 - Y_s^4) + |Z_s|^2\right) ds + \int_t^T 2Y_s Z_s\, dW_s.$$

Taking expectations and using the martingale property of the stochastic integral, we obtain

$$\mathbb{E}[Y_t^2] = \mathbb{E}[Y_T^2] + \int_t^T \mathbb{E}\big(2(Y_s^2 - Y_s^4) - |Z_s|^2\big)\, ds.$$

Since for all $y \in \mathbb{R}$,

$$y^2 - y^4 = y^2(1 - y^2) \leq \tfrac{1}{4},$$

we deduce that

$$\mathbb{E}[Y_t^2] \leq \mathbb{E}[Y_T^2] + \frac{T - t}{2} \leq \mathbb{E}[g(X_T)^2] + \frac{T}{2}.$$

This proves the uniform L^2-boundedness of $(Y_t)_{t \in [0,T]}$. The bound on $\mathbb{E}|Y_t|$ follows immediately from the Cauchy–Schwarz inequality. □

Appendix C: Convergence for HJB-type equations

This appendix provides the complete convergence analysis for the Deep BSDE method applied to HJB-type equations with quadratic gradient growth generators. The main result establishes convergence rates under appropriate regularity assumptions.

We begin by establishing fundamental error bounds for the discrete approximation schemes.

Lemma 6 (Forward Process Error Bound)**.** *Assume that H_1 in Assumption 1 holds. Then the SDE*

$$dX_t = \mu(t, X_t)\, dt + \sigma(t, X_t)\, dW_t, \qquad X_0 = x \in \mathbb{R}^d,$$

admits a unique solution. Moreover, we have

$$\mathbb{E}|\delta X_n^\pi|^2 \le Ch,$$

where $\delta X_n^\pi = X_{t_n} - X_n^\pi$, and C is independent of h.

For the proof of this Lemma 6, we refer the reader to [7, 22, 10]. The core technical challenge lies in analyzing the convergence of the Deep BSDE system (13) to the truncated reference system (25). Before proving Theorem 3 and 4, we first introduce a proposition and a lemma.

Proposition 1 (Regularity results on (X, Y^B, Z^B); Propositions 3.1 and 3.2 in [7])**.** *Under the assumptions of Theorem 5, the following regularity bounds hold:*

(Y-component) *For all $p \le 1$:*

$$\sup_{0\le j\le N-1} \mathbb{E}\left[\sup_{t_j\le s\le t_{j+1}} |Y_s^B - Y_{t_j}^B|^{2p}\right] \le C_p h^p. \tag{45}$$

(Z-component) *For all $p \ge 1$:*

$$\sum_{n=0}^{N-1} \mathbb{E}\left[\left(\int_{t_n}^{t_{n+1}} |Z_s^B - \bar{Z}_n^B|^2 ds\right)^p\right] \le C_p h^p, \tag{46}$$

where $\bar{Z}_n^B = \frac{1}{h}\int_{t_n}^{t_{n+1}} Z_s^B ds$.

Lemma 7. *For any $0 \le n \le N-1$, we have*

$$\mathbb{E}_n\left|\tilde{Z}_n^\pi - \bar{Z}_n^B\right| \;\le\; Ch^{1/2}.$$

Proof. By definition,

$$\bar{Z}_n^B := \frac{1}{h}\,\mathbb{E}_n\left[\int_{t_n}^{t_{n+1}} Z_s^B ds\right], \qquad \tilde{Z}_n^\pi := \mathbb{E}_n\left[\frac{1}{h} Y_{t_{n+1}}^B\,(W_{n+1} - W_n)\right]. \tag{47}$$

thanks to assumptions on f^B (f^B is B-Lipschitz-continuous with respect to z), and Cauchy–Schwarz inequality, for $n \le N$,

$$\begin{aligned} h\mathbb{E}_n\big[\,|\tilde{Z}_n^\pi - \bar{Z}_n^B|^2\,\big] &= h\mathbb{E}_n\left[\left|\mathbb{E}_n\left[\int_{t_n}^{t_{n+1}} f^B(s, X_s, Y_s^B, Z_s^B)\, ds\;\frac{W_{t_{n+1}} - W_{t_n}}{h_n}\right]\right|^2\right] \\ &\le h\mathbb{E}_n\left[\int_{t_n}^{t_{n+1}} \left|f^B(s, X_s, Y_s^B, Z_s^B)\right|^2 ds\right] \\ &\le C\left(h^2 + (1+B^2)h\,\mathbb{E}_n\left[\int_{t_n}^{t_{n+1}} |Z_s^B|^2 ds\right]\right). \end{aligned}$$

□

Proof of Theorem 4. Define $\delta Y_n^\pi = \widetilde{Y}_n^\pi - Y_n^\pi$, $\delta Z_n^\pi = \widetilde{Z}_n^\pi - Z_n^{\theta,\pi}$. By the system (24) and (13), we have

$$\mathbb{E}[\delta Y_n^\pi] = \mathbb{E}_n\big[\delta Y_{n+1}^\pi + h\big(f(t_n, X_n^\pi, Y_n^\pi, Z_n^{\theta,\pi}) - f^B(t_n, X_n^\pi, \widetilde{Y}_n^\pi, \widetilde{Z}_n^\pi)\big)\big] + \mathbb{E}_n\big[\Upsilon_n^Y\big]. \tag{48}$$

and

$$\mathbb{E}\big|\delta Y_{n+1}^\pi\big| \le \mathbb{E}_n\big|\delta Y_n^\pi\big| + \mathbb{E}_n\big|f(t_n, X_n^\pi, Y_n^\pi, Z_n^{\theta,\pi}) - f^B(t_n, X_n^\pi, \widetilde{Y}_n^\pi, \widetilde{Z}_n^\pi)\big|h + \mathbb{E}_n\big|\Upsilon_n^Y\big|. \tag{49}$$

The key technical challenge lies in controlling the perturbation term $\mathbb{E}_n|\Upsilon_n^Y|$. We decompose:

$$\mathbb{E}_n|\Upsilon_n^Y| = \mathbb{E}_n|I_n^1| + \mathbb{E}_n|I_n^2| + \mathbb{E}_n|I_n^3|, \tag{50}$$

$$\mathbb{E}_n|I_n^1| = \mathbb{E}_n\left|\int_{t_n}^{t_{n+1}} \left(f^B(s, X_s, Y_s^B, Z_s^B) - f^B(t_n, X_n^\pi, Y_s^B, Z_s^B)\right)ds\right|, \tag{51}$$

$$\mathbb{E}_n|I_n^2| = \mathbb{E}_n\left|\int_{t_n}^{t_{n+1}} \left(f^B(t_n, X_n^\pi, Y_s^B, Z_s^B) - f^B(t_n, X_n^\pi, Y_{t_n}^B, Z_s^B)\right)ds\right|, \tag{52}$$

$$\mathbb{E}_n|I_n^3| = \mathbb{E}_n\left|\int_{t_n}^{t_{n+1}} \left(f^B(t_n, X_n^\pi, Y_{t_n}^B, Z_s^B) - f^B(t_n, X_n^\pi, Y_{t_n}^B, \widetilde{Z}_n^\pi)\right)ds\right|. \tag{53}$$

Applying the Lipschitz properties of f^B with respect to the spatial and temporal variables, combined with Lemma 6 and Proposition 1, we obtain

$$\mathbb{E}_n|I_1| \le C_1 h^{\frac{3}{2}}, \quad \mathbb{E}_n|I_2| \le C_2 h^{\frac{3}{2}},$$

for positive constants C_1 and C_2 independent of the discretization parameter h. The proof of I_3 constitutes the most technically demanding component, requiring careful analysis of the quadratic structure and projection properties.

$$\begin{aligned}
\mathbb{E}_n|I_3| &= \mathbb{E}_n\left|\int_{t_n}^{t_{n+1}} \left(f^B\big(t_n, X_n^\pi, Y_n^B, Z_s^B\big) - f^B\big(t_n, X_n^\pi, Y_n^B, \widetilde{Z}_n^\pi\big)\right) ds\right| \\
&\le \mathbb{E}_n\left|\int_{t_n}^{t_{n+1}} \left(f^B\big(t_n, X_n^\pi, Y_n^B, Z_s^B\big) - f^B\big(t_n, X_n^\pi, Y_n^B, \bar{Z}_n^B\big)\right) ds\right| \\
&\quad + \mathbb{E}_n\left|\int_{t_n}^{t_{n+1}} \left(f^B\big(t_n, X_n^\pi, Y_n^B, \bar{Z}_n^B\big) - f^B\big(t_n, X_n^\pi, Y_n^B, \widetilde{Z}_n^\pi\big)\right) ds\right| \\
&\le C_f\left[\int_{t_n}^{t_{n+1}} \mathbb{E}_n\big|Z_s^B - \bar{Z}_n^B\big|^2 ds + \int_{t_n}^{t_{n+1}} \mathbb{E}_n\left|\bar{Z}_n^B - \widetilde{Z}_n^\pi\right|^2 ds\right].
\end{aligned} \tag{54}$$

Exploiting the truncated Lipschitz properties of f^B and applying the regularity estimates from Proposition 1 and Lemma 7, we derive

$$\mathbb{E}_n|I_3| \le C_3 h^{\frac{3}{2}},$$

where the constant C_3 is independent of h. consequently, the error evolution satisfies

$$\begin{aligned}
\mathbb{E}\big|\delta Y_{n+1}^\pi\big| &\le \mathbb{E}_n|\delta Y_n^\pi| + \mathbb{E}_n\Big|f(t_n, X_n^\pi, Y_n^\pi, Z_n^{\theta,\pi}) - f^B(t_n, X_n^\pi, \widetilde{Y}_n^\pi, \widetilde{Z}_n^\pi)\Big|\, h + \mathbb{E}_n\big|\Upsilon_n^Y\big| \\
&\le (1 + C_f h)\,\mathbb{E}_n|\delta Y_n^\pi| + C_f\,\mathbb{E}_n\left[\left|Z_n^{\theta,\pi} - \widetilde{Z}_n^\pi\right|^2\right] h + O(h^{3/2})
\end{aligned} \tag{55}$$

By applying the discrete Gronwall inequality, for $n \le N-1$, we obtain:

$$\mathbb{E}\big|\delta Y_{n+1}^\pi\big| \le e^{C_f T}\left(\mathbb{E}_0|Y_0 - \theta_{u_0}| + \sum_{k=0}^{n} \mathbb{E}_k\left[\left|Z_k^{\theta,\pi} - \widetilde{Z}_k^\pi\right|^2\right] h\right) + O(h^{1/2}). \tag{56}$$

□

Appendix D: A posteriori estimation of the simulation error for HJB-type equation

Lemma 8. *Let $0 \le s_1 < s_2$, given $Q \in L^2(\Omega, \mathcal{F}_{s_2}, \mathbb{P})$, by the martingale representation theorem, there exists an $\mathcal{F}_t$-adapted process $\{H_s\}_{s_1 \le s \le s_2}$ such that $\int_{s_1}^{s_2} E|H_s|^2\, ds < \infty$ and $Q = E[Q|\mathcal{F}_{s_1}] + \int_{s_1}^{s_2} H_s\, dW_s$. Then we have $E[Q(W_{s_2} - W_{s_1})|\mathcal{F}_{s_1}] = E[\int_{s_1}^{s_2} H_s\, ds|\mathcal{F}_{s_1}]$.*

The lemma is from [15].
Rewite (25), we have

$$\widetilde{Y}_{n+1}^{\pi} = \widetilde{Y}_{n}^{\pi} - \int_{t_n}^{tn+1} f^B(s, X_s, Y_s^B, Z_s^B)\, ds + \int_{t_n}^{tn+1} Z_s^B\, dW_s \tag{57}$$

we apply Lemma 8 to (47), (57) and get

$$\tilde{Z}_n^{\pi} = \frac{1}{h}\mathbb{E}\left[\int_{t_n}^{t_{n+1}} Z_t^B dt | \mathcal{F}_n\right],$$

which implies, by the Cauchy inequality,

$$\begin{aligned}
\mathbb{E}|\widetilde{Z}_n^{\pi} - Z_n^{\theta,\pi}|^2 h &= \sum_{k=1}^{d} \mathbb{E}|(\widetilde{Z}_n^{\pi} - Z_n^{\theta,\pi})_k|^2 h = \sum_{k=1}^{d} \frac{1}{h}\mathbb{E}\left|\mathbb{E}\left[\int_{t_n}^{t_{n+1}} (Z_t^B - Z_n^{\theta,\pi})_k\, dt \Big| \mathcal{F}_{t_n}\right]\right|^2 \\
&\le \sum_{k=1}^{d} \frac{1}{h} E\left|\int_{t_n}^{t_{n+1}} (Z_t^B - Z_n^{\theta,\pi})_k\, dt\right|^2 \le \sum_{k=1}^{d} \int_{t_n}^{t_{n+1}} E|(Z_t^B - Z_n^{\theta,\pi})_k|^2\, dt \\
&= \int_{t_n}^{t_{n+1}} E|(Z_t^B - Z_n^{\theta,\pi})|^2\, dt,
\end{aligned} \tag{58}$$

Proof of Theorem 3. Define $\delta Y_n^{\pi} = \widetilde{Y}_n^{\pi} - Y_n^{\pi}$,

$$\delta Y_{n+1}^{\pi} = \delta Y_n^{\pi} - \Big(f^B(t_n, X_n^{\pi}, \widetilde{Y}_n^{\pi}, \widetilde{Z}_n^{\pi}) - f(t_n, X_{t_n}^{\pi}, Y_{t_n}^{\pi}, Z_{t_n}^{\theta,\pi})\Big)h + \Big(\int_{t_n}^{t_{n+1}} Z_s^B - Z_{t_n}^{\theta,\pi} dW_s\Big) + \Upsilon_n^Y, \tag{59}$$

where Υ_n^Y is from equation (26). From equation (59), by H_1, H_3 in Assumption 1, Assumption 2 Assumption 3, and the root-mean square and geometric mean inequality (RMS-GM inequality), for any $\lambda_1 > 0$, we have

$$\begin{aligned}
\mathbb{E}|\delta Y_{n+1}^{\pi}|^2 = \mathbb{E}|\delta Y_n^{\pi}|^2 &+ \mathbb{E}\big[|f^B(t_n, X_n^{\pi}, \widetilde{Y}_n^{\pi}, \widetilde{Z}_n^{\pi}) - f^B(t_n, X_n^{\pi}, Y_n^{\pi}, \widetilde{Z}_n^{\pi})|^2\big]h^2 + \mathbb{E}|\Upsilon_n^Y|^2 + \int_{t_n}^{t_{n+1}} \mathbb{E}|Z_s^B - Z_n^{\theta,\pi}|^2\, ds \\
&- 2\,\mathbb{E}\big[(f^B(t_n, X_n^{\pi}, \widetilde{Y}_n^{\pi}, \widetilde{Z}_n^{\pi}) - f^B(t_n, X_n^{\pi}, Y_n^{\pi}, \widetilde{Z}_n^{\pi})\delta Y_n^{\pi}\big]h - 2\mathbb{E}|\Upsilon_n^Y \delta Y_n^{\pi}| \\
&- 2\,\mathbb{E}\big[(f^B(t_n, X_n^{\pi}, \widetilde{Y}_n^{\pi}, \widetilde{Z}_n^{\pi}) - f^B(t_n, X_n^{\pi}, Y_n^{\pi}, \widetilde{Z}_n^{\pi})\Upsilon_n^Y\big]h \\
\ge\ \mathbb{E}|\delta Y_n^{\pi}|^2 &+ \int_{t_n}^{t_{n+1}} \mathbb{E}|Z_s^B - Z_{t_n}^{\theta,\pi}|^2\, ds - 2\mathbb{E}\Big[\Big(f^B(t_n, X_n^{\pi}, \widetilde{Y}_n^{\pi}, \widetilde{Z}_n^{\pi}) - f^B(t_n, X_n^{\pi}, Y_n^{\pi}, \widetilde{Z}_n^{\pi})\Big)\delta Y_n^{\pi}\Big]h \\
&- 2\mathbb{E}\Big[\Big(f^B(t_n, X_n^{\pi}, Y_n^{\pi}, \widetilde{Z}_n^{\pi}) - f(t_n, X_n^{\pi}, Y_n^{\pi}, Z_n^{\theta,\pi})\Big)\delta Y_n^{\pi}\Big]h + O(h^{3/2}) \\
\ge\ \mathbb{E}|\delta Y_n^{\pi}|^2 &+ \int_{t_n}^{t_{n+1}} \mathbb{E}|Z_s^B - Z_{t_n}^{\theta,\pi}|^2\, ds - 2C_f\mathbb{E}|\delta Y_n^{\pi}|^2 h \\
&- \Big[\lambda_1 \mathbb{E}|\delta Y_n^{\pi}|^2 + \frac{1}{\lambda_1}\Big(C_f(1 + |\varphi^B(\widetilde{Z}_n^{\pi})|^2 + |Z_n^{\theta,\pi}|^2)|\widetilde{Z}_n^{\pi} - Z_n^{\theta,\pi}|^2\Big)\Big]h + O(h^{3/2}).
\end{aligned} \tag{60}$$

Plugging it into (58) gives us

$$\mathbb{E}|\delta Y_{n+1}^{\pi}|^2 \ge [1 - (2C_f + \lambda_1)h]\,\mathbb{E}|\delta Y_n^{\pi}|^2 + \left[1 - \frac{C_f}{\lambda_1}(1 + 2B^2)\right] hE|\delta Z_n^{\pi}|^2 + O(h^{3/2}). \tag{61}$$

Then there exists a constant C, for any $\lambda_1 \ge C_f(1+2B^2)$ and sufficiently small h satisfying $(2C_f + \lambda_1)h < 1$, we have

$$\begin{aligned}
\mathbb{E}|\delta Y_n^{\pi}|^2 &\le e^{-h^{-1}ln[1-(2C_f+\lambda_1)h](N-n)h}\left[\mathbb{E}|\widetilde{Y}_N^{\pi} - Y_N^{\pi}|^2 + O(h^{1/2})\right] \\
&\le C\left[\mathbb{E}|\widetilde{Y}_N^{\pi} - Y_N^{\pi}|^2 + O(h^{1/2})\right].
\end{aligned}$$

□

Proof of Theorem 5. Combine Lemma 4 and Theorem 3, there exists a constant $C_1' > 0$ such that

$$\begin{aligned}\mathbb{E}|Y_{t_n} - Y_n^\pi|^2 &\le e^{-h^{-1}ln[1-(2C_f+\lambda_1)h](N-n)h}\left[\mathbb{E}|g(X_T^\pi) - Y_N^\pi|^2 + O(h^{1/2}) + \mathbb{E}|g(X_T^\pi) - Y_T^B|^2\right] + \mathbb{E}|Y_{t_n} - Y_{t_n}^B|^2 \\ &= e^{-h^{-1}ln[1-(2C_f+\lambda_1)h](N-n)h}\left[\mathbb{E}|g(X_T^\pi) - Y_N^\pi|^2 + O(h^{1/2}) + C_pD_\beta B^{-\frac{\beta}{2\bar{q}}}\right] + C_pD_\beta B^{-\frac{\beta}{2\bar{q}}} \\ &\le C_1'(h^{1/2} + \mathbb{E}|g(X_T^\pi) - Y_N^\pi|^2 + C_pD_\beta B^{-\frac{\beta}{2\bar{q}}}).\end{aligned}$$

Specially,

$$\mathbb{E}|u(0,\xi) - \theta_{u_0}|^2 \le C_1'(h^{1/2} + \mathbb{E}|g(X_T^\pi) - Y_N^\pi|^2 + C_pD_\beta B^{-\frac{\beta}{2\bar{q}}})\cdot$$

Combine Lemma 4 and Theorem 4, there exists a constant $C_2' > 0$ such that

$$\begin{aligned}\sup_{0\le n\le N}\mathbb{E}\left|Y_{t_n} - Y_n^\pi\right| &\le \sup_{0\le n\le N}\mathbb{E}\left|\delta Y_n^\pi\right| + \sup_{0\le n\le N}\mathbb{E}\left|Y_{t_n}^B - Y_{t_n}\right| \\ &\le C_2'\left(h^{\frac{1}{2}} + \mathbb{E}\left|Y_0 - \theta_{u_0}\right|^2 + \sum_{n=0}^{N-1}\mathbb{E}\left|Z_n^{\theta,\pi} - \tilde{Z}_n^\pi\right|^2 h\right) + \sqrt{C_pD_\beta}B^{-\frac{\beta}{4q}}.\end{aligned} \tag{62}$$

Specially,

$$\mathbb{E}\left|g(X_T^\pi) - Y_N^\pi\right| \le C_2'\left(h^{\frac{1}{2}} + \mathbb{E}\left|Y_0 - \theta_{u_0}\right|^2 + \sum_{n=0}^{N-1}\mathbb{E}\left|\widetilde{Z}_n^\pi - Z_n^{\theta,\pi}\right|^2 h\right) + \sqrt{C_pD_\beta}B^{-\frac{\beta}{4q}}. \tag{63}$$

After the neural network $\phi_n^\theta\left(t_n, X_{t_n}^\pi\right)$ has been sufficiently trained, we obtain

$$\begin{aligned}\inf_{\theta_{u_0},\theta_{\nabla u_0}\in\mathcal{N}_0,\phi_n\in\mathcal{N}_n}\mathbb{E}\left|g(X_T^\pi) - Y_N^\pi\right| \le C_2'\Bigg(&h^{\frac{1}{2}} + \inf_{\theta_{u_0},\theta_{\nabla u_0}\in\mathcal{N}_0}\mathbb{E}\left|Y_0 - \theta_{u_0}\right|^2 + \mathbb{E}\left|\widetilde{Z}_0^\pi - Z_0^{\theta,\pi}\right|^2 h \\ &+ \inf_{\phi_n^\theta\in\mathcal{N}_n}\sum_{n=1}^{N-1}\mathbb{E}\left|\widetilde{Z}_n^\pi - Z_n^{\theta,\pi}\right|^2 h\Bigg) + \sqrt{C_pD_\beta}B^{-\frac{\beta}{4q}}.\end{aligned} \tag{64}$$

□