Bergische Universität Wuppertal

Fakultät für Mathematik und Naturwissenschaften

Institute of Mathematical Modelling, Analysis and Computational Mathematics (IMACM)

Preprint BUW-IMACM 23/15

Lorenc Kapllani, Long Teng and Matthias Rottmann

Uncertainty quantification for deep learning-based schemes for solving high-dimensional backward stochastic differential equations

October 10, 2023

http://www.imacm.uni-wuppertal.de

Uncertainty quantification for deep learning-based schemes for solving high-dimensional backward stochastic differential equations

Lorenc Kapllani, Long Teng and Matthias Rottmann

Chair of Applied and Computational Mathematics,
Faculty of Mathematics and Natural Sciences,
University of Wuppertal,
Gaußstr. 20, 42119 Wuppertal, Germany

Abstract

Deep learning-based numerical schemes for solving high-dimensional backward stochastic differential equations (BSDEs) have recently raised plenty of scientific interest. While they enable numerical methods to approximate very high-dimensional BSDEs, their reliability has not been studied and is thus not understood. In this work, we study uncertainty quantification (UQ) for a class of deep learning-based BSDE schemes. More precisely, we review the sources of uncertainty involved in the schemes and numerically study the impact of different sources. Usually, the standard deviation (STD) of the approximate solutions obtained from multiple runs of the algorithm with different datasets is calculated to address the uncertainty. This approach is computationally quite expensive, especially for high-dimensional problems. Hence, we develop a UQ model that efficiently estimates the STD of the approximate solution using only a single run of the algorithm. The model also estimates the mean of the approximate solution, which can be leveraged to initialize the algorithm and improve the optimization process. Our numerical experiments show that the UQ model produces reliable estimates of the mean and STD of the approximate solution for the considered class of deep learning-based BSDE schemes. The estimated STD captures multiple sources of uncertainty, demonstrating its effectiveness in quantifying the uncertainty. Additionally, the model illustrates the improved performance when comparing different schemes based on the estimated STD values. Furthermore, it can identify hyperparameter values for which the scheme achieves good approximations.

Keywords *backward stochastic differential equations, high-dimensional problems, deep neural networks, uncertainty quantification, heteroscedastic nonlinear regression*

1 Introduction

Deep learning has attracted the interest of academics in various fields, including computer vision and natural language processing, as well as traditional sciences such as physics, chemistry, biology, and finance. Since the predictions of models that use deep learning in decision-making processes are prone to noise and model errors [27], assessing the model's reliability before it can be used in practice is critical. An example of such decisions is the pricing and hedging of different contracts in finance. Companies may suffer from significant financial losses as a result of poor judgments. Thus, it is highly desirable to understand the uncertainties in deep learning-based numerical schemes and develop methods to quantify them.

Backward stochastic differential equations (BSDEs) are important tools used to model problems in scientific fields due to their connections to partial differential equations (PDEs) and stochastic control problems via the nonlinear Feynman–Kac formula [32]. In most cases, nonlinear BSDEs

cannot be solved explicitly. Hence, advanced numerical techniques to approximate their solutions become desired. Over the years, many numerical methods have been proposed for solving BSDEs, e.g., [6, 52, 18, 37, 53, 5, 38, 55, 17, 8, 54, 46, 45]. However, most of those methods are not suitable for solving high-dimensional BSDEs due to the curse of dimensionality. Some techniques such as parallel computing using GPU computing [19, 30] or sparse grid methods [51, 12, 7] can only solve moderate dimensional BSDEs within a reasonable computational time.

Recently, the authors in [11, 20] developed a deep learning-based scheme (we refer to it as the DBSDE scheme in the rest of the paper) for solving high-dimensional nonlinear BSDEs. The method employs deep neural networks (DNNs) to approximate the unknown gradient of the solution by reformulating the BSDE problem as a stochastic optimization problem. Due to the universal approximation capability [23, 9] of neural networks (NNs), the objective function can be effectively optimized in practice. Therefore, the function values of interest (the unknown solution and its gradient) are obtained quite accurately. The method has opened the door to solve such problems in hundreds of dimensions within a reasonable computational time. After the DBSDE scheme, there are several works proposed to improve it, e.g., by algorithmic adjustments or methodological extensions [44, 13, 25, 3, 48, 15, 29, 31]. Furthermore, some convergence analysis of the DBSDE scheme have also been studied, e.g., see [21, 28] for the error analysis (utilizing universal approximation capability of NNs) and [49] for its gradient convergence (under a restrictive choice of NN setting).

As an example, we consider the pioneering DBSDE scheme to study uncertainty quantification (UQ) and introduce our UQ model. In the DBSDE scheme, the authors reformulate the BSDE as a stochastic control problem and use the Euler-Maruyama method to discretize the integrals. The unknown functions (solution at initial time and its gradient on the whole time domain) are estimated using DNNs. The parameters of DNNs are then optimized using the stochastic gradient descent (SGD) algorithm. The DBSDE method incorporates various sources of uncertainty, including finite time discretization, restrictive choice of DNN specifications, the lack of convergence guarantees of the SGD algorithm, and finite sample size during stochastic optimization. It is crucial to identify and quantify these different sources of uncertainty in the DBSDE scheme for practical applications. Therefore, we review these sources of uncertainty and numerically demonstrate the impact of different sources. Our numerical experiments show that it is practically challenging to disentangle them, emphasizing the importance of quantifying uncertainty before using the scheme in practice. Usually, the standard deviation (STD) of the approximate solutions obtained from multiple runs of the DBSDE algorithm with different datasets is calculated to account for the uncertainty in a given prediction, see [11]. This approach is computationally expensive, especially in high-dimensional cases. Therefore, we develop a UQ model to estimate the STD of the approximate solution without requiring multiple runs. Several techniques have been proposed in the literature to quantify uncertainty, such as Monte Carlo Dropout [14], Monte Carlo DropConnnect [39], deep ensembles [36, 43], Flipout-based variational inference [50], Markov Chain Monte Carlo [35], and many others [22, 42, 41]. A review of these techniques can be found in [1]. To the best of our knowledge, there are no applications or developments of UQ models specifically for deep learning-based BSDE schemes. Hence, we develop a UQ model with the aim of addressing this gap.

Our UQ model is based on a commonly used approach for quantifying uncertainty in heteroscedastic nonlinear regression, see [2] for heteroscedastic least square type regression methods and [40, 36, 33] for heteroscedastic NN regression methods. We make the assumption that the residuals or errors of the DBSDE scheme follow a normal distribution with zero mean and the STD depending on the chosen parameter set of the discretized BSDE. This is a standard assumption in heteroscedastic regression. In our method, we use a DNN to learn two functions that estimate the mean and STD of the approximate solution. To train the DNN, we construct a dataset of independent and identically distributed (i.i.d) samples, which includes different

parameter sets of the discretized BSDE and the corresponding approximate solutions obtained from the DBSDE algorithm. After generating a moderate number of samples, we optimize the network parameters by minimizing the negative log-likelihood. Our UQ model provides an estimate of the STD of the approximate solution in a more computationally efficient manner compared to running multiple iterations of the algorithm per parameter set. Additionally, the estimated mean of the approximate solution from our model can be used to initialize the algorithm and improve the optimization process. Note that the proposed UQ model is applicable not only to the DBSDE scheme but also to other deep learning-based schemes for solving BSDEs. In addition to the DBSDE scheme, we apply our UQ model to the Locally additive DBSDE (LaDBSDE) scheme [31], which exhibits better convergence than the DBSDE scheme. Our numerical experiments demonstrate that the UQ model produces reliable estimates of the mean and STD of the approximate solution for both the DBSDE and LaDBSDE schemes. Moreover, we show multiple practical implications of using the UQ model. Firstly, the estimated STD captures multiple sources of uncertainty, demonstrating its effectiveness in quantifying the uncertainty. Secondly, the UQ model illustrates the improved performance of the LaDBSDE scheme in comparison to the DBSDE scheme based on the corresponding estimated STD values. Finally, our UQ model can be utilized to determine DNN hyperparameter values for which the respective scheme performs well, e.g. the number of discretization points.
The remainder of the paper is organized as follows. In Section 2, we provide the necessary foundations to view the BSDE as a learning problem. In Section 3, we introduce the DBSDE scheme for solving high-dimensional BSDEs, discuss the sources of uncertainty in the scheme, and present a UQ model to estimate the STD of the approximate solution. In Section 4, we analyze the practical impact of different sources of uncertainty in the DBSDE scheme and demonstrate the effectiveness of our UQ model through numerical tests for both the DBSDE and LaDBSDE schemes. Finally, in Section 5, we conclude our work.

2 Backward stochastic differential equations as a learning problem

In the following, we introduce the notions of the BSDEs that are used throughout the paper.

2.1 Preliminaries

Let $(\Omega, \mathcal{F}, \mathbb{P}, \{\mathcal{F}_t\}_{0\leq t\leq T})$ be a complete, filtered probability space. In this space a standard d-dimensional Brownian motion W_t is defined, such that the filtration $\{\mathcal{F}_t\}_{0\leq t\leq T}$ is the natural filtration of W_t. Throughout the whole paper we rely on the following notations

- $|x|$ for the standard Euclidean norm of $x \in \mathbb{R}$ or $x \in \mathbb{R}^d$.
- $\mathbb{S}^2\left([0,T]\times\Omega;\mathbb{R}^d\right)$ for the space of continuous and progressively measurable stochastic processes $X : [0,T]\times\Omega \to \mathbb{R}^d$ such that $\mathbb{E}\big[\sup_{0\leq t\leq T}|X_t|^2\big] < \infty$.
- $\Delta = \{t_n | t_n \in [0,T], n = 0,1,\dots,N, t_n < t_{n+1}, \Delta t = t_{n+1} - t_n, t_0 = 0, t_N = T\}$ is a uniform discretization of the time interval $[0,T]$.
- $\mathbb{H}^2\left([0,T]\times\Omega;\mathbb{R}^{1\times d}\right)$ for the space of progressively measurable stochastic processes $Z : [0,T]\times\Omega \to \mathbb{R}^{1\times d}$ such that $\mathbb{E}\big[\int_0^T |Z_t|^2\,dt\big] < \infty$.
- $\mathbb{H}^{\Delta,2}\left(\{0,1,\dots,N\}\times\Omega;\mathbb{R}^{1\times d}\right)$ for the space of progressively measurable discrete stochastic processes $Z^\Delta : \{0,1,\dots,N\}\times\Omega \to \mathbb{R}^{1\times d}$ such that $\mathbb{E}\big[\sum_{n=0}^N |Z^\Delta_{t_n}|^2\,\Delta t\big] < \infty$.

- $\mathbb{L}^2_{\mathcal{F}_t}\left(\Omega;\mathbb{R}^d\right)$ for the space of $\mathcal{F}_t$-measurable random variables $\xi : \Omega \to \mathbb{R}^d$ such that $\mathbb{E}\left[|\xi|^2\right] < \infty$.
- $C^{1,2}\left([0,T]\times\mathbb{R}^d;\mathbb{R}\right)$ for the space of continuous and differentiable functions of two arguments $u : [0,T]\times\mathbb{R}^d \to \mathbb{R}$, differentiable with respect to the first argument and twice differentiable with respect to the second argument.
- $D^\top$ for the transpose of matrix $D \in \mathbb{R}^{d\times d}$.
- $\mathrm{Tr}\,[D]$ for the trace of matrix $D \in \mathbb{R}^{d\times d}$.

2.2 Nonlinear Feynman-Kac formula

We present the nonlinear Feynman–Kac formula, which provides a probabilistic representation for the solution of a semi-linear parabolic PDE through a (decoupled) forward-backward stochastic differential equation (FBSDE).
Let $T \in (0,\infty)$, $d \in \mathbb{N}$, the functions $f : [0,T]\times\mathbb{R}^d\times\mathbb{R}\times\mathbb{R}^{1\times d} \to \mathbb{R}$ and $g : \mathbb{R}^d \to \mathbb{R}$ are continuous, and $(a,b) : [0,T]\times\mathbb{R}^d \to \mathbb{R}^d\times\mathbb{R}^{d\times d}$ is continuously differentiable for all variables. Let $u \in C^{1,2}([0,T]\times\mathbb{R}^d;\mathbb{R})$ satisfy that $u(T,x) = g(x)$ and the semi-linear parabolic PDE

$$\frac{\partial u}{\partial t} + \nabla u(t,x)\, a(t,x) + \frac{1}{2}\,\mathrm{Tr}\left[bb^\top \,\mathrm{Hess}\, u(t,x)\right] + f\left(t,x,u,\nabla u\, b\right)(t,x) = 0, \tag{1}$$

for all $(t,x) \in ([0,T]\times\mathbb{R}^d)$, where $\mathrm{Hess}\, u$ is the Hessian matrix and ∇u the gradient of u with respect to the spatial variable x. Consider the following decoupled FBSDE

$$\begin{cases} X_t &= x_0 + \int_0^t a\left(s,X_s\right)\, ds + \int_0^t b\left(s,X_s\right)\, dW_s, \\ Y_t &= g\left(X_T\right) + \int_t^T f\left(s,X_s,Y_s,Z_s\right)\, ds - \int_t^T Z_s\, dW_s, \end{cases} \tag{2}$$

where $f : [0,T]\times\mathbb{R}^d\times\mathbb{R}\times\mathbb{R}^{1\times d} \to \mathbb{R}$ is the driver function, $f(t,X_t,Y_t,Z_t)$ is $\mathcal{F}_t$-adapted, the third term on the right-hand side is an Itô-type integral and $g : \mathbb{R}^d \to \mathbb{R}$ is the function for the terminal condition depending on the final value X_T of the forward stochastic differential equation (SDE). The triple of processes $\{(X_t,Y_t,Z_t)\}_{0\le t\le T} \in \mathbb{S}^2\left([0,T]\times\Omega;\mathbb{R}^d\right)\times\mathbb{S}^2\left([0,T]\times\Omega;\mathbb{R}\right)\times\mathbb{H}^2\left([0,T]\times\Omega;\mathbb{R}^{1\times d}\right)$ is the solution of the above FBSDE if it satisfies (2) $\mathbb{P}$-a.s. $\forall\, t \in [0,T]$. Assume that (1) has a classical solution $u \in C^{1,2}([0,T]\times\mathbb{R}^d;\mathbb{R})$ and the usual regularity conditions of (2) are satisfied [32]. Then the solution of (2) can be represented $\mathbb{P}$-a.s. by

$$Y_t = u\left(t,X_t\right), \quad Z_t = \nabla u\left(t,X_t\right) b\left(t,X_t\right) \quad \forall\, t \in [0,T)\,. \tag{3}$$

2.3 Formulation of the BSDE as a suitable stochastic control problem

We now view the solution (X_t,Y_t,Z_t) of (2) as the stochastic control problem

$$\begin{aligned} &\inf_{Y_0\in\mathbb{L}^2_{\mathcal{F}_0}(\Omega;\mathbb{R}),\, Z\in\mathbb{H}^2\left([0,T]\times\Omega;\mathbb{R}^{1\times d}\right)} \mathbf{L}\left(Y_0,Z\right), \\ s.t.\quad & X_t = x_0 + \int_0^t a\left(s,X_s\right)\, ds + \int_0^t b\left(s,X_s\right)\, dW_s, \\ & Y_t = Y_0 - \int_0^t f\left(s,X_s,Y_s,Z_s\right)\, ds + \int_0^t Z_s\, dW_s, \end{aligned} \tag{4}$$

for $t \in [0,T]$, where

$$\mathbf{L}\left(Y_0,Z\right) := \mathbb{E}\left[|g(X_T) - Y_T|^2\right].$$

The solution of (2) is a minimizer of (4) since the loss function attains zero when it is evaluated at the solution. In addition, the wellposedness of the BSDEs (under the usual regularity conditions [32]) ensures the existence and uniqueness of the minimizer. Due to (3), we seek a function approximator for $u : \mathbb{R}^d \to \mathbb{R}$ and $\nabla u\, b : [0,T] \times \mathbb{R}^d \to \mathbb{R}^{1\times d}$ to approximate the unknown solution $Y_0 = u(t_0, x_0)$ and $Z_t = \nabla u\,(t, X_t)\, b\,(t, X_t)\ \forall\ t \in [0,T]$. Due to their approximation capability in high dimensions, NNs are a promising candidate.

2.4 Neural networks as function approximators

For our purpose, we consider fully connected feedforward NNs or DNNs. Let $d_0, d_1 \in \mathbb{N}$ be the input and output dimensions, respectively. We fix the global number of layers as $L+2$, $L \in \mathbb{N}$ the number of hidden layers each with $\eta \in \mathbb{N}$ neurons. The first layer is the input layer with d_0 neurons and the last layer is the output layer with d_1 neurons. A DNN is a function $\phi(\cdot;\theta) : \mathbb{R}^{d_0} \to \mathbb{R}^{d_1}$ composed of a sequence of simple functions, which therefore can be collected in the following form

$$x \in \mathbb{R}^{d_0} \longmapsto A_{L+1}(\cdot;\theta(L+1)) \circ \varrho \circ A_L(\cdot;\theta(L)) \circ \varrho \circ \ldots \circ \varrho \circ A_1(x;\theta(1)) \in \mathbb{R}^{d_1},$$

where $\theta := (\theta(1), \ldots, \theta(L+1)) \in \mathbb{R}^P$ and P is the total number of network parameters, $x \in \mathbb{R}^{d_0}$ is called an input vector. Moreover, $A_l(\cdot;\theta(l)), l = 1, 2, \ldots, L+1$ are affine transformations: $A_1(\cdot;\theta(1)) : \mathbb{R}^{d_0} \to \mathbb{R}^{\eta}$, $A_l(\cdot;\theta(l)), l = 2, \ldots, L : \mathbb{R}^{\eta} \to \mathbb{R}^{\eta}$ and $A_{L+1}(\cdot;\theta(L+1)) : \mathbb{R}^{\eta} \to \mathbb{R}^{d_1}$, represented by

$$A_l(v;\theta(l)) = \mathcal{W}_l v + \mathcal{B}_l, \quad v \in \mathbb{R}^{\eta_{l-1}},$$

where $\theta(l) := (\mathcal{W}_l, \mathcal{B}_l)$, $\mathcal{W}_l \in \mathbb{R}^{\eta_l \times \eta_{l-1}}$ is the weight matrix and $\mathcal{B}_l \in \mathbb{R}^{\eta_l}$ is the bias vector with $\eta_0 = d_0, \eta_{L+1} = d_1, \eta_l = \eta$ for $l = 1, \ldots, L$ and $\varrho : \mathbb{R} \to \mathbb{R}$ is a nonlinear function (called the activation function), and applied component-wise on the outputs of $A_l(\cdot;\theta(l))$. Common choices are $\tanh(\cdot), \sin(\cdot), \max(0,\cdot)$ etc. All these matrices $\mathcal{W}_l$ and vectors $\mathcal{B}_l$ form the parameters θ of the DNN and can be collected as

$$P = \sum_{l=1}^{L+1} \eta_l(\eta_{l-1}+1) = \eta(d_0+1) + \eta(\eta+1)(L-1) + d_1(\eta+1),$$

for fixed d_0, d_1, L and η. We denote by Θ the set of possible parameters for the DNN $\phi(\cdot;\theta)$ with $\theta \in \Theta$. The Universal Approximation Theorem [23, 9] justifies the use of NNs as function approximators.

3 Uncertainty in the DBSDE scheme

In this section, we discuss the sources of uncertainty in the DBSDE scheme and propose a UQ model to estimate the STD of the approximate solution.

3.1 The DBSDE scheme

We use the time discretization Δ, and for notational convenience we write $W_n = W_{t_n}$, $\Delta W_n = W_{n+1} - W_n$, $(X_n, Y_n, Z_n) = (X_{t_n}, Y_{t_n}, Z_{t_n})$, $(X_n^\Delta, Y_n^\Delta, Z_n^\Delta)$ the approximations of (X_n, Y_n, Z_n). By using the Euler-Maruyama method one obtains

$$\begin{aligned} X_{n+1}^\Delta &= X_n^\Delta + a\left(t_n, X_n^\Delta\right)\Delta t + b\left(t_n, X_n^\Delta\right)\Delta W_n, \\ Y_{n+1}^\Delta &= Y_n^\Delta - f\left(t_n, X_n^\Delta, Y_n^\Delta, Z_n^\Delta\right)\Delta t + Z_n^\Delta \Delta W_n, \end{aligned}$$

where $n = 0, \dots, N-1$. Since the Brownian motions are independent, $\Delta W_n \sim \mathcal{N}(\mathbf{0}_d, \Delta t\, \mathbf{I}_d)$, with $\mathbf{0}_d \in \mathbb{R}^d$ a vector of zeros and $\mathbf{I}_d \in \mathbb{R}^{d\times d}$ the identity matrix. The discretized counterpart of (4) is given as

$$\begin{aligned} &\inf_{Y_0^\Delta \in \mathbb{L}^2_{\mathcal{F}_0}(\Omega;\mathbb{R}),\, Z^\Delta \in \mathbb{H}^{\Delta,2}(\{0,1,\dots,N-1\}\times\Omega;\mathbb{R}^{1\times d})} \mathbf{L}^\Delta\big(Y_0^\Delta, Z^\Delta\big), \\ s.t.\quad & X_0^\Delta = x_0, \\ & X_{n+1}^\Delta = X_n^\Delta + a\left(t_n, X_n^\Delta\right)\Delta t + b\left(t_n, X_n^\Delta\right)\Delta W_n, \\ & Y_{n+1}^\Delta = Y_n^\Delta - f\left(t_n, X_n^\Delta, Y_n^\Delta, Z_n^\Delta\right)\Delta t + Z_n^\Delta \Delta W_n, \end{aligned} \tag{5}$$

for $n = 0, 1, \dots, N-1$, where

$$\mathbf{L}^\Delta\big(Y_0^\Delta, Z^\Delta\big) := \mathbb{E}\big[|g(X_N^\Delta) - Y_N^\Delta|^2\big].$$

The authors in [11] considered DNNs, namely $\phi_0^y : \mathbb{R}^d \to \mathbb{R}$ and $\phi_n^z : \mathbb{R}^d \to \mathbb{R}^{1\times d}$ for $n = 0, 1, \dots, N-1$ to estimate the solution of (5). For a sample of size m we have

$$\begin{aligned} &\min_{\theta\in\Theta} \mathbf{L}^{\Delta,m}\big(\phi_0^y(x_0;\theta_0^y), \phi^z(X^\Delta;\theta^z)\big), \\ s.t.\quad & X_0^\Delta = x_0, \quad Y_0^{\Delta,\theta} = \phi_0^y(x_0;\theta_0^y), \\ & X_{n+1,j}^\Delta = X_{n,j}^\Delta + a\left(t_n, X_{n,j}^\Delta\right)\Delta t + b\left(t_n, X_{n,j}^\Delta\right)\sqrt{\Delta t}\mathcal{Z}_j, \\ & Z_{n,j}^{\Delta,\theta} = \phi_n^z(X_{n,j}^\Delta;\theta_n^z), \\ & Y_{n+1,j}^{\Delta,\theta} = Y_{n,j}^{\Delta,\theta} - f\left(t_n, X_{n,j}^\Delta, Y_{n,j}^{\Delta,\theta}, Z_{n,j}^{\Delta,\theta}\right)\Delta t + Z_{n,j}^{\Delta,\theta}\sqrt{\Delta t}\mathcal{Z}_j, \end{aligned} \tag{6}$$

for $n = 0, 1, \dots, N-1$, $j = 1, \dots, m$, $\mathcal{Z}_j \sim \mathcal{N}(\mathbf{0}_d, \mathbf{I}_d)$ and

$$\mathbf{L}^{\Delta,m}\big(\phi_0^y(x_0;\theta_0^y), \phi^z(X^\Delta;\theta^z)\big) := \frac{1}{m}\sum_{j=1}^m |g(X_{N,j}^\Delta) - Y_{N,j}^{\Delta,\theta}|^2.$$

Note that $Y_0^{\Delta,\theta} = \theta_0^y \in \mathbb{R}$ and $Z_0^{\Delta,\theta} = \theta_0^z \in \mathbb{R}^{1\times d}$ are considered as learnable parameters in [11], which are initialized by sampling from uniform distributions. Furthermore, $N-1$ DNNs are employed to calculate $Z_n^{\Delta,\theta}$ for $n = 1, 2, \dots, N-1$. The architecture of the DBSDE scheme is displayed in Figure 1.

In the numerical section, we also consider the LaDBSDE scheme [31] to demonstrate the applicability of our UQ model to other deep learning-based BSDE schemes. The LaDBSDE scheme addresses the issues encountered in the DBSDE scheme, such as convergence to an approximation far from the true solution or even divergence, especially for a complex solution structure and a long terminal time. In the LaDBSDE scheme, the BSDE problem is formulated as a global optimization problem with local loss functions at each time step. The process Y is approximated using the same DNN ϕ^y, while the process Z is obtained through automatic differentiation due to (3). These approximations are performed by globally minimizing quadratic local loss functions defined at each time step, which always includes the terminal condition Y_T. The loss functions are obtained by iterating the Euler-Maruyama discretization of the integrals with the terminal condition Y_T. For further details, we refer to [31]. In the following sections, we introduce all the sources of uncertainty in the DBSDE scheme and propose a UQ model to estimate the uncertainty. It is important to emphasize that our approach is applicable not only to the DBSDE scheme but also to the LaDBSDE scheme and potentially to other deep learning-based BSDE schemes as well.

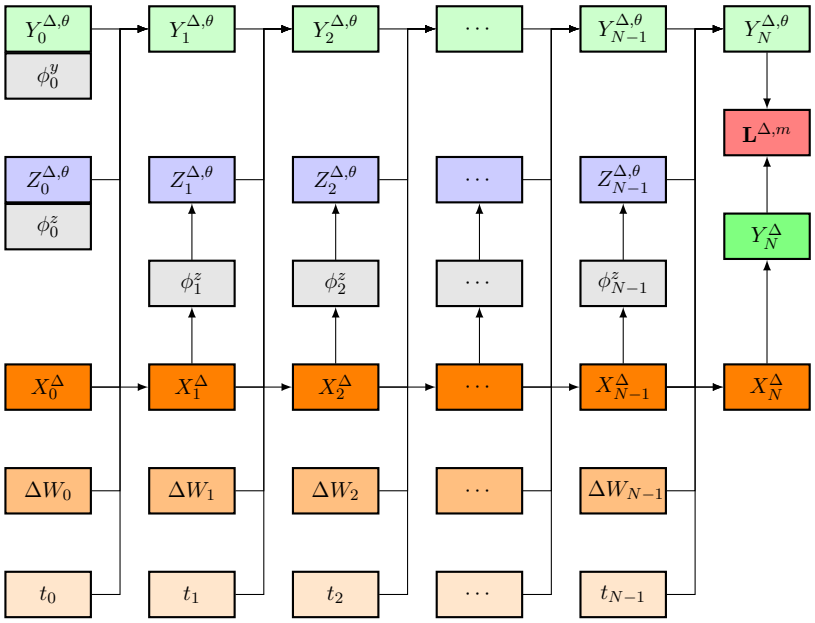


Figure 1: Architecture of the DBSDE scheme.

3.2 Sources of uncertainty in the DBSDE scheme

For notational convenience, we consider

- $\mathbf{Y}^\star := \arg\min_{\mathbf{Y}\in\mathbb{L}^2_{\mathcal{F}_0}(\Omega;\mathbb{R})\times\mathbb{H}^2([0,T]\times\Omega;\mathbb{R}^{1\times d})} \mathbf{L}(\mathbf{Y})$ the solution to (4), where $\mathbf{Y} = (Y_0, Z)$.
- $\mathbf{Y}^{\Delta,\star} := \arg\min_{\mathbf{Y}^\Delta\in\mathbb{L}^2_{\mathcal{F}_0}(\Omega;\mathbb{R})\times\mathbb{H}^{\Delta,2}(\{0,1,\ldots,N-1\}\times\Omega;\mathbb{R}^{1\times d})} \mathbf{L}^\Delta(\mathbf{Y}^\Delta)$ the solution to (5), where $\mathbf{Y}^\Delta = \left(Y_0^\Delta, Z^\Delta\right)$.
- $\theta^\star := \arg\min_{\theta\in\Theta} \mathbf{L}^\Delta(\mathbf{Y}^\theta)$ the optimal parameters, where $\mathbf{Y}^\theta = \left(\phi_0^y(x_0;\theta_0^y), \phi^z(X^\Delta;\theta^z)\right)$.
- $\theta^{m,\star} := \arg\min_{\theta\in\Theta} \mathbf{L}^{\Delta,m}(\mathbf{Y}^\theta)$ the optimal parameters in (6).
- $\mathcal{A} : \mathbb{N} \to \Theta$ the optimization algorithm and $\hat{\theta}^m = \mathcal{A}(m)$ an estimate for the sample of size m.
- $\varphi^N : \mathbb{L}^2_{\mathcal{F}_0}(\Omega;\mathbb{R}) \times \mathbb{H}^{\Delta,2}\left(\{0,\ldots,N-1\}\times\Omega;\mathbb{R}^{1\times d}\right) \to \mathbb{L}^2_{\mathcal{F}_0}(\Omega;\mathbb{R}) \times \mathbb{H}^2\left([0,T]\times\Omega;\mathbb{R}^{1\times d}\right)$ is an interpolation scheme. We choose φ^N such that $\mathbf{L}^\Delta - \mathbf{L}\circ\varphi^N = 0$.

Then, constructing a telescopic sum we obtain the following error decomposition of the DBSDE scheme

$$
\begin{aligned}
\mathbf{L}(\varphi^N(\mathbf{Y}^{\hat{\theta}^m})) \leq\ & \mathbf{L}(\mathbf{Y}^\star) \\
& + \mathbf{L}(\varphi^N(\mathbf{Y}^{\Delta,\star})) - \mathbf{L}(\mathbf{Y}^\star) \\
& + |\mathbf{L}^\Delta(\mathbf{Y}^{\Delta,\star}) - \mathbf{L}(\varphi^N(\mathbf{Y}^{\Delta,\star}))| \\
& + \mathbf{L}^\Delta(\mathbf{Y}^{\theta^\star}) - \mathbf{L}^\Delta(\mathbf{Y}^{\Delta,\star}) \\
& + \mathbf{L}^\Delta(\mathbf{Y}^{\theta^{m,\star}}) - \mathbf{L}^\Delta(\mathbf{Y}^{\theta^\star}) \\
& + |\mathbf{L}^{\Delta,m}(\mathbf{Y}^{\theta^{m,\star}}) - \mathbf{L}^\Delta(\mathbf{Y}^{\theta^{m,\star}})| \\
& + \mathbf{L}^{\Delta,m}(\mathbf{Y}^{\hat{\theta}^m}) - \mathbf{L}^{\Delta,m}(\mathbf{Y}^{\theta^{m,\star}}) \\
& + |\mathbf{L}^\Delta(\mathbf{Y}^{\hat{\theta}^m}) - \mathbf{L}^{\Delta,m}(\mathbf{Y}^{\hat{\theta}^m})| \\
& + |\mathbf{L}(\varphi^N(\mathbf{Y}^{\hat{\theta}^m})) - \mathbf{L}^\Delta(\mathbf{Y}^{\hat{\theta}^m})|.
\end{aligned}
$$

Since $\mathbf{L}^{\Delta} - \mathbf{L} \circ \varphi^N = 0$, we have

$$\begin{aligned}
\mathbf{L}(\varphi^N(\mathbf{Y}^{\hat{\theta}^m})) \leq\ & \mathbf{L}(\varphi^N(\mathbf{Y}^{\Delta,\star})) && (7)\\
& + \mathbf{L}^{\Delta}(\mathbf{Y}^{\theta^\star}) - \mathbf{L}^{\Delta}(\mathbf{Y}^{\Delta,\star}) && (8)\\
& + \mathbf{L}^{\Delta}(\mathbf{Y}^{\theta^{m,\star}}) - \mathbf{L}^{\Delta}(\mathbf{Y}^{\theta^\star}) && (9)\\
& + |\mathbf{L}^{\Delta,m}(\mathbf{Y}^{\theta^{m,\star}}) - \mathbf{L}^{\Delta}(\mathbf{Y}^{\theta^{m,\star}})| && (10)\\
& + \mathbf{L}^{\Delta,m}(\mathbf{Y}^{\hat{\theta}^m}) - \mathbf{L}^{\Delta,m}(\mathbf{Y}^{\theta^{m,\star}}) && (11)\\
& + |\mathbf{L}^{\Delta}(\mathbf{Y}^{\hat{\theta}^m}) - \mathbf{L}^{\Delta,m}(\mathbf{Y}^{\hat{\theta}^m})|. && (12)
\end{aligned}$$

In the decomposition above, each term corresponds to a specific source of uncertainty. The naming convention we adopt here is motivated by [24]. The term (7) captures the discretization error associated with the Euler-Maruyama method. Note that a DNN architecture has to be chosen when implementing the algorithm. Hence, (8) represents the model or approximation error [4, 26]. Since the scheme optimizes the empirical loss, (9) denotes the estimation error, as the empirical loss is only an estimate of the true loss. Selecting an SGD-type algorithm to optimize the DNN parameters introduces the optimization error, represented by (11). Finally, (10) and (12) correspond to the sampling errors, which are insignificant compared to the other error sources. Identifying these errors is important to evaluate their practical impact on the uncertainty of the scheme. In the numerical section, we discuss such errors.

3.3 The UQ model

In practice, it is challenging to disentangle the sources of uncertainty in the DBSDE scheme, as we demonstrate in the numerical experiments. Consequently, quantifying the uncertainty of the DBSDE scheme becomes crucial for practical applications. In this section, we develop a UQ model to estimate the uncertainty. After applying the DBSDE scheme, we obtain the random variables $Y_0^{\Delta,\hat{\theta}^m} \in \mathbb{R}$ and $Z_0^{\Delta,\hat{\theta}^m} \in \mathbb{R}^{1\times d}$ that approximate Y_0 and Z_0, respectively. To evaluate the quality of such approximations, one can use the expected squared error when the exact solution is known. This metric accounts for all the error sources in the DBSDE scheme and is calculated as

$$\epsilon^y := \sqrt{\mathbb{E}\big[(Y_0^{\Delta,\hat{\theta}^m} - Y_0)^2\big]} \in \mathbb{R}^+, \quad \epsilon^{z_k} := \sqrt{\mathbb{E}\big[(Z_0^{\Delta,\hat{\theta}^m,k} - Z_0^k)^2\big]} \in \mathbb{R}^+,$$

for $k = 1, \dots, d$. However, (Y_0, Z_0) is usually unknown, the STD of the approximate solutions

$$\sigma^y := \sqrt{\mathbb{E}\big[(Y_0^{\Delta,\hat{\theta}^m} - \mu^y)^2\big]} \in \mathbb{R}^+, \quad \sigma^{z_k} := \sqrt{\mathbb{E}\big[(Z_0^{\Delta,\hat{\theta}^m,k} - \mu^{z_k})^2\big]} \in \mathbb{R}^+,$$

is often used, where $k = 1, \dots, d$, $\mu^y := \mathbb{E}\big[Y_0^{\Delta,\hat{\theta}^m}\big] \in \mathbb{R}$ and $\mu^{z_k} := \mathbb{E}\big[Z_0^{\Delta,\hat{\theta}^m,k}\big] \in \mathbb{R}$. To compute the STD (and the expected squared error when the exact solution (Y_0, Z_0) is available), Q runs of the DBSDE algorithm must be done. The STD and the expected squared error are used as the benchmark in our experiments. To this end, we have the root mean squared error (RMSE) and the ensemble (biased sample) STD as

$$\tilde{\epsilon}^y := \sqrt{\frac{1}{Q}\sum_{q=1}^{Q}\Big(Y_{0,q}^{\Delta,\hat{\theta}^m} - Y_0\Big)^2}, \quad \tilde{\sigma}^y := \sqrt{\frac{1}{Q}\sum_{q=1}^{Q}\Big(Y_{0,q}^{\Delta,\hat{\theta}^m} - \tilde{\mu}^y\Big)^2},$$

for Y_0 and

$$\tilde{\epsilon}^{z_k} := \sqrt{\frac{1}{Q}\sum_{q=1}^{Q}\Big(Z_{0,q}^{\Delta,\hat{\theta}^m,k} - Z_0^k\Big)^2}, \quad \tilde{\sigma}^{z_k} := \sqrt{\frac{1}{Q}\sum_{q=1}^{Q}\Big(Z_{0,q}^{\Delta,\hat{\theta}^m,k} - \tilde{\mu}^{z_k}\Big)^2},$$

for Z_0, $k = 1, \ldots, d$, where $\left(Y_{0,q}^{\Delta,\hat{\theta}^m}, Z_{0,q}^{\Delta,\hat{\theta}^m}\right)$ represents the approximated solutions from the q-th run of the algorithm, $\tilde{\mu}^y := \frac{1}{Q}\sum_{q=1}^{Q} Y_{0,q}^{\Delta,\hat{\theta}^m}$ and $\tilde{\mu}^{z_k} := \frac{1}{Q}\sum_{q=1}^{Q} Z_{0,q}^{\Delta,\hat{\theta}^m,k}$ are the ensemble (sample) means of the approximate solution. Note that one can use the ensemble unbiased STD. However, the estimate of the STD from the UQ model is biased (as a maximum likelihood estimate). Therefore, for the purpose of comparison in numerical experiments, it is more convenient to use the ensemble biased STD. Usually, $Q = 10$ is used, which is computationally expensive in high dimensions. Therefore, we propose a UQ model to estimate the STDs for $\left(Y_0^{\Delta,\hat{\theta}^m}, Z_0^{\Delta,\hat{\theta}^m}\right)$ by using only $Q = 1$ run of the algorithm. The model is based on an approach commonly used to quantify uncertainty in heteroscedastic nonlinear regression. We make the assumption that the errors of the DBSDE scheme follow a normal distribution with zero mean and the STD depending on the parameter set of the discretized BSDE (such as T, x_0, Δt, etc.). This assumption aligns with the standard practice in heteroscedastic regression. To train the UQ model, we construct a dataset of length M with i.i.d samples $\mathcal{D} = \{\mathbf{x}_i, \mathbf{y}_i, \mathbf{z}_i\}_{i=1}^{M}$. Here, $\mathbf{x}_i \in \mathbb{R}^{\mathfrak{n}}$ represents $\mathfrak{n}$ parameters of the discretized BSDE, for example, $\mathbf{x}_i := (x_{0,i}, T_i, \Delta t_i)$, and $(\mathbf{y}_i, \mathbf{z}_i) := \left(Y_0^{\Delta,\hat{\theta}^m}(\mathbf{x}_i), Z_0^{\Delta,\hat{\theta}^m}(\mathbf{x}_i)\right) \in \mathbb{R} \times \mathbb{R}^{1\times d}$ are the approximations of $(Y_0(\mathbf{x}_i), Z_0(\mathbf{x}_i))$ obtained from the DBSDE scheme using parameter set $\mathbf{x}_i$. We use uniform distributions to select the parameter set $\mathbf{x}_i$

$$x_{0,i} \sim \mathcal{U}[x_0^{min},\, x_0^{max}], \quad T_i \sim \mathcal{U}[T^{min},\, T^{max}],$$

where (x_0^{min}, x_0^{max}) and (T^{min}, T^{max}) are the boundaries of the corresponding uniform distributions for x_0 and T. The dataset $\mathcal{D}$ is generated using Algorithm 1. Note that one can use the entire dataset to build a learning algorithm that considers Y_0 and Z_0 as pairs (the BSDE solution at t_0 is the pair (Y_0, Z_0)). However, this approach may introduce increased complexity for the learning algorithm, mainly because the magnitudes of the solutions for $\mathbf{y}$ and $\mathbf{z}$ over different parameter sets $\mathbf{x}$ can differ significantly. Additionally, assumptions regarding their correlation might be necessary. Hence, we divide the dataset $\mathcal{D}$ into two datasets, $\mathcal{D}^y = \{\mathbf{x}_i, \mathbf{y}_i\}_{i=1}^{M}$ and $\mathcal{D}^z = \{\mathbf{x}_i, \mathbf{z}_i\}_{i=1}^{M}$, and develop two different learning algorithms. Given the input feature $\mathbf{x}$, we use one DNN to model the probabilistic predictive distribution $p^{\omega}(\mathbf{y}|\mathbf{x})$ and another for $p^{\psi}(\mathbf{z}|\mathbf{x})$, where ω and ψ are the parameters of the corresponding DNNs. More precisely, we treat each observed value as a sample from a Gaussian distribution (multivariate Gaussian for Z_0), with the mean and STD as functions of the parameter set, namely

$$\mathbf{y}_i \sim \mathcal{N}(\mu^y(\mathbf{x}_i),\, (\sigma^y)^2(\mathbf{x}_i)), \quad \mathbf{z}_i \sim \mathcal{N}(\mu^z(\mathbf{x}_i),\, \mathbf{I}_d\,(\sigma^z)^2(\mathbf{x}_i)), \tag{13}$$

and allow the networks to calculate their estimates as $(\hat{\mu}^{y,\omega}(\mathbf{x}_i), \hat{\sigma}^{y,\omega}(\mathbf{x}_i)) \in \mathbb{R} \times \mathbb{R}^+$ and $\left(\hat{\mu}^{z,\psi}(\mathbf{x}_i), \hat{\sigma}^{z,\psi}(\mathbf{x}_i)\right) \in \mathbb{R}^d \times \mathbb{R}^{d,+}$. These calculations are performed by minimizing the negative log-likelihood

$$\mathbf{L}^{\omega,M}(\mathcal{D}^y) := -\log\left(p^{\omega}(\mathbf{y}|\mathbf{x})\right) = \frac{1}{M}\sum_{i=1}^{M}\left(\log\left(\hat{\sigma}^{y,\omega}(\mathbf{x}_i)\right) + \frac{1}{2}\frac{(\mathbf{y}_i - \hat{\mu}^{y,\omega}(\mathbf{x}_i))^2}{(\hat{\sigma}^{y,\omega})^2(\mathbf{x}_i)}\right) + c^y,$$

for Y_0 and assuming that the covariance matrix of $Z_0^{\Delta,\hat{\theta}^m}$ is diagonal, then

$$\begin{aligned}\mathbf{L}^{\psi,M}(\mathcal{D}^z) := -\log\left(p^{\psi}(\mathbf{z}|\mathbf{x})\right) = \frac{1}{M}\sum_{i=1}^{M}\Bigg(&\sum_{k=1}^{d}\log\left(\hat{\sigma}^{z_k,\psi}(\mathbf{x}_i)\right) \\ &+\frac{1}{2}\left(\mathbf{z}_i - \hat{\mu}^{z,\psi}(\mathbf{x}_i)\right)^{\top}\left((\hat{\sigma}^{z,\psi})^2(\mathbf{x}_i)\right)^{-1}\left(\mathbf{z}_i - \hat{\mu}^{z,\psi}(\mathbf{x}_i)\right)\Bigg) + c^z,\end{aligned}$$

for Z_0, where $c^y > 0, c^z > 0$ are constants. We use Algorithm 2 and 3 to estimate the parameters in (13) for Y_0 and Z_0, respectively.

Algorithm 1: Algorithm generating dataset $\mathcal{D}$ for UQ model

Input: $\left(N, M, d, x_0^{min}, x_0^{max}, T^{min}, T^{max}, Y_0^{min}, Y_0^{max}\right)$ - problem related parameters
Input: (a, b, f, g) - functions of BSDE system
Input: $(\alpha, \mathfrak{K}, L, \eta, \varrho, m)$ - DNN hyperparameters in DBSDE scheme
Output: $\mathcal{D} = \{\mathbf{x}_i, \mathbf{y}_i, \mathbf{z}_i\}_{i=1}^{M}$ - Dataset for UQ model
for $i = 1 : M$ **do**
$x_{0,i} \sim \mathcal{U}[x_0^{min},\, x_0^{max}]$
$T_i \sim \mathcal{U}[T^{min},\, T^{max}]$
$\Delta t_i = \frac{T_i}{N}$
$\mathbf{x}_i = (x_{0,i}, T_i, \Delta t_i)$
for $n = 0 : N$ **do**
$t_n = n\,\Delta t_i$
end
Initialize parameter set θ
$\hat{\theta}_0^{y,m,0} = Y_0^{\Delta,\hat{\theta}^{m,0}}(\mathbf{x}_i) \sim \mathcal{U}[Y_0^{min},\, Y_0^{max}]$
$\hat{\theta}_0^{z,m,0} = Z_0^{\Delta,\hat{\theta}^{m,0}}(\mathbf{x}_i) \sim \mathcal{U}[-\mathbf{1}_d,\, \mathbf{1}_d]$ - $\mathbf{1}_d$ vector of all ones
$\left(\hat{\theta}_1^{z,m,0}, \ldots, \hat{\theta}_{N-1}^{z,m,0}\right)$- Xavier normal initializer [16]
$\hat{\theta}^{m,0} = \left(\hat{\theta}_0^{y,m,0}, \hat{\theta}_0^{z,m,0}, \hat{\theta}_1^{z,m,0}, \ldots, \hat{\theta}_{N-1}^{z,m,0}\right)$
Optimization or training part
for $\kappa = 1 : \mathfrak{K}$ **do**
for $j = 1 : m$ **do**
$X_{0,j}^{\Delta} = x_{0,i}$
for $n = 0 : N-1$ **do**
Euler-Maruyama for the forward SDE
$\Delta W_{n,j} \sim \mathcal{N}(\mathbf{0}_d, \Delta t_i\, \mathbf{I}_d)$
$X_{n+1,j}^{\Delta} = X_{n,j}^{\Delta} + a\left(t_n, X_{n,j}^{\Delta}\right)\Delta t_i + b\left(t_n, X_{n,j}^{\Delta}\right)\Delta W_{n,j}$
Use DNN with (L, η, ϱ) **for** Z **and Euler-Maruyama for** Y
$d_0 = d_1 = d$
if $n < N-1$ **then**
$Z_{n+1,j}^{\Delta,\hat{\theta}^{m,\kappa-1}} = \phi_{n+1}^{z}(X_{n+1,j}^{\Delta}; \hat{\theta}_{n+1}^{z,m,\kappa-1})$
$Y_{n+1,j}^{\Delta,\hat{\theta}^{m,\kappa-1}} = Y_{n,j}^{\Delta,\hat{\theta}^{m,\kappa-1}} - f(t_n, X_{n,j}^{\Delta}, Y_{n,j}^{\Delta,\hat{\theta}^{m,\kappa-1}}, Z_{n,j}^{\Delta,\hat{\theta}^{m,\kappa-1}})\Delta t_i + Z_{n,j}^{\Delta,\hat{\theta}^{m,\kappa-1}}\Delta W_{n,j}$
else
$Y_{n+1,j}^{\Delta,\hat{\theta}^{m,\kappa-1}} = Y_{n,j}^{\Delta,\hat{\theta}^{m,\kappa-1}} - f(t_n, X_{n,j}^{\Delta}, Y_{n,j}^{\Delta,\hat{\theta}^{m,\kappa-1}}, Z_{n,j}^{\Delta,\hat{\theta}^{m,\kappa-1}})\Delta t_i + Z_{n,j}^{\Delta,\hat{\theta}^{m,\kappa-1}}\Delta W_{n,j}$
end
end
end
$\mathbf{L}^{\Delta,m}(\hat{\theta}^{m,\kappa-1}) = \frac{1}{m}\sum_{j=1}^{m} |g(X_{N,j}^{\Delta}) - Y_{N,j}^{\Delta,\hat{\theta}^{m,\kappa-1}}|^2$
Adam optimization step
$\hat{\theta}^{m,\kappa}$ - trained parameters with Adam optimizer [34], learning rate α
end
$\hat{\theta}^{m} = \hat{\theta}^{m,\mathfrak{K}}$ - final estimated parameters after $\mathfrak{K}$ optimization steps
$(\mathbf{y}_i, \mathbf{z}_i) = \left(Y_0^{\Delta,\hat{\theta}^{m}}(\mathbf{x}_i), Z_0^{\Delta,\hat{\theta}^{m}}(\mathbf{x}_i)\right)$
end

Algorithm 2: Algorithm estimating the parameters of (13) for Y_0

Input: $\left(M, \mathfrak{n}, M^{valid}, M^{test}, \mathcal{D}^y\right)$ - parameters and dataset for UQ model

Input: $(L^y, \eta^y, \varrho^y, \alpha^y, B^y, ep^y, \lambda^y)$ - DNN hyperparameters

Output: $\left(\hat{\mu}^{y,\hat{\omega}^M}(\mathbf{x}), \hat{\sigma}^{y,\hat{\omega}^M}(\mathbf{x})\right)$ - estimates of UQ model for Y_0

Split $\mathcal{D}^y$ into training, validation and testing samples

$M^{train} = M - M^{valid} - M^{test}$

$(\mathbf{x}^{train}, \mathbf{y}^{train})$

$(\mathbf{x}^{valid}, \mathbf{y}^{valid})$

$(\mathbf{x}^{test}, \mathbf{y}^{test})$

Normalize input data x based on training statistics

$(\mathbf{x}^{train,nr}, \mathbf{x}^{valid,nr}, \mathbf{x}^{test,nr})$

$d_0 = \mathfrak{n}$

$d_1^y = 2$

Initialize parameters ω

$\hat{\omega}^{M,0}$- Xavier normal initializer [16]

$\kappa = 0$

for $e = 1 : ep^y$ **do**

  $\kappa = \kappa + 1$

  for $I = 1 : \frac{M^{train}}{B^y}$ **do**

    Batch data

    $\mathcal{D}^{y,train,nr} = \{\mathbf{x}_i^{train,nr}, \mathbf{y}_i^{train}\}_{i=(I-1)\,B^y+1}^{I\,B^y}$

    Use DNN with (L^y, η^y, ϱ^y) to estimate parameters in (13) for Y_0

    $\left(\hat{\mu}^{y,\hat{\omega}^{M,\kappa-1}}(\mathbf{x}), \hat{\sigma}^{y,\hat{\omega}^{M,\kappa-1}}(\mathbf{x})\right) = \phi^y\left(\{\mathbf{x}_i^{train,nr}\}_{i=(I-1)\,B^y+1}^{I\,B^y}; \hat{\omega}^{M,\kappa-1}\right)$

    Calculate loss including L2 regularization

$$\mathbf{L}^{\hat{\omega}^{M,\kappa-1},M}(\mathcal{D}^{y,train,nr}) = \frac{1}{B^y}\sum_{i=(I-1)\,B^y+1}^{I\,B^y}\left(\log\left(\hat{\sigma}^{y,\hat{\omega}^{M,\kappa-1}}(\mathbf{x}_i^{train,nr})\right) + \frac{1}{2}\frac{\left(\mathbf{y}_i^{train,nr} - \hat{\mu}^{y,\hat{\psi}^{M,\kappa-1}}(\mathbf{x}_i^{train,nr})\right)^2}{(\hat{\sigma}^{y,\hat{\omega}^{M,\kappa-1}})^2(\mathbf{x}_i^{train,nr})}\right) + \lambda^y\sum_{l=1}^{L^y+1}\left(\hat{\omega}^{M,\kappa-1}(l)\right)^2$$

    Adam optimization step

    $\hat{\omega}^{M,\kappa}$ - trained parameters with Adam optimizer [34], learning rate α^y

  end

end

$\hat{\omega}^M = \hat{\omega}^{M,\kappa}$ - final estimated parameters of DNN after ep^y epochs, each with $\frac{M^{train}}{B^y}$ number of batches

$\left(\hat{\mu}^{y,\hat{\omega}^M}(\mathbf{x}), \hat{\sigma}^{y,\hat{\omega}^M}(\mathbf{x})\right)$ - estimated parameters of (13) for Y_0, $\mathbf{x}$ training, validation or testing sample

Algorithm 3: Algorithm estimating the parameters of (13) for Z_0

Input: $\left(M, \mathfrak{n}, M^{valid}, M^{test}, \mathcal{D}^z\right)$ - parameters and dataset for UQ model
Input: $(L^z, \eta^z, \varrho^z, \alpha^z, B^z, ep^z, \lambda^z)$ - DNN hyperparameters
Output: $\left(\hat{\mu}^{z,\hat{\psi}^M}(\mathbf{x}), \hat{\sigma}^{z,\hat{\psi}^M}(\mathbf{x})\right)$ - estimates of UQ model for Z_0
Split $\mathcal{D}^z$ into training, validation and testing samples
$M^{train} = M - M^{valid} - M^{test}$
$(\mathbf{x}^{train}, \mathbf{z}^{train})$
$(\mathbf{x}^{valid}, \mathbf{z}^{valid})$
$(\mathbf{x}^{test}, \mathbf{z}^{test})$
Normalize input data x based on training statistics
$(\mathbf{x}^{train,nr}, \mathbf{x}^{valid,nr}, \mathbf{x}^{test,nr})$
$d_0 = \mathfrak{n}$
$d_1^z = 2d$
Initialize parameters ψ
$\hat{\psi}^{M,0}$- Xavier normal initializer [16]
$\kappa = 0$
for $e = 1 : ep^z$ **do**
  $\kappa = \kappa + 1$
  for $I = 1 : \frac{M^{train}}{B^z}$ **do**
    Batch data
    $\mathcal{D}^{z,train,nr} = \{\mathbf{x}_i^{train,nr}, \mathbf{z}_i^{train}\}_{i=(I-1)\,B^z+1}^{I\,B^z}$
    Use DNN with (L^z, η^z, ϱ^z) to estimate parameters in (13) for Z_0
    $\left(\hat{\mu}^{z,\hat{\psi}^{M,\kappa-1}}(\mathbf{x}), \hat{\sigma}^{z,\hat{\psi}^{M,\kappa-1}}(\mathbf{x})\right) = \phi^z\left(\{\mathbf{x}_i^{train,nr}\}_{i=(I-1)\,B^z+1}^{I\,B^z}; \hat{\psi}^{M,\kappa-1}\right)$
    Calculate loss including L2 regularization

$$\mathbf{L}^{\hat{\psi}^{M,\kappa-1},M}(\mathcal{D}^{z,train,nr}) = \frac{1}{B^z}\sum_{i=(I-1)\,B^z+1}^{I\,B^z}\Bigg(\sum_{k=1}^{d}\log\left(\hat{\sigma}^{z_k,\hat{\psi}^{M,\kappa-1}}(\mathbf{x}_i^{train,nr})\right) + \frac{1}{2}\left(\mathbf{z}_i^{train,nr} - \hat{\mu}^{z,\hat{\psi}^{M,\kappa-1}}(\mathbf{x}_i^{train,nr})\right)^{\top} \left((\hat{\sigma}^{z,\hat{\psi}^{M,\kappa-1}})^2(\mathbf{x}_i^{train,nr})\right)^{-1}\left(\mathbf{z}_i^{train,nr} - \hat{\mu}^{z,\hat{\psi}^{M,\kappa-1}}(\mathbf{x}_i^{train,nr})\right)\Bigg) + \lambda^z\sum_{l=1}^{L^z+1}\left(\hat{\psi}^{M,\kappa-1}(l)\right)^2$$

    Adam optimization step
    $\hat{\psi}^{M,\kappa}$ - trained parameters with Adam optimizer [34], learning rate α^z
  end
end
$\hat{\psi}^M = \hat{\psi}^{M,\kappa}$ - final estimated parameters of DNN after ep^z epochs, each with $\frac{M^{train}}{B^z}$ number of batches
$\left(\hat{\mu}^{z,\hat{\psi}^M}(\mathbf{x}), \hat{\sigma}^{z,\hat{\psi}^M}(\mathbf{x})\right)$ - estimated parameters of (13) for Z_0, $\mathbf{x}$ training, validation or testing sample

4 Numerical results

In this section, we take the DBSDE scheme as an example to illustrate the impact of different sources of uncertainty in the scheme and apply our UQ model to both the DBSDE and LaDBSDE schemes. All the experiments were run in PYTHON using TensorFlow on the PLEIADES cluster, which consists of 268 workernodes. Each workernode has 2 sockets with an AMD EPYC 7452 32-Core processor (256GB of memory). For more information, see PLEIADES documentation[1].

4.1 Experimental setup

To efficiently generate the dataset for the UQ model, we consider a straightforward parallelization of Algorithm 1, namely one core is used to simulate the DBSDE scheme for one parameter set. Hence, multiple cores provide a parallel generation of the dataset $\mathcal{D}$. The implementation of Algorithm 1 follows the hyperparameters considered in [11] for the DBSDE scheme. Each of the DNNs consists of $L = 2$ hidden layers with $\eta = 10 + d$ neurons per hidden layer, and the rectifier function (ReLU) $\varrho(x) = \max(0, x) \in [0, \infty)$ is used as the activation function. Batch normalization is applied right after each matrix multiplication and before activation. Furthermore, all network parameters are initialized using a normal distribution without any pre-training. The Adam optimizer with learning rate α and $\mathfrak{K}$ optimization steps is used as an SGD-type algorithm. For the implementation of Algorithm 2, we first split the dataset $\mathcal{D}^y$ into training, validation, and testing samples, whose sizes are denoted by M^{train}, M^{valid} and M^{test}, respectively. Note that the validation set is used to validate the performance of our UQ model when tuning its hyperparameters. The input layer of the DNN has $\mathfrak{n}$ neurons, and the output layer has 2 neurons. The first neuron in the output layer estimates the mean of the approximate solution $\mu^y(\mathbf{x})$, and the second neuron in the output layer estimates the STD of the approximate solution $\sigma^y(\mathbf{x})$, where the softplus activation function $\varrho(x) = \ln(1 + e^x) \in (0, \infty)$ is applied to obtain positive estimates. The ReLU activation function is used for the L^y hidden layers. Note that it is appropriate to choose $\eta^y > d_1^y$. The input data $\mathbf{x}$ is normalized based on the training data. We use the Adam optimizer with a batch size B^y, L2 regularization with parameter λ^y, and a specified number of epochs ep^y. The hyperparameters are set in a similar fashion for the implementation of Algorithm 3.

To visually and quantitatively compare our estimates of the mean and STD of the approximate solution to benchmark values, such as the exact solution, RMSE, ensemble mean, and ensemble STD, we consider both linear and nonlinear BSDEs with available analytical solutions. We take the Black-Scholes BSDE as a linear 1-dimensional example, which is used for pricing the European options.

Example 1. *The Black-Scholes BSDE reads [55]*

$$\begin{cases} dS_t &= aS_t\,dt + bS_t\,dW_t, \quad S_0 = s_0, \\ -dY_t &= -\left(RY_t + (a - R + \delta)\,\frac{Z_t}{b}\right)\,dt - Z_t\,dW_t, \\ Y_T &= (S_T - K)^+ \end{cases}$$

Note that a represents the expected return of the stock S_t, b denotes the volatility of the stock returns, δ is the dividend rate the stock pays, and S_0 the price of the stock at $t = 0$. Moreover, T denotes the maturity of the option contract, while K represents the contract's strike price. Finally, R corresponds to the risk-free interest rate. The analytic solution (the option price Y_t and its delta hedging strategy Z_t) is given by

$$\begin{cases} Y_t &= S_t \exp\left(-\delta\,(T-t)\right)\Phi\left(d_1\right) - K\exp\left(-R\,(T-t)\right)\Phi\left(d_2\right), \\ Z_t &= S_t \exp\left(-\delta\,(T-t)\right)\Phi\left(d_1\right) b, \\ d_{1/2} &= \frac{\ln\left(\frac{S_t}{K}\right) + \left(R - \delta \pm \frac{b^2}{2}\right)(T-t)}{b\sqrt{T-t}}, \end{cases}$$

[1] https://pleiadesbuw.github.io/PleiadesUserDocumentation/

where $\Phi(\cdot)$ is the standard normal cumulative distribution function. For the nonlinear case, we consider the nonlinear high-dimensional Burgers type BSDE.

Example 2. *The Burgers type BSDE reads [11]*

$$\begin{cases} dX_t &= b\, dW_t, \quad X_0 = 0, \\ -dY_t &= \left(\frac{b}{d}Y_t - \frac{2d+b^2}{2bd}\right)\left(\sum_{k=1}^{d} Z_t^k\right) dt - Z_t\, dW_t, \\ Y_T &= \frac{\exp\left(T+\frac{1}{d}\sum_{k=1}^{d} X_T^k\right)}{1+\exp\left(T+\frac{1}{d}\sum_{k=1}^{d} X_T^k\right)}, \end{cases}$$

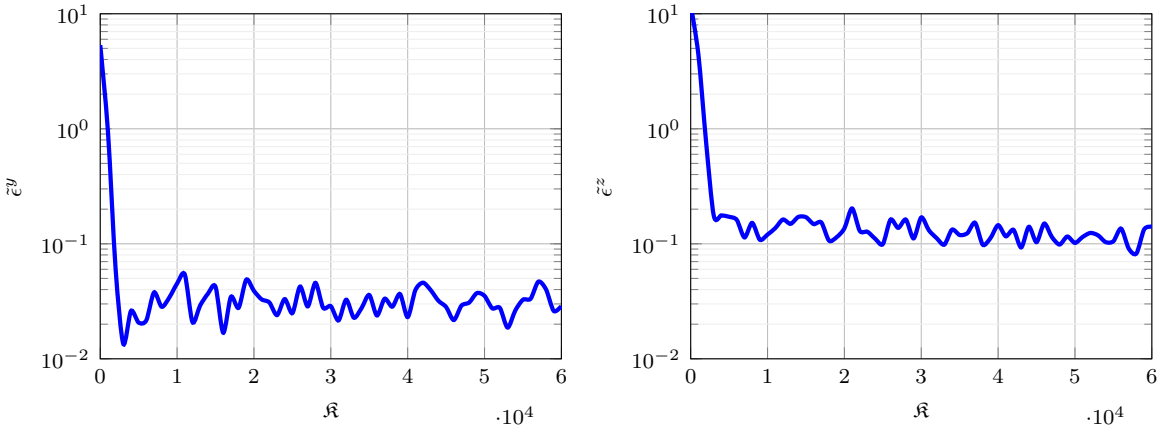
where, $W_t = (W_t^1, W_t^2, \ldots, W_t^d)^\top$, $X_t = (X_t^1, X_t^2, \ldots, X_t^d)^\top$ and $Z_t = (Z_t^1, Z_t^2, \ldots, Z_t^d)$. The analytic solution is given by

$$\begin{cases} Y_t &= \frac{\exp\left(t+\frac{1}{d}\sum_{k=1}^{d} X_t^k\right)}{1+\exp\left(t+\frac{1}{d}\sum_{k=1}^{d} X_t^k\right)}, \\ Z_t &= \frac{b}{d}\frac{\exp\left(t+\frac{1}{d}\sum_{k=1}^{d} X_t^k\right)}{\left(1+\exp\left(t+\frac{1}{d}\sum_{k=1}^{d} X_t^k\right)\right)^2}\mathbf{1}_d. \end{cases}$$

At t_0 we have that $(Y_0, Z_0) = \left(0.5, \frac{b}{4d}\mathbf{1}_d\right)$.
The following experiments are organized as follows. Firstly, we illustrate the impact of the sources of uncertainty in the DBSDE scheme for both examples by visualizing the effect of different errors on the uncertainty. Secondly, we assess the performance of our UQ model for the mean and STD of the approximate solution by comparing them with the benchmark values. Additionally, we determine the number of runs of the DBSDE algorithm for which the ensemble STD is comparable to the estimated STD. Furthermore, the computational cost of generating the training data for the UQ model is evaluated. To demonstrate the applicability of the UQ model to other deep learning-based BSDE schemes, we apply it to the LaDBSDE scheme. Finally, we show the practical implications of our UQ model.

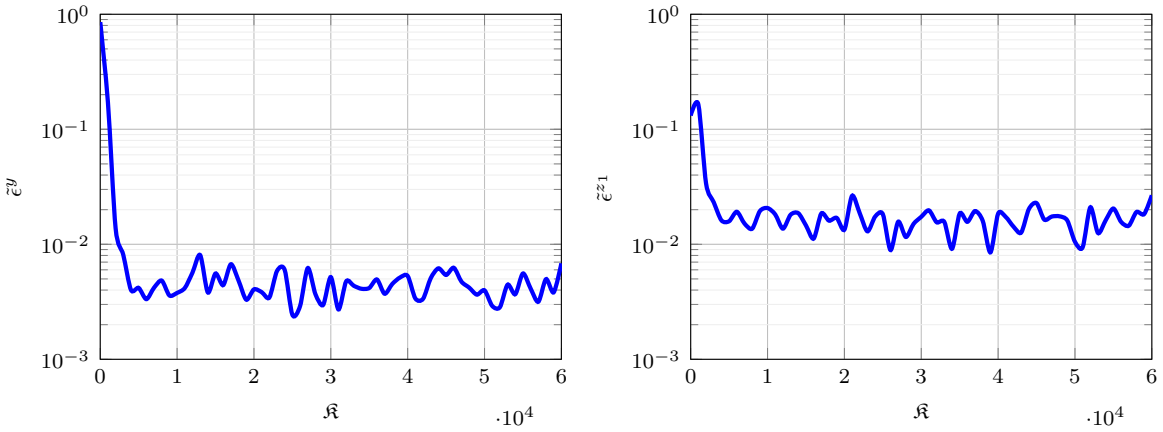
4.2 The impact of the sources of uncertainty in the DBSDE scheme

To demonstrate the impact of the sources of uncertainty in the DBSDE scheme, we fix the parameter set of the BSDE and vary the hyperparameters of the DBSDE scheme that affect the corresponding source of uncertainty. Initially, we focus on the estimation and optimization errors. By using a high number of optimization steps $\mathfrak{K}$ and different learning rate approaches, the effect of these errors on the uncertainty of the DBSDE scheme is analyzed. Afterward, we investigate the impact of the model error by increasing the number of hidden neurons η. Lastly, the number of discretization points N is varied in order to study how the discretization error contributes to the uncertainty of the DBSDE scheme.
For the parameter values $T = 1, K = 100, S_0 = 100, a = 0.05, b = 0.2, R = 0.03$ and $\delta = 0$ in Example 1, the exact solution is $(Y_0, Z_0) = (9.4134, 11.9741)$. In Example 2, we chose $d = 50$ and fix $b = 25, T = 0.25$. The exact solution is $(Y_0, Z_0) = (0.5, 0.125\mathbf{1}_{50})$. For the DBSDE algorithm, we start with the following hyperparameters: a constant learning rate approach (C-LR) with $\mathfrak{K} = 60000$ optimization steps of the Adam algorithm, a learning rate $\alpha = 1\text{e}{-2}$, and a batch size of $m = 128$. To analyze the effect of estimation and optimization errors on the RMSE, we plot the RMSE values in Figures 2 and 3 for Examples 1 and 2, respectively, for increasing values of $\mathfrak{K}$. The RMSE values of Z_0 are plotted only for the first component in Example 2 as it is similar for the other components in our experiments. We use $N = 32$ discretization points and $Q = 10$ runs of the DBSDE algorithm. Note that each run of the DBSDE algorithm involves a different seed for generating the dataset and different initialization values of DNN parameters. As $\mathfrak{K}$ increases, the sum of estimation and optimization errors decreases for both examples as expected. This is because the DBSDE algorithm uses a new sample of size 128 after each optimization step, and the optimizer tends to perform better with more training data and



(a) RMSE values for Y_0. (b) RMSE values for Z_0.

Figure 2: RMSE values are plotted for Example 1 for increasing $\mathfrak{K}$, where $T = 1, K = 100, S_0 = 100, a = 0.05, b = 0.2, R = 0.03$ and $\delta = 0$.



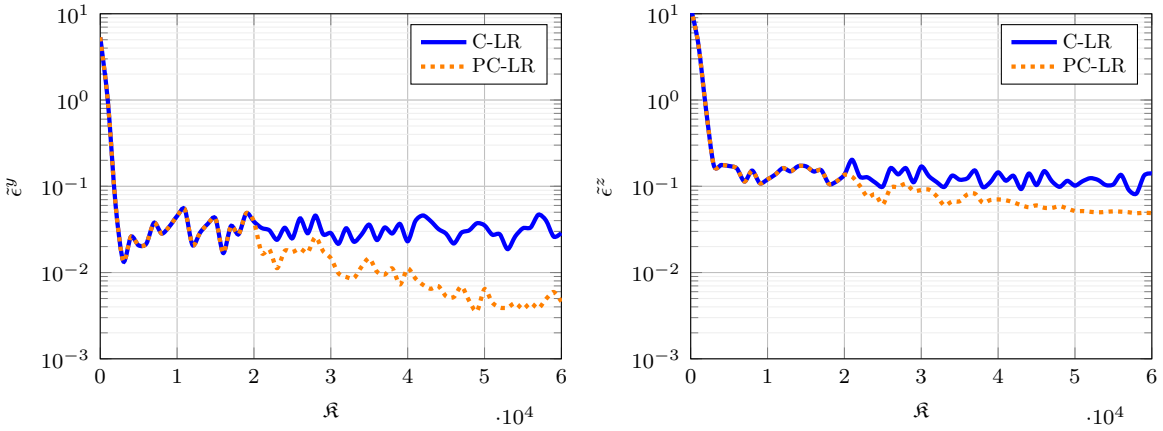
(a) RMSE values for Y_0. (b) RMSE values for Z_0^1.

Figure 3: RMSE values are plotted for Example 2 while increasing $\mathfrak{K}$, where $T = 0.25$ and $b = 25$.

optimization steps. This reduction in RMSE is evident until around 5000 optimization steps. However, for $\mathfrak{K} > 5000$, the RMSE plateaus due to other error sources, such as model and discretization errors, which are higher than the optimization error. Note that the RMSE values for Example 2 are lower than those for Example 1 because the exact solution has a smaller value. To further reduce the optimization error, we use a piecewise constant learning rate approach (PC-LR) with $\alpha = \{1\text{e}{-2}, 3\text{e}{-3}, 1\text{e}{-3}, 3\text{e}{-4}, 1\text{e}{-4}\}$ and $\mathfrak{K} = \{20000, 30000, 40000, 50000, 60000\}$. We compare the RMSE values using C-LR and PC-LR in Figure 4 for Example 1. A similar behavior is observed for Example 2, see Figure 20 in Appendix A.

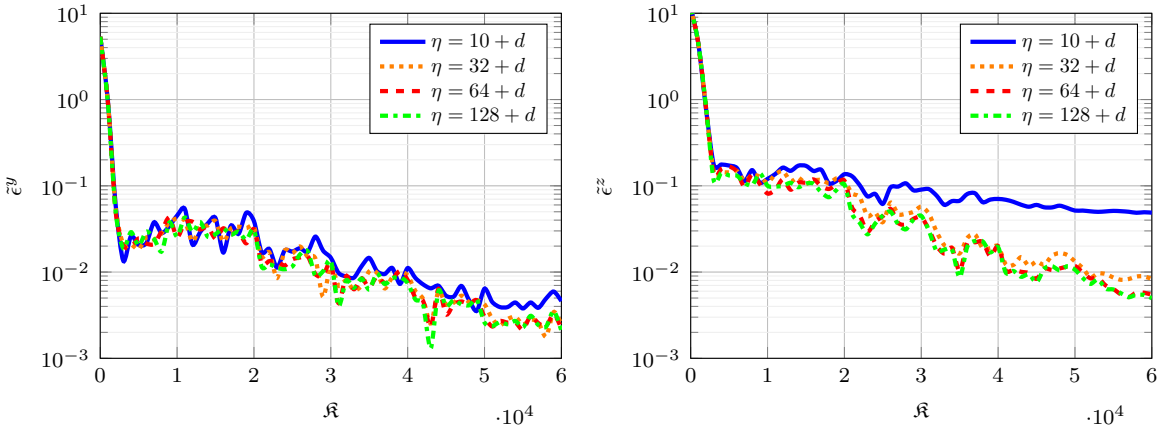
Next, we consider the model error. To try reduce the model error, one can increase the number of hidden neurons η or the number of hidden layers L. We report the RMSE values for $\eta \in \{10+d, 32+d, 64+d, 128+d\}$ in Figures 5 and 6 using PC-LR, for Examples 1 and 2, respectively. We observe that for Example 1 the RMSE decreases when increasing η, but this trend only persists until $\eta = 64 + d$. This is not the case for Example 2. However, note that $d = 50$ in Example 2, i.e., the starting value of η is 60, which may explain why the RMSE did not decrease as observed in Example 1.

Finally, we consider the discretization error, which appears to be larger than the model error. To visualize this, we use the PC-LR approach while setting $\eta = 128 + d$, and plot the RMSE values



(a) RMSE values for Y_0. (b) RMSE values for Z_0.

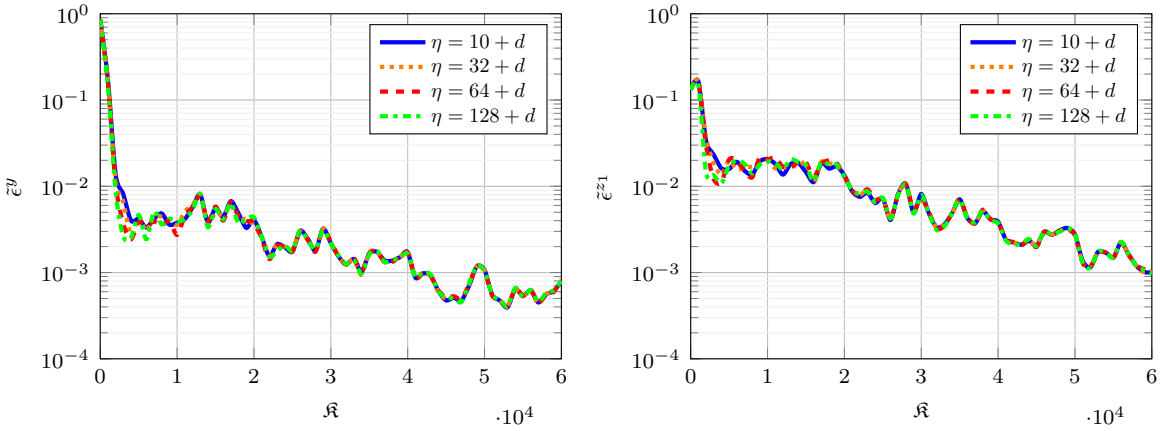
Figure 4: RMSE values are plotted for Example 1 using different learning rate approaches, where $T = 1, K = 100, S_0 = 100, a = 0.05, b = 0.2, R = 0.03$ and $\delta = 0$.



(a) RMSE values for Y_0. (b) RMSE values for Z_0.

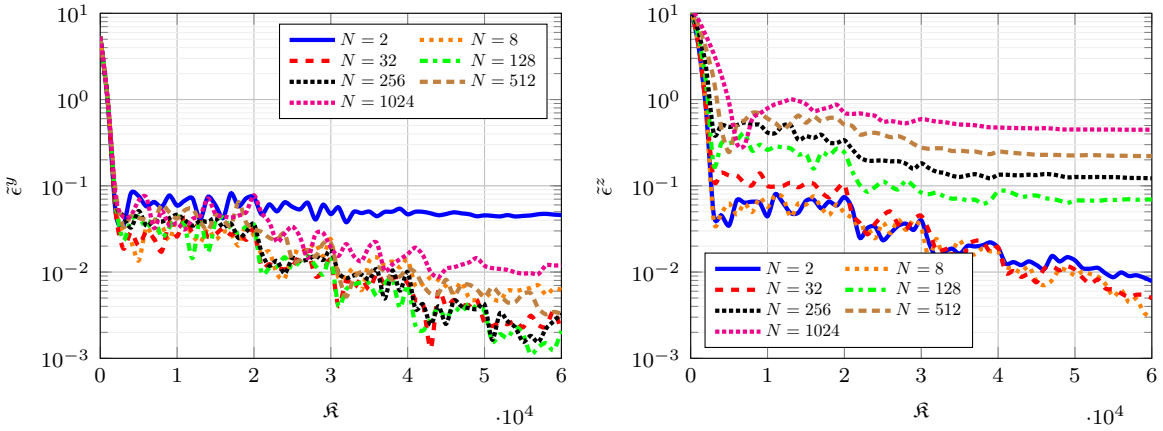
Figure 5: RMSE values are plotted for Example 1 using $\eta \in \{10 + d, 32 + d, 64 + d, 128 + d\}$, where $T = 1, K = 100, S_0 = 100, a = 0.05, b = 0.2, R = 0.03$ and $\delta = 0$.

in Figure 7 for $N \in \{2, 8, 32, 128, 256, 512, 1024\}$ in the case of Example 1. When considering the approximation of Y_0, the RMSE decreases with increasing N until $N = 32$, after which it starts to increase. This phenomenon is even more pronounced for the approximation of Z_0. It is worth noting that approximating Z is generally more challenging than approximating Y for BSDEs. While a higher value of N can reduce the discretization error, it can also lead to a larger error due to the increased number of DNNs and network parameters to be optimized. Additionally, the propagated errors over time in the DBSDE scheme become larger with a higher value of N. A similar behavior is observed for Example 2, see Figure 21 in Appendix A. To provide further clarity, we display the RMSE values and the absolute errors $\tilde{\epsilon}_q^y = |Y_{0,q}^{\Delta,\hat{\theta}^m} - Y_0|$ and $\tilde{\epsilon}_q^z = |Z_{0,q}^{\Delta,\hat{\theta}^m} - Z_0|$ from the q-th run for $N \in \{32, 1024\}$ in Figure 8 for Example 1. The variation of absolute errors from different runs around the corresponding RMSE values indicates that the increase in RMSE in Figure 7 for $N > 32$ is caused mainly by the propagated errors (same is observed for Example 2, see Figure 22 in Appendix A). Note that choosing $\eta > 128 + d$ does not reduce the RMSE. Hence, disentangling the sources of uncertainty is practically challenging, making it essential to quantify the uncertainty of the DBSDE scheme for practical applications.



(a) RMSE values for Y_0. (b) RMSE values for Z_0^1.

Figure 6: RMSE values are plotted for Example 2 using $\eta \in \{10 + d, 32 + d, 64 + d, 128 + d\}$, where $T = 0.25$ and $b = 25$.

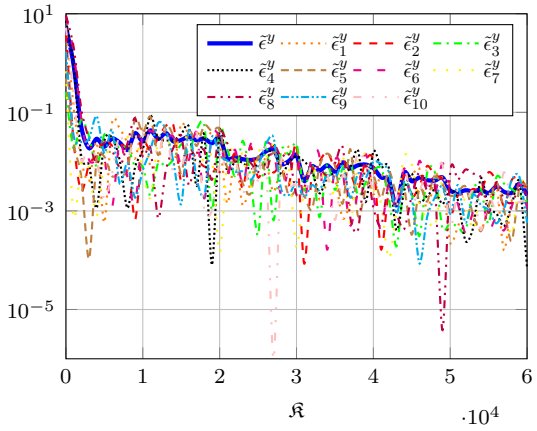


(a) RMSE values for Y_0. (b) RMSE values for Z_0.

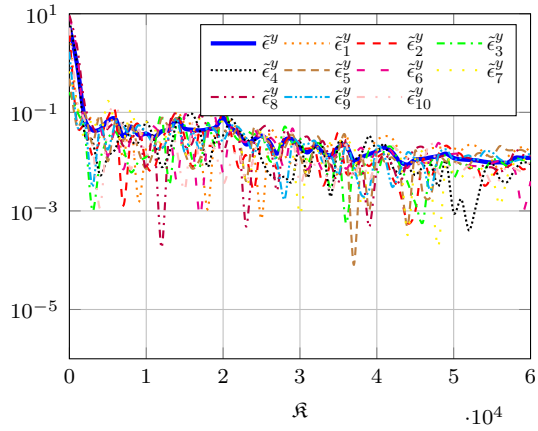
Figure 7: RMSE values are plotted for Example 1 using $N \in \{2, 8, 32, 128, 256, 512, 1024\}$, where $T = 1, K = 100, S_0 = 100, a = 0.05, b = 0.2, R = 0.03$ and $\delta = 0$.

4.3 Performance of the UQ model

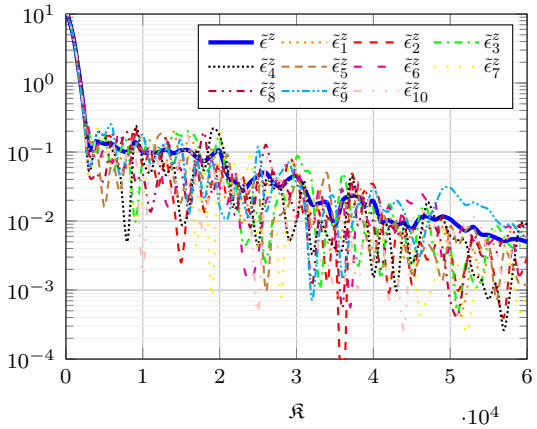
In this section, we evaluate the quality of the mean and STD of the approximate solution estimated from our UQ model for both the DBSDE and LaDBSDE schemes. We start with the DBSDE scheme and consider Examples 1 and 2. In each example, the following structure is used. Firstly, the dataset for training and evaluating the UQ model is generated. To gain insights into the RMSE and ensemble STD, we visualize these values and determine an appropriate metric for assessing the accuracy of the ensemble STD in approximating the RMSE. Secondly, we focus on evaluating the accuracy of the estimated STD from the UQ model in approximating the RMSE, and compare it with the performance of the ensemble STD. Additionally, we determine the number of DBSDE runs that the estimated STD achieves the same quality as the ensemble STD. The computational cost for training the UQ model is assessed in terms of the number of DBSDE runs needed to train it. Finally, we evaluate the accuracy of the estimated mean from the UQ model in approximating the exact solution, and compare it with the performance of the ensemble mean. Regarding the LaDBSDE scheme, only Example 2 is analyzed to show the applicability of the UQ model, following a similar structure as for the DBSDE scheme.



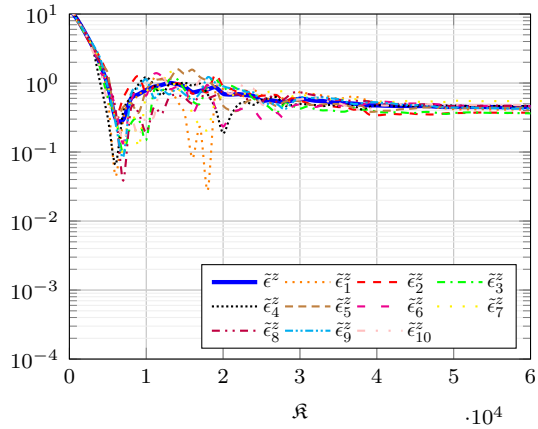
(a) RMSE values and the absolute error from each of $Q = 10$ DBSDE runs for Y_0 with $N = 32$.



(b) RMSE values and the absolute error from each of $Q = 10$ DBSDE runs for Y_0 with $N = 1024$.



(c) RMSE values and the absolute error from each of $Q = 10$ DBSDE runs for Z_0 with $N = 32$.



(d) RMSE values and the absolute error from each of $Q = 10$ DBSDE runs for Z_0 with $N = 1024$.

Figure 8: RMSE values and the absolute errors from each of $Q = 10$ DBSDE runs are plotted for Example 1 using $N \in \{32, 1024\}$, where $T = 1, K = 100, S_0 = 100, a = 0.05, b = 0.2, R = 0.03$ and $\delta = 0$.

4.3.1 The DBSDE scheme for the Black-Scholes example

The dataset $\mathcal{D} = \{\mathbf{x}_i, \mathbf{y}_i, \mathbf{z}_i\}_{i=1}^M$ for the UQ model is generated using Algorithm 1, where $\mathbf{x}_i$ includes the parameter set of the Black-Scholes BSDE, i.e. $\mathbf{x}_i = (a_i, b_i, S_{0,i}, R_i, \delta_i, K_i, T_i)$. Note that $\mathbf{y}_i = Y_0^{\Delta,\hat{\theta}^m}(\mathbf{x}_i)$ and $\mathbf{z}_i = Z_0^{\Delta,\hat{\theta}^m}(\mathbf{x}_i)$ are the approximated solutions given by the DBSDE algorithm for the parameter set $\mathbf{x}_i$ of the Black-Scholes BSDE. The parameter set $\mathbf{x}_i$ is generated using uniform distributions, where the bounds are chosen to account for all different scenarios of a European (call) option, namely in the money (ITM), at the money (ATM), and out of the money (OTM). To calculate $(\mathbf{y}_i, \mathbf{z}_i)$, we set $\mathfrak{K} = 30000$, $\alpha = 1\text{e}{-2}$ and $m = 128$ for the DBSDE algorithm. We generate 3 datasets where we treat T and N differently for each dataset. In dataset $\mathcal{D}_1$, we fix the values of T and N. In dataset $\mathcal{D}_2$, we vary T, but we keep the step size Δt fixed. In the last dataset $\mathcal{D}_3$, both T and Δt vary. This way, we analyze the performance of our UQ model in the case where the maturity value and discretization error vary. Additionally, we analyze the cases where the maturity value or the discretization error is fixed. The range of values for the parameters on each dataset is given in Table 1, where we choose $a = 0.05, \delta = 0$, and $K = 100$. We consider $M = 2560$ different parameter sets and run the DBSDE algorithm $Q = 10$ times for each parameter set. Thus, we construct a dataset of length $M = 2560$ for the

Dataset	Parameter range					
	b	S_0	R	T	N	Δt
$\mathcal{D}_1$	$[0.1, 0.4]$	$[K-20, K+20]$	$[0.001, 0.1]$	0.25	10	0.025
$\mathcal{D}_2$	$[0.1, 0.4]$	$[K-20, K+20]$	$[0.001, 0.1]$	$[\frac{1}{12}, 1]$	$\frac{T}{\Delta t}$	0.025
$\mathcal{D}_3$	$[0.1, 0.4]$	$[K-20, K+20]$	$[0.001, 0.1]$	$[\frac{1}{12}, 1]$	16	$\frac{T}{N}$

Table 1: Parameter range for Example 1.

UQ model by selecting the parameter set and the corresponding approximated solutions from the first run of the DBSDE algorithm. We also calculate benchmark values such as the RMSE, ensemble mean, and ensemble STD for each parameter set. Note that the datasets $\mathcal{D}_\mathrm{j}$ are split into $\mathcal{D}_\mathrm{j}^y$ and $\mathcal{D}_\mathrm{j}^z$ for $\mathrm{j} = 1, 2, 3$ in order to built the UQ model for Y_0 and Z_0. To gain insights into the benchmark values (the RMSE and ensemble STD), we display these values in Figure 9 and 10, sorted by the value of the exact solution (Y_0 and Z_0 respectively) for each dataset in the log-domain. Since the exact solution is different for each parameter set $\mathbf{x}$, we also display their relative estimates, where,

$$\tilde{\epsilon}^{y,r}(\mathbf{x}) := \frac{\tilde{\epsilon}^y(\mathbf{x})}{|Y_0(\mathbf{x})|}, \quad \tilde{\sigma}^{y,r}(\mathbf{x}) := \frac{\tilde{\sigma}^y(\mathbf{x})}{|\tilde{\mu}^y(\mathbf{x})|},$$

are the corresponding relative estimates for Y_0 and

$$\tilde{\epsilon}^{z,r}(\mathbf{x}) := \frac{\tilde{\epsilon}^z(\mathbf{x})}{|Z_0(\mathbf{x})|}, \quad \tilde{\sigma}^{z,r}(\mathbf{x}) := \frac{\tilde{\sigma}^z(\mathbf{x})}{|\tilde{\mu}^z(\mathbf{x})|},$$

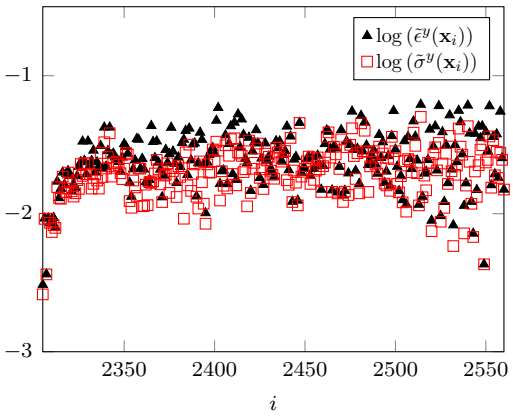
are the corresponding relative measures for Z_0. Note that we show only the last 256 out of 2560 values for better visualization. The RMSE, ensemble STD, and also their relative values exhibit a strong positive correlation. Hence, we use the correlation to quantify the strength of the relationship between the RMSE and ensemble STD values. The correlation values in the log-domain are reported in Table 2. Note that all the following calculations for evaluating the estimated STD from the UQ model are conducted in the log-domain throughout this section. We observe that the relative values provide a more reasonable measure than the absolute ones as the exact solution varies for each parameter set $\mathbf{x}$. Therefore, the relative values are used to evaluate the performance of the UQ model in estimating the STD of the approximate solution. Furthermore,

		Dataset		
Measure	Correlation	$\mathcal{D}_1$	$\mathcal{D}_2$	$\mathcal{D}_3$
Absolute measure for Y_0	$\rho\left(\log(\tilde{\epsilon}^y(\mathbf{x})), \log(\tilde{\sigma}^y(\mathbf{x}))\right)$	0.9207	0.8960	0.8910
Relative measure for Y_0	$\rho\left(\log(\tilde{\epsilon}^{y,r}(\mathbf{x})), \log(\tilde{\sigma}^{y,r}(\mathbf{x}))\right)$	0.9862	0.9804	0.9797
Absolute measure for Z_0	$\rho\left(\log(\tilde{\epsilon}^z(\mathbf{x})), \log(\tilde{\sigma}^z(\mathbf{x}))\right)$	0.7281	0.6903	0.6847
Relative measure for Z_0	$\rho\left(\log(\tilde{\epsilon}^{z,r}(\mathbf{x})), \log(\tilde{\sigma}^{z,r}(\mathbf{x}))\right)$	0.9538	0.9040	0.9191

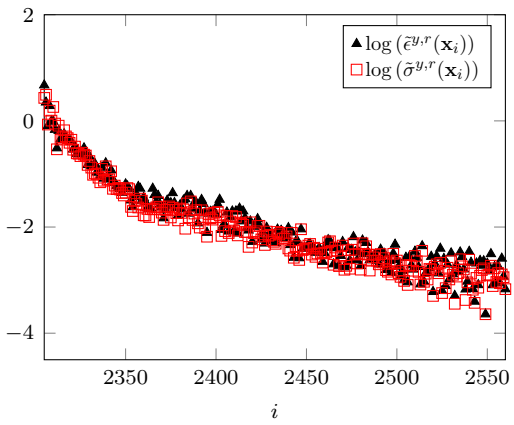
Table 2: Correlation between the RMSE and ensemble STD values, and their relative values for $\mathcal{D}_\mathrm{j}$, $\mathrm{j} = 1, 2, 3$ in Example 1.

we find that the cases with high relative RMSE values in Figure 9 and 10 correspond to deep OTM options, for which the DBSDE algorithm may produce negative estimates of the option price Y_0 or its delta hedging strategy Z_0, indicating divergence. The number of such cases is given in Table 3.

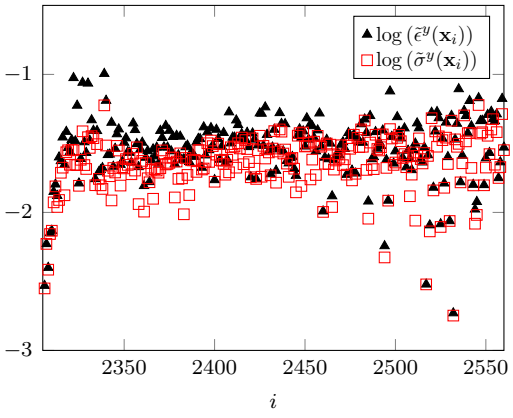
To estimate the STD and mean of the approximate solution for Y_0 and Z_0, we use Algorithm 2 and 3, respectively. Note that there is no evidence against the normality assumption for our UQ model, see Appendix B. The datasets $\mathcal{D}_\mathrm{j}$ are split into training, validation, and testing samples, where we set $M^{valid} = M^{test} = 256$, and the rest for training,



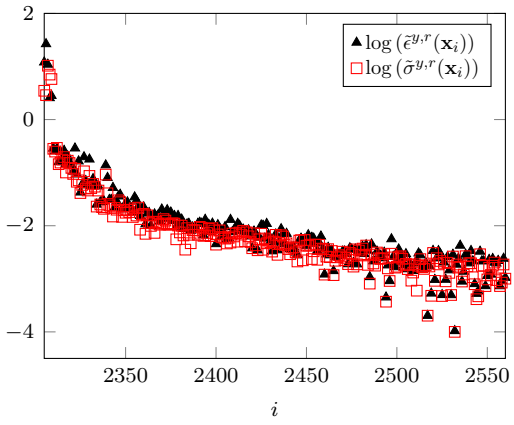
(a) Estimates for $\mathcal{D}_1^y$.



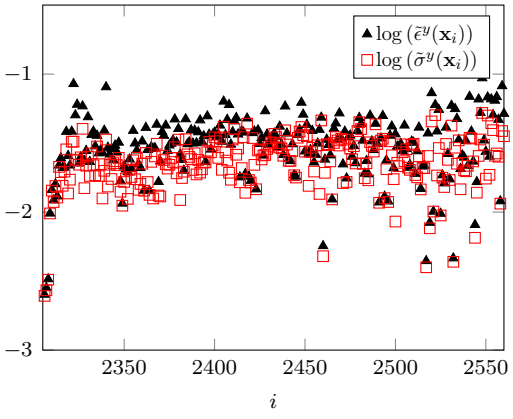
(b) Relative estimates for $\mathcal{D}_1^y$.



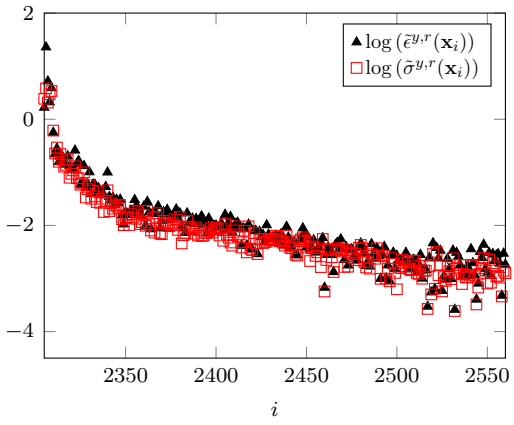
(c) Estimates for $\mathcal{D}_2^y$.



(d) Relative estimates for $\mathcal{D}_2^y$.



(e) Estimates for $\mathcal{D}_3^y$.

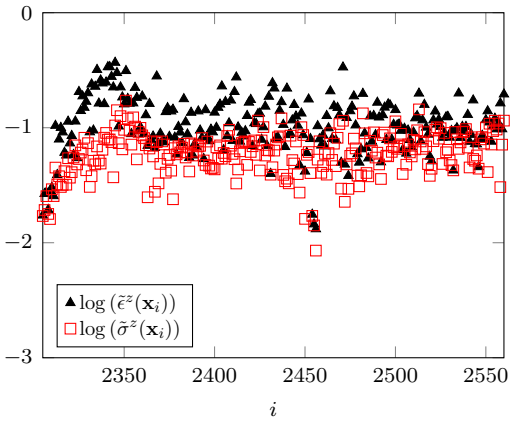


(f) Relative estimates for $\mathcal{D}_3^y$.

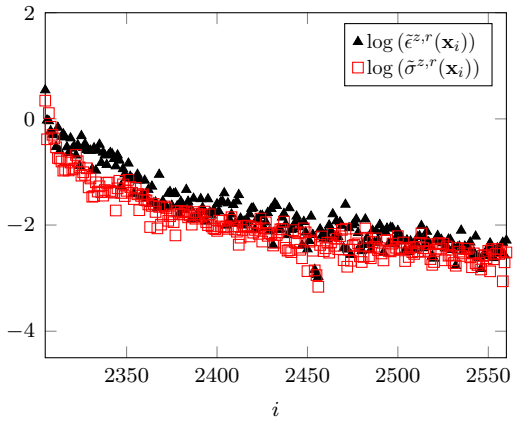
Figure 9: RMSE, the ensemble STD and their relative estimates for $\mathcal{D}_\mathrm{j}^y$, $\mathrm{j} = 1, 2, 3$ sorted by the value of the exact solution in Example 1.

	Dataset		
Condition for divergence	$\mathcal{D}_1$	$\mathcal{D}_2$	$\mathcal{D}_3$
$Y_0^{\Delta,\hat{\theta}^m}(\mathbf{x}) < 0$ or $Z_0^{\Delta,\hat{\theta}^m}(\mathbf{x}) < 0$	52	35	41

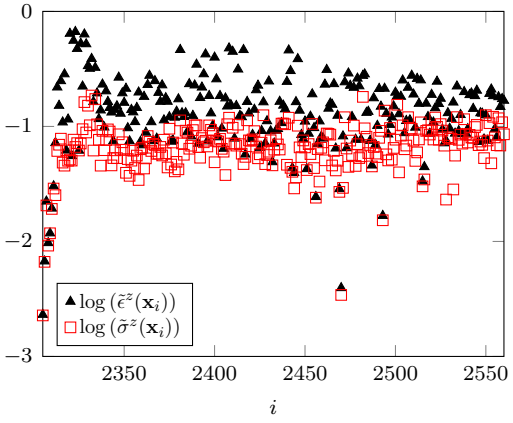
Table 3: Number of diverged cases of the DBSDE scheme for $\mathcal{D}_\mathrm{j}$, $\mathrm{j} = 1, 2, 3$ in Example 1.



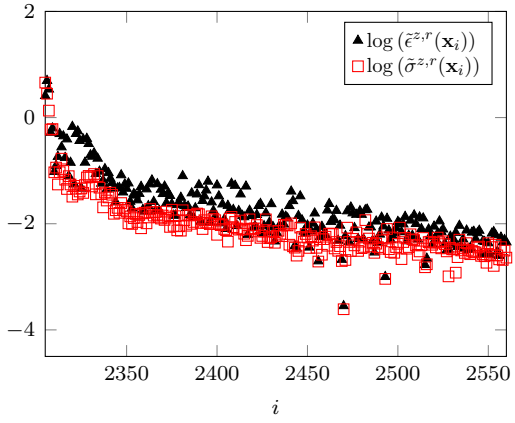
(a) Estimates for $\mathcal{D}_1^z$.



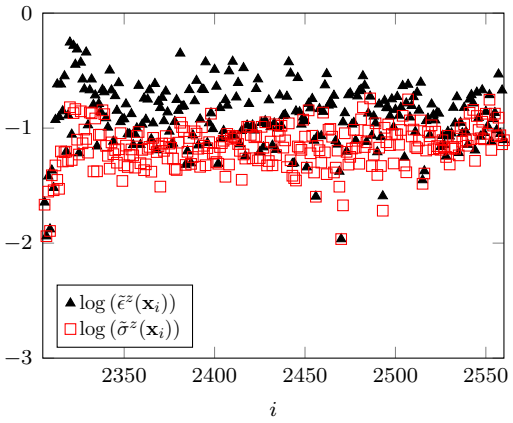
(b) Relative estimates for $\mathcal{D}_1^z$.



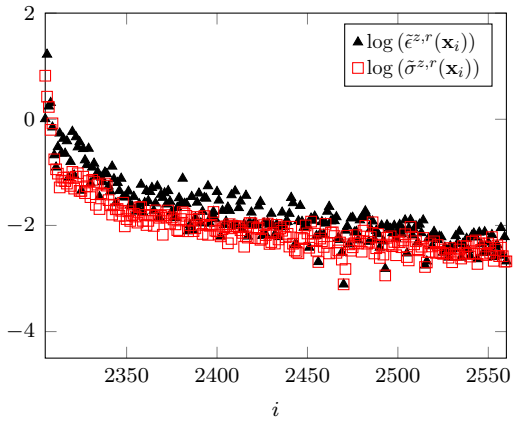
(c) Estimates for $\mathcal{D}_2^z$.



(d) Relative estimates for $\mathcal{D}_2^z$.



(e) Estimates for $\mathcal{D}_3^z$.



(f) Relative estimates for $\mathcal{D}_3^z$.

Figure 10: RMSE, the ensemble STD and their relative estimates for $\mathcal{D}_{\mathfrak{j}}^z$, $\mathfrak{j} = 1, 2, 3$ sorted by the value of the exact solution in Example 1.

$M^{train} = M - M^{valid} - M^{test}$. To account for deviations in our results, we repeat each experiment 10 times, training 10 different UQ models, and provide the mean as well as the STD. For $\mathcal{D}_1$, $\mathfrak{n} = 3$, since $\mathbf{x}_i = (b_i, S_{0,i}, R_i)$ is the parameter set, $\mathfrak{n} = 5$ for $\mathcal{D}_2$ and $\mathcal{D}_3$ where (T, N) and $(T, \Delta t)$ are also varied, respectively. We use $\eta^y = \eta^z = 128$ and $L^y = L^z = 2$. The learning rate α, number of epochs ep, batch size B, and L2 regularization parameter λ are tuned. Based on the performance of the UQ model in the validation sample, the fine-tuned hyper-

parameters for Y_0 in each dataset are: $B^y = 128$, $\lambda^y = 3\text{e}{-2}$, and a PC-LR approach with $\alpha^y = \{1\text{e}{-3}, 3\text{e}{-4}, 1\text{e}{-4}, 3\text{e}{-5}, 1\text{e}{-5}\}$ and $ep^y = \{1000, 100, 100, 100, 100\}$. For Z_0, the fine-tuned hyperparameters are: $B^z = 128$, $\lambda^z = 1\text{e}{-2}$, and the same PC-LR approach as for Y_0. Note that all the following results are shown for the testing sample. To evaluate the quality of the estimated STD, we report in Table 4 the correlation between the relative RMSE and ensemble STD values $\rho\left(\log(\tilde{\epsilon}^r(\mathbf{x})), \log(\tilde{\sigma}^r(\mathbf{x}))\right)$, as well as the mean correlation between the relative RMSE and estimated STD values $\overline{\rho}\left(\log(\tilde{\epsilon}^r(\mathbf{x})), \log(\hat{\sigma}^r(\mathbf{x}))\right)$, where the averaging corresponds to the number of repetitions of our experiment computed by

$$\overline{\rho}\left(\log(\tilde{\epsilon}^{y,r}(\mathbf{x})), \log(\hat{\sigma}^{y,\hat{\omega}^M,r}(\mathbf{x}))\right) := \frac{1}{10}\sum_{\mathfrak{i}=1}^{10} \rho\left(\log(\tilde{\epsilon}^{y,r}(\mathbf{x})), \log(\hat{\sigma}^{y,\hat{\omega}_{\mathfrak{i}}^M,r}(\mathbf{x}))\right),$$

$$\overline{\rho}\left(\log(\tilde{\epsilon}^{z,r}(\mathbf{x})), \log(\hat{\sigma}^{z,\hat{\psi}^M,r}(\mathbf{x}))\right) := \frac{1}{10}\sum_{\mathfrak{i}=1}^{10} \rho\left(\log(\tilde{\epsilon}^{z,r}(\mathbf{x})), \log(\hat{\sigma}^{z,\hat{\psi}_{\mathfrak{i}}^M,r}(\mathbf{x}))\right).$$

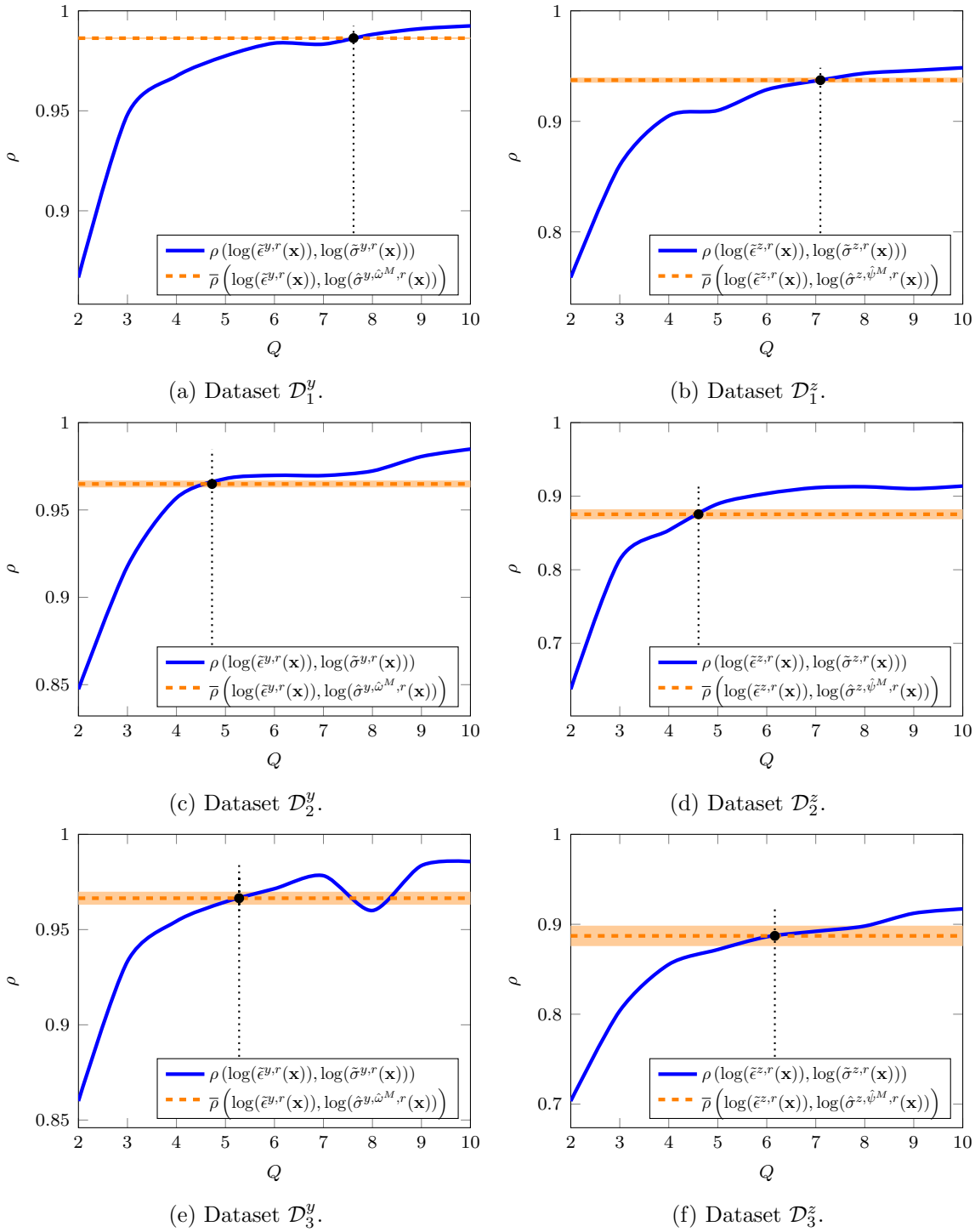
Moreover, $\hat{\sigma}^{y,\hat{\omega}^M,r}(\mathbf{x}))$ and $\hat{\sigma}^{z,\hat{\psi}^M,r}(\mathbf{x}))$ represents the estimated relative STD values from 10 different trained UQ models for Y_0 and Z_0, respectively. The index $\mathfrak{i}$ corresponds to the values estimated from the $\mathfrak{i}$-th trained UQ model. The STD of the correlation is given in the brackets. The correlation values for the relative ensemble STD and the mean correlation values for the

		Dataset		
UQ approach	Metric	$\mathcal{D}_1$	$\mathcal{D}_2$	$\mathcal{D}_3$
Ensemble for Y_0	$\rho\left(\log(\tilde{\epsilon}^{y,r}(\mathbf{x})), \log(\tilde{\sigma}^{y,r}(\mathbf{x}))\right)$	0.9925	0.9848	0.9857
UQ model for Y_0	$\overline{\rho}\left(\log(\tilde{\epsilon}^{y,r}(\mathbf{x})), \log(\hat{\sigma}^{y,\hat{\omega}^M,r}(\mathbf{x}))\right)$	0.9863 (0.0004)	0.9649 (0.0020)	0.9665 (0.0034)
Ensemble for Z_0	$\rho\left(\log(\tilde{\epsilon}^{z,r}(\mathbf{x})), \log(\tilde{\sigma}^{z,r}(\mathbf{x}))\right)$	0.9485	0.9138	0.9171
UQ model for Z_0	$\overline{\rho}\left(\log(\tilde{\epsilon}^{z,r}(\mathbf{x})), \log(\hat{\sigma}^{z,\hat{\psi}^M,r}(\mathbf{x}))\right)$	0.9373 (0.0023)	0.8754 (0.0068)	0.8871 (0.0113)

Table 4: Correlation between the relative RMSE and ensemble STD values, and the mean correlation between the relative RMSE and estimated STD values from the UQ model for $\mathcal{D}_\mathfrak{j}$, $\mathfrak{j} = 1, 2, 3$ using the testing sample in Example 1. The STD of the correlation is given in the brackets.

relative estimated STD from our UQ model are very close for Y_0 and Z_0. This demonstrates that the relative estimated STD effectively approximates the relative ensemble STD. Moreover, we determine the number of runs of the DBSDE algorithm for which the relative ensemble STD is approximately equal to the relative estimated STD. Therefore, we display in Figure 11 the correlation between the relative RMSE and ensemble STD values $\rho\left(\log(\tilde{\epsilon}^r(\mathbf{x})), \log(\tilde{\sigma}^r(\mathbf{x}))\right)$ for different DBSDE runs, the mean correlation between the relative RMSE and estimated STD values $\overline{\rho}\left(\log(\tilde{\epsilon}^r(\mathbf{x})), \log(\hat{\sigma}^r(\mathbf{x}))\right)$, and their intersection. The shaded area gives the STD of the correlation. We observe that the relative estimated STD from the UQ model is as good as the relative ensemble STD calculated from around $Q = 8$ runs of the DBDSE algorithm for Y_0 and $Q = 7$ for Z_0 in dataset $\mathcal{D}_1$. When the maturity T is also varied in dataset $\mathcal{D}_2$, the relative estimated STD is as good as the relative ensemble STD calculated from around $Q = 5$ runs of the DBDSE algorithm for Y_0 and Z_0. For the last dataset $\mathcal{D}_3$ where the maturity T and the step size Δt are also varied, the relative estimated STD is as good as the relative ensemble STD calculated from around $Q = 6$ runs of the DBSDE algorithm for Y_0 and $Q = 6$ for Z_0. We conclude that using a training dataset of length $M^{train} = 2048$ to train the UQ model, the relative estimated STD can perform as well as the relative ensemble STD of at least $Q = 5$ DBSDE runs.

Using a larger M^{train}, the UQ model is expected to provide a better estimate of the relative STD. However, this increases the computational cost as the number of DBSDE runs needed to



(a) Dataset $\mathcal{D}_1^y$.

(b) Dataset $\mathcal{D}_1^z$.

(c) Dataset $\mathcal{D}_2^y$.

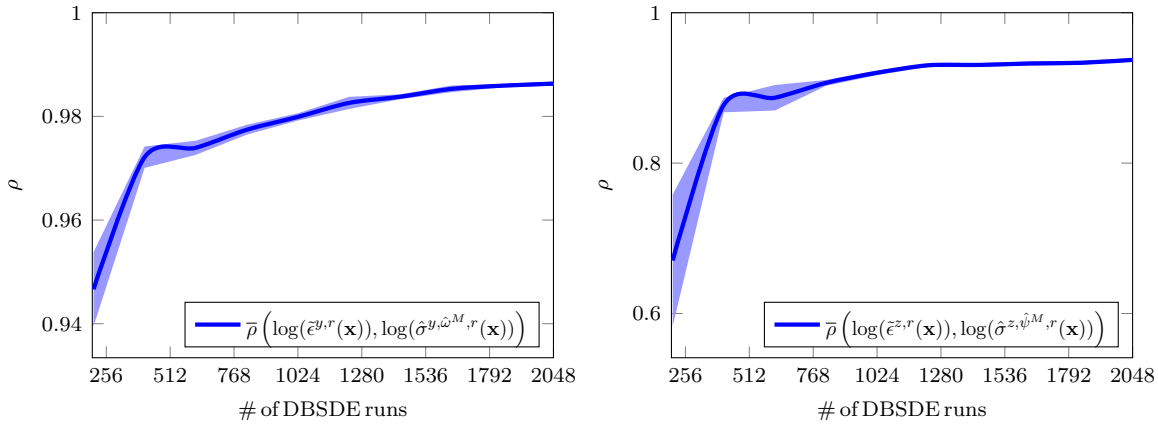
(d) Dataset $\mathcal{D}_2^z$.

(e) Dataset $\mathcal{D}_3^y$.

(f) Dataset $\mathcal{D}_3^z$.

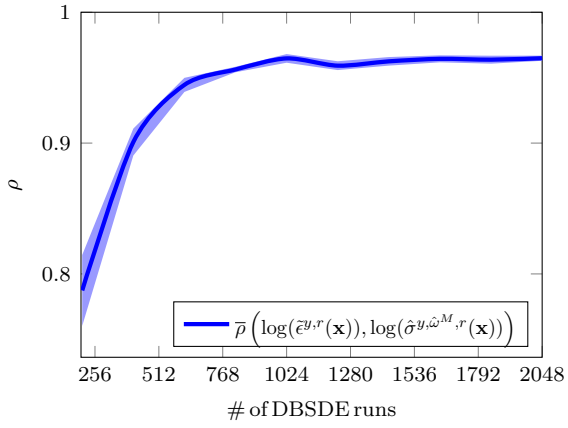
Figure 11: Correlation between the relative RMSE and ensemble STD values for different DBSDE runs, and the mean correlation between the relative RMSE and estimated STD values from the UQ model (STD of correlation given in the shaded area) for $\mathcal{D}_\mathrm{j}$, $\mathrm{j} = 1, 2, 3$ using the testing sample in Example 1. The black dot defines their intersection.

train the UQ model is equal to M^{train}. To show such a trade-off, we display in Figure 12 the mean correlation between the relative RMSE and estimated STD values $\overline{\rho}\left(\log(\tilde{\epsilon}^r(\mathbf{x})), \log(\hat{\sigma}^r(\mathbf{x}))\right)$ while increasing the number of DBSDE runs to train the UQ model from 10% to 100% of M^{train}. We observe that the UQ model can give a good estimate of the relative STD even when trained

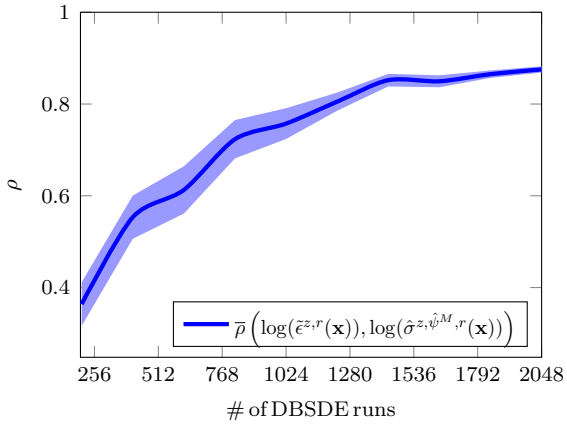


(a) Dataset $\mathcal{D}_1^y$.

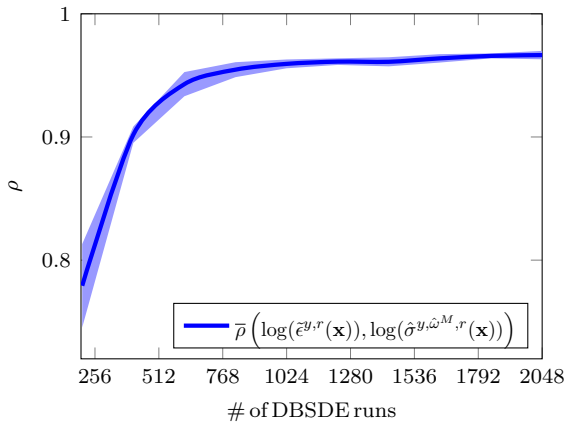
(b) Dataset $\mathcal{D}_1^z$.



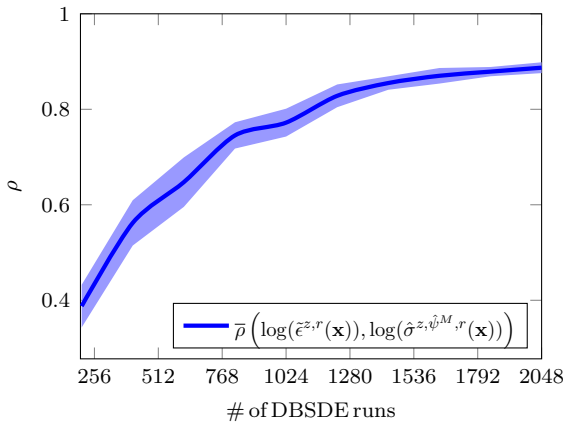
(c) Dataset $\mathcal{D}_2^y$.



(d) Dataset $\mathcal{D}_2^z$.



(e) Dataset $\mathcal{D}_3^y$.



(f) Dataset $\mathcal{D}_3^z$.

Figure 12: Mean correlation between the relative RMSE and estimated STD values from the UQ model while increasing the number of DBSDE runs to train the model from 10% to 100% of M^{train} for $\mathcal{D}_\mathrm{j}$, $\mathrm{j} = 1, 2, 3$ using the testing sample in Example 1. The STD of the correlation is given in the shaded area.

with 1024 DBSDE runs.

Our UQ model does not only estimates the STD of the approximate solution but also its mean. We use the RMSE to measure the quality of the estimated mean compared to the ensemble mean and the expected (exact) solution, which are presented in Table 5. The mean RMSE ($\overline{RMSE}$)

corresponds to the number of repetitions of our experiment computed by

$$\overline{RMSE}\left(Y_0(\mathbf{x}), \hat{\mu}^{y,\hat{\omega}^M}(\mathbf{x})\right) := \frac{1}{10}\sum_{\mathfrak{i}=1}^{10} RMSE\left(Y_0(\mathbf{x}), \hat{\mu}^{y,\hat{\omega}_{\mathfrak{i}}^M}(\mathbf{x})\right),$$

$$\overline{RMSE}\left(Z_0(\mathbf{x}), \hat{\mu}^{z,\hat{\psi}^M}(\mathbf{x})\right) := \frac{1}{10}\sum_{\mathfrak{i}=1}^{10} RMSE\left(Z_0(\mathbf{x}), \hat{\mu}^{z,\hat{\psi}_{\mathfrak{i}}^M}(\mathbf{x})\right),$$

where $\hat{\mu}^{y,\hat{\omega}^M}(\mathbf{x})$ and $\hat{\mu}^{z,\hat{\psi}^M}(\mathbf{x})$ represents the estimated mean values from 10 different trained UQ models for Y_0 and Z_0, respectively. The index $\mathfrak{i}$ corresponds to the values estimated from the $\mathfrak{i}$-th trained UQ model. The STD of the RMSE is given in the brackets. The RMSE between the

		Dataset		
UQ approach	Metric	$\mathcal{D}_1$	$\mathcal{D}_2$	$\mathcal{D}_3$
Ensemble for Y_0	$RMSE\left(Y_0(\mathbf{x}), \tilde{\mu}^y(\mathbf{x})\right)$	1.79e−2	2.36e−2	2.50e−2
UQ model for Y_0	$\overline{RMSE}\left(Y_0(\mathbf{x}), \hat{\mu}^{y,\hat{\omega}^M}(\mathbf{x})\right)$	2.11e−2 (7.73e−4)	3.72e−2 (2.29e−3)	3.95e−2 (1.56e−3)
Ensemble for Z_0	$RMSE\left(Z_0(\mathbf{x}), \tilde{\mu}^z(\mathbf{x})\right)$	1.15e−1	1.83e−1	1.52e−1
UQ model for Z_0	$\overline{RMSE}\left(Z_0(\mathbf{x}), \hat{\mu}^{z,\hat{\psi}^M}(\mathbf{x})\right)$	1.15e−1 (1.62e−3)	1.90e−1 (2.78e−3)	1.63e−1 (3.65e−3)

Table 5: RMSE between the exact solution and ensemble mean values, and the mean RMSE between the exact solution and estimated mean values from the UQ model for $\mathcal{D}_\mathfrak{j}$, $\mathfrak{j} = 1, 2, 3$ using the testing sample in Example 1. The STD of the RMSE is given in the brackets.

exact solution and ensemble mean values $RMSE\left(Y_0(\mathbf{x}), \tilde{\mu}^y(\mathbf{x})\right)$, and the mean RMSE between the exact solution and estimated mean values $\overline{RMSE}\left(Y_0(\mathbf{x}), \hat{\mu}^{y,\hat{\omega}^M}(\mathbf{x})\right)$ are very close. The same can be concluded for Z_0. Hence, the estimated mean given by our UQ model can be used as highly accurate initializers of $(Y_0^{\Delta,\theta}, Z_0^{\Delta,\theta})$ in the DBSDE algorithm, instead of initializing them randomly using uniform distributions.

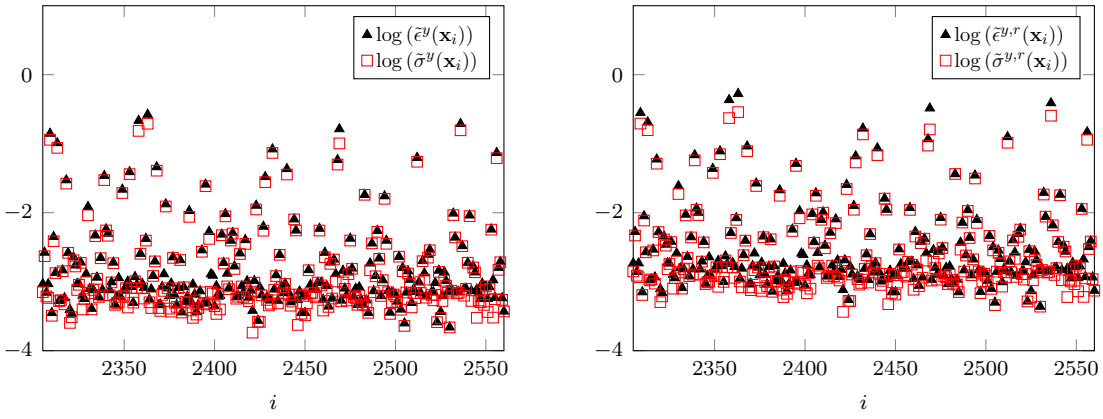
4.3.2 The DBSDE scheme for the Burgers type example

We generate dataset $\mathcal{D} = \{\mathbf{x}_i, \mathbf{y}_i, \mathbf{z}_i\}_{i=1}^M$ using Algorithm 1, where $\mathbf{x}_i$ now includes the parameter set of the Burgers type BSDE, namely $\mathbf{x}_i = (b_i, T_i)$, and $\mathbf{y}_i \in \mathbb{R}$ and $\mathbf{z}_i \in \mathbb{R}^{1\times d}$ the corresponding approximate solution for $Y_0(\mathbf{x}_i)$ and $Z_0(\mathbf{x}_i)$. We keep the same hyperparameter values of the DBSDE algorithm as in Example 1 to generate the dataset $\mathcal{D}$ and consider only varying the parameter set as done for the dataset $\mathcal{D}_3$ in Example 1, namely $(b, T, \Delta t)$ are varied. In Table 6 the range of parameter values is reported. We set again $M = 2560$ and $Q = 10$ for each parameter

Dataset	Parameter range			
	b	T	N	Δt
$\mathcal{D}$	$[0.2, 40]$	$[\frac{1}{12}, 0.3]$	32	$\frac{T}{N}$

Table 6: Parameter range for Example 2.

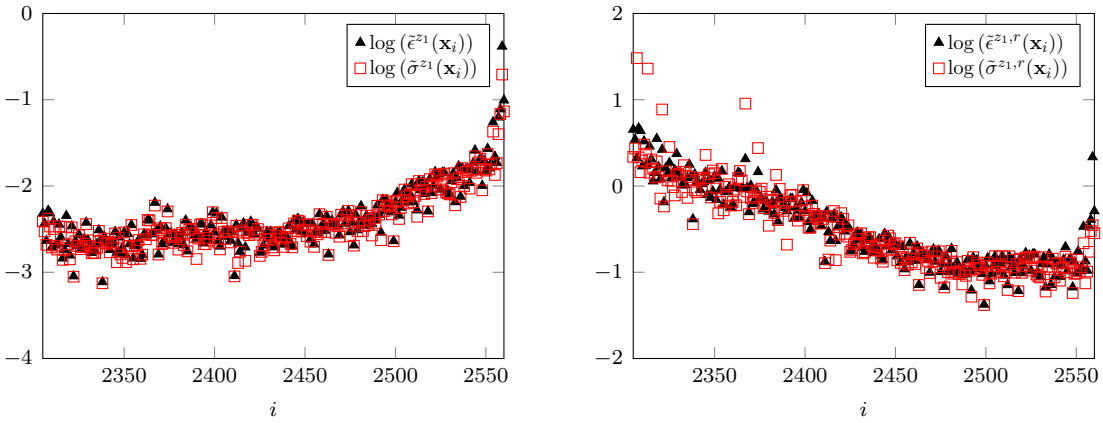
set, and split the dataset $\mathcal{D}$ into $\mathcal{D}^y$ and $\mathcal{D}^z$. The RMSE, ensemble STD, and their relative values for Y_0 and Z_0^1 are displayed in Figures 13 and 14, sorted by the value of the corresponding exact solution. Note that for Z_0, only the results for the first component are shown (with similar behavior observed for other components of Z_0). We find that the DBSDE algorithm produces negative approximations of Z_0 in a significant number of cases (1184 cases), especially for small values of b ($b \approx 0.2$). Additionally, for large values of b and T ($b > 40$ and $T > 0.3$), the



(a) Estimates for $\mathcal{D}^y$.

(b) Relative estimates for $\mathcal{D}^y$.

Figure 13: RMSE, the ensemble STD and their relative estimates for $\mathcal{D}^y$ sorted by the value of the exact solution in Example 2.



(a) Estimates for $\mathcal{D}^z$.

(b) Relative estimates for $\mathcal{D}^z$.

Figure 14: RMSE, the ensemble STD and their relative estimates for $\mathcal{D}^z$ sorted by the value of the exact solution in Example 2.

relative RMSE values become very large. Similar to the previous example, Figures 13 and 14 demonstrate a strong positive correlation among the RMSE, ensemble STD, and their relative values. The correlation values in the log-domain are reported in Table 7 for dataset $\mathcal{D}$.

Measure	Correlation	Dataset $\mathcal{D}$
Absolute measure for Y_0	$\rho\left(\log(\tilde{\epsilon}^{y}(\mathbf{x})), \log(\tilde{\sigma}^{y}(\mathbf{x}))\right)$	0.9887
Relative measure for Y_0	$\rho\left(\log(\tilde{\epsilon}^{y,r}(\mathbf{x})), \log(\tilde{\sigma}^{y,r}(\mathbf{x}))\right)$	0.9881
Absolute measure for Z_0^1	$\rho\left(\log(\tilde{\epsilon}^{z_1}(\mathbf{x})), \log(\tilde{\sigma}^{z_1}(\mathbf{x}))\right)$	0.9889
Relative measure for Z_0^1	$\rho\left(\log(\tilde{\epsilon}^{z_1,r}(\mathbf{x})), \log(\tilde{\sigma}^{z_1,r}(\mathbf{x}))\right)$	0.9269

Table 7: Correlation between the RMSE and ensemble STD values, and their relative values for $\mathcal{D}$ in Example 2.

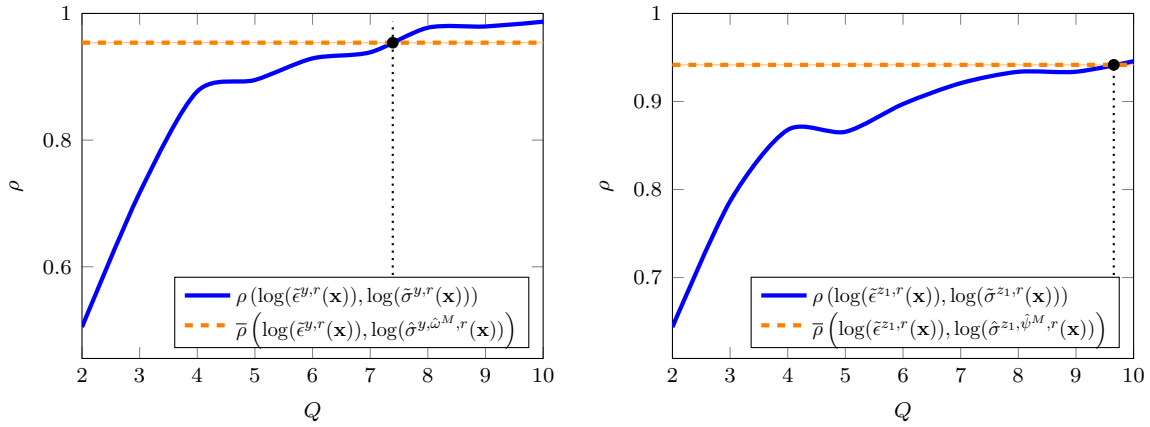
To train the UQ model, we follow the same procedure as in Example 1. We again choose a testing and validation sample of 256 and use the rest for training the UQ model. Note that $\mathfrak{n} = 3$ since $\mathbf{x}_i = (b_i, T_i, \Delta t_i)$. Based on the validation sample, the fine-tuned hyperparameters for Y_0 are as follows: $B^y = 32$, $\lambda^y = 1\text{e}{-3}$, and a PC-LR approach with $\alpha^y = \{1\text{e}{-3}, 3\text{e}{-4}, 1\text{e}{-4}, 3\text{e}{-5}, 1\text{e}{-5}\}$

and $ep^y = \{5000, 500, 500, 500, 500\}$. For Z_0^1, the fine-tuned hyperparameters are: $B^z = 128$, $\lambda^z = 3\text{e}{-2}$, and a PC-LR approach with $\alpha^z = \alpha^y$ and $ep^z = \{1000, 100, 100, 100, 100\}$. In Table 8, we present the correlation between the relative RMSE and ensemble STD values $\rho\left(\log(\tilde{\epsilon}^r(\mathbf{x})), \log(\tilde{\sigma}^r(\mathbf{x}))\right)$, as well as the mean correlation between the relative RMSE and estimated STD values $\overline{\rho}\left(\log(\tilde{\epsilon}^r(\mathbf{x})), \log(\hat{\sigma}^r(\mathbf{x}))\right)$. The correlation values for the relative ensemble

UQ approach	Metric	Dataset $\mathcal{D}$
Ensemble for Y_0	$\rho\left(\log(\tilde{\epsilon}^{y,r}(\mathbf{x})), \log(\tilde{\sigma}^{y,r}(\mathbf{x}))\right)$	0.9870
UQ model for Y_0	$\overline{\rho}\left(\log(\tilde{\epsilon}^{y,r}(\mathbf{x})), \log(\hat{\sigma}^{y,\hat{\omega}^M,r}(\mathbf{x}))\right)$	0.9538 (0.0011)
Ensemble for Z_0^1	$\rho\left(\log(\tilde{\epsilon}^{z_1,r}(\mathbf{x})), \log(\tilde{\sigma}^{z_1,r}(\mathbf{x}))\right)$	0.9457
UQ model for Z_0^1	$\overline{\rho}\left(\log(\tilde{\epsilon}^{z_1,r}(\mathbf{x})), \log(\hat{\sigma}^{z_1,\hat{\psi}^M,r}(\mathbf{x}))\right)$	0.9416 (0.0008)

Table 8: Correlation between the relative RMSE and ensemble STD values, and the mean correlation between the relative RMSE and estimated STD values from the UQ model for $\mathcal{D}$ using the testing sample in Example 2. The STD of the correlation is given in the brackets.

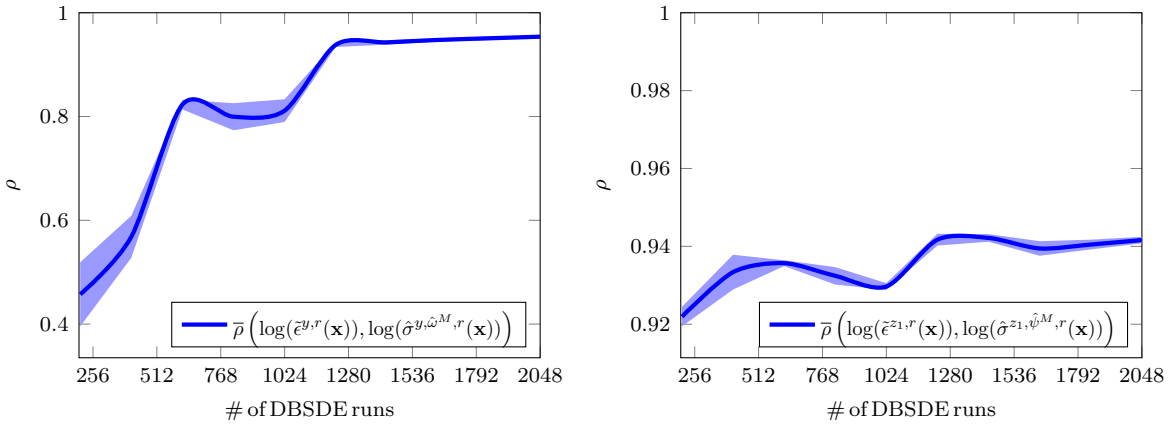
STD and the mean correlation values for the relative estimated STD from our UQ model are very close for Y_0 and Z_0^1, indicating that our UQ model can provide highly accurate estimates of the relative ensemble STD also in high dimensions. In Figure 15, we display the correla-



(a) Dataset $\mathcal{D}^y$. (b) Dataset $\mathcal{D}^z$.

Figure 15: Correlation between the relative RMSE and ensemble STD values for different DBSDE runs, and the mean correlation between the relative RMSE and estimated STD values from the UQ model (STD of correlation given in the shaded area) for $\mathcal{D}$ using the testing sample in Example 2. The black dot defines their intersection.

tion between the relative RMSE and ensemble STD values $\rho\left(\log(\tilde{\epsilon}^r(\mathbf{x})), \log(\tilde{\sigma}^r(\mathbf{x}))\right)$ for different DBSDE runs, the mean correlation between the relative RMSE and estimated STD values $\overline{\rho}\left(\log(\tilde{\epsilon}^r(\mathbf{x})), \log(\hat{\sigma}^r(\mathbf{x}))\right)$, and their intersection. The relative estimated STD from the UQ model achieves the quality of the relative ensemble STD calculated from around $Q = 8$ runs of the DBDSE algorithm for Y_0 and around $Q = 10$ for Z_0^1. In this example, the performance of the UQ model improves compared to the previous example. One possible explanation for this improvement is the exact solution in this example, which remains the same for Y_0 and slightly varies for Z_0 across different parameter sets $\mathbf{x}$, namely Example 2 is less challenging than Example 1.
To show the computation cost of generating the training data for the UQ model, we display in Figure 16 the mean correlation between the relative RMSE and estimated STD values $\overline{\rho}\left(\log(\tilde{\epsilon}^r(\mathbf{x})), \log(\hat{\sigma}^r(\mathbf{x}))\right)$ while increasing the number of DBSDE runs to train the model. Even



(a) Dataset $\mathcal{D}^y$. (b) Dataset $\mathcal{D}^z$.

Figure 16: Mean correlation between the relative RMSE and estimated STD values from the UQ model while increasing the number of DBSDE runs to train the model from 10% to 100% of M^{train} for $\mathcal{D}$ using the testing sample in Example 2. The STD of the correlation is given in the shaded area.

when training the UQ model with 1024 DBSDE runs, a good estimate of the STD is achieved. Next, we examine the performance of our UQ model for the mean of the approximate solution. We calculate the RMSE between the exact solution and ensemble mean values $RMSE\left(Y_0(\mathbf{x}), \tilde{\mu}^y(\mathbf{x})\right)$, as well as the mean RMSE between the exact solution and estimated mean values $\overline{RMSE}\left(Y_0(\mathbf{x}), \hat{\mu}^{y,\hat{\omega}^M}(\mathbf{x})\right)$ for Y_0, and similarly for Z_0^1. The corresponding values are reported in Table 9. Based on the results, we can conclude that the estimated means given by

UQ approach	Metric	Dataset $\mathcal{D}$
Ensemble for Y_0	$RMSE\left(Y_0(\mathbf{x}), \tilde{\mu}^y(\mathbf{x})\right)$	2.00e−2
UQ model for Y_0	$\overline{RMSE}\left(Y_0(\mathbf{x}), \hat{\mu}^{y,\hat{\omega}^M}(\mathbf{x})\right)$	2.57e−2 (2.22e−3)
Ensemble for Z_0^1	$RMSE\left(Z_0^1(\mathbf{x}), \tilde{\mu}^{z_1}(\mathbf{x})\right)$	2.38e−2
UQ model for Z_0^1	$\overline{RMSE}\left(Z_0^1(\mathbf{x}), \hat{\mu}^{z_1,\hat{\psi}^M}(\mathbf{x})\right)$	1.71e−2 (5.81e−4)

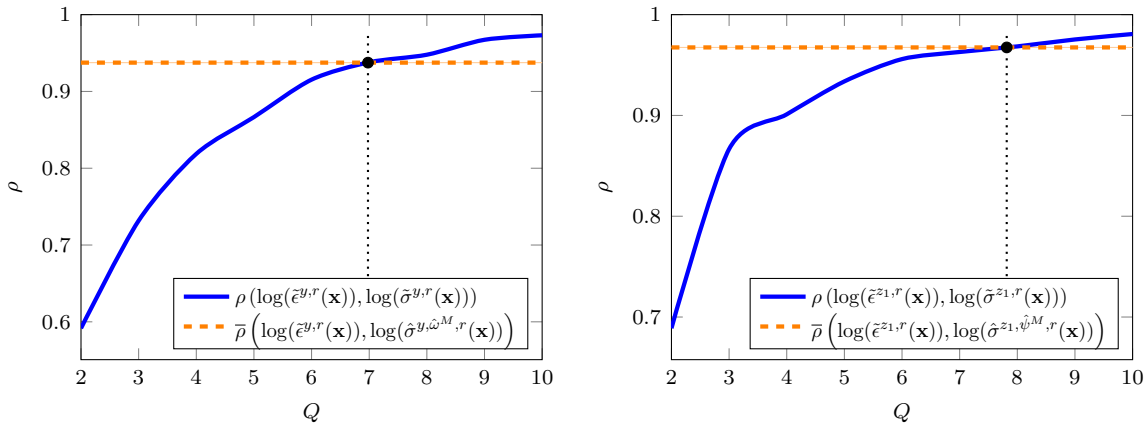
Table 9: RMSE between the exact solution and ensemble mean values, and the mean RMSE between the exact solution and estimated mean values from the UQ model for $\mathcal{D}$ using the testing sample in Example 2. The STD of the RMSE is given in the brackets.

our UQ model can serve as highly accurate initializers for $(Y_0^{\Delta,\theta}, Z_0^{\Delta,\theta})$ in the DBSDE algorithm also in high dimensions.

4.3.3 The LaDBSDE scheme for the Burgers type example

We now demonstrate that the proposed UQ model can be applied to other deep learning-based BSDE schemes, specifically the LaDBSDE scheme [31]. As application, we select the Burgers type BSDE, due to its nonlinearity and higher dimensionality. We choose the hyperparameters for the LaDBSDE scheme similarly to those used in the DBSDE scheme. More precisely, we consider $\mathfrak{K} = 30000$, $\alpha = 1\text{e}{-3}$, $m = 128$, $\eta = 10 + d$, $L = 4$, and $\varrho(x) = \tanh(x)$. Batch normalization is applied after each matrix multiplication and before activation functions. The Adam optimizer is used as an SGD-type algorithm. Using parameter set $\mathbf{x}$ as outlined in Table 6, we apply the LaDBSDE scheme and collect the corresponding approximation of $Y_0(\mathbf{x})$ and $Z_0(\mathbf{x})$.

To distinguish this dataset from the one generated by the DBSDE scheme, we denote it as $\tilde{\mathcal{D}}$. We split the dataset $\tilde{\mathcal{D}}$ into $\tilde{\mathcal{D}}^y$ and $\tilde{\mathcal{D}}^z$ to train and test the UQ model for Y_0 and Z_0, respectively. Using a validation sample of 256, the fine-tuned hyperparameters of the UQ model for Y_0 are: $B^y = 128$, $\lambda^y = 3\text{e}{-2}$, and a PC-LR approach with $\alpha^y = \{1\text{e}{-3}, 3\text{e}{-4}, 1\text{e}{-4}, 3\text{e}{-5}, 1\text{e}{-5}\}$ and $ep^y = \{2000, 400, 400, 400, 400\}$. For Z_0^1, we have $B^z = 128$, $\lambda^z = 1\text{e}{-3}$, and a PC-LR approach with $\alpha^z = \alpha^y$ and $ep^z = \{1000, 500, 500, 500, 500\}$. In Figure 17, we display the correlation between the relative RMSE and ensemble STD values $\rho\left(\log(\tilde{\epsilon}^r(\mathbf{x})), \log(\tilde{\sigma}^r(\mathbf{x}))\right)$ for different LaDBSDE runs, the mean correlation between the relative RMSE and estimated STD values $\overline{\rho}\left(\log(\tilde{\epsilon}^r(\mathbf{x})), \log(\hat{\sigma}^r(\mathbf{x}))\right)$, and their intersection. The relative estimated STD from the UQ model



(a) Dataset $\tilde{\mathcal{D}}^y$.

(b) Dataset $\tilde{\mathcal{D}}^z$.

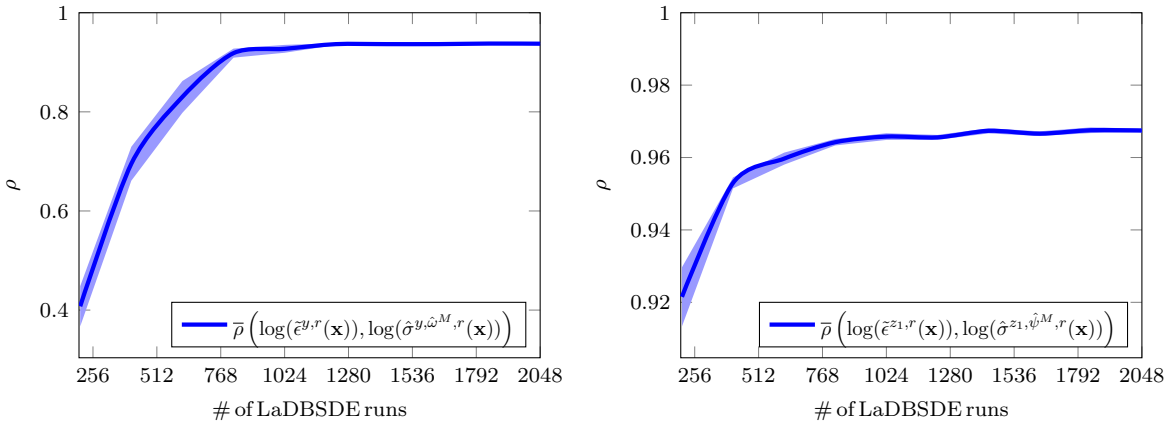
Figure 17: Correlation between the relative RMSE and ensemble STD values for different LaDBSDE runs, and the mean correlation between the relative RMSE and estimated STD values from the UQ model (STD of correlation given in the shaded area) for $\tilde{\mathcal{D}}$ using the testing sample in Example 2. The black dot defines their intersection.

achieves the quality of the relative ensemble STD calculated from $Q = 7$ runs of the LaDBDSE algorithm for Y_0 and around $Q = 8$ for Z_0^1 (similar performance for other components of Z_0). This demonstrates that the UQ model can be applied to other deep learning-based BSDE schemes, which work in a similar manner to the DBSDE scheme. The computational cost to train the UQ model in the case of the LaDBSDE scheme is shown in Figure 18. We can draw the same conclusions as for the DBSDE scheme from the results obtained in this case.
In Table 10, we present the RMSE between the exact solution and ensemble mean values $RMSE\left(Y_0(\mathbf{x}), \tilde{\mu}^y(\mathbf{x})\right)$, as well as the mean RMSE between the exact solution and estimated mean values $\overline{RMSE}\left(Y_0(\mathbf{x}), \hat{\mu}^{y,\hat{\omega}^M}(\mathbf{x})\right)$ for Y_0 (similar for Z_0^1). The RMSE between the exact

UQ approach	Metric	Dataset $\tilde{\mathcal{D}}$
Ensemble for Y_0	$RMSE\left(Y_0(\mathbf{x}), \tilde{\mu}^y(\mathbf{x})\right)$	9.71e−4
UQ model for Y_0	$\overline{RMSE}\left(Y_0(\mathbf{x}), \hat{\mu}^{y,\hat{\omega}^M}(\mathbf{x})\right)$	6.53e−4 (2.02e−5)
Ensemble for Z_0^1	$RMSE\left(Z_0^1(\mathbf{x}), \tilde{\mu}^{z_1}(\mathbf{x})\right)$	1.04e−3
UQ model for Z_0^1	$\overline{RMSE}\left(Z_0^1(\mathbf{x}), \hat{\mu}^{z_1,\hat{\psi}^M}(\mathbf{x})\right)$	8.88e−4 (5.08e−5)

Table 10: RMSE between the exact solution and ensemble mean values, and the mean RMSE between the exact solution and estimated mean values from the UQ model for $\tilde{\mathcal{D}}$ using the testing sample in Example 2. The STD of the RMSE is given in the brackets.

solution and ensemble mean values $RMSE\left(Y_0(\mathbf{x}), \tilde{\mu}^y(\mathbf{x})\right)$, and the mean RMSE between the



(a) Dataset $\tilde{\mathcal{D}}^y$. (b) Dataset $\tilde{\mathcal{D}}^z$.

Figure 18: Mean correlation between the relative RMSE and estimated STD values from the UQ model while increasing the number of LaDBSDE runs from 10% to 100% of M^{train} for $\tilde{\mathcal{D}}$ using the testing sample in Example 2. The STD of the correlation is given in the shaded area.

exact solution and estimated mean values $\overline{RMSE}\left(Y_0(\mathbf{x}), \hat{\mu}^{y,\hat{\omega}^M}(\mathbf{x})\right)$ are very close. The same conclusion can be drawn for Z_0^1. This indicates that our UQ model can also provide highly accurate approximations of the mean of the approximate solution for the LaDBSDE scheme. Moreover, the RMSE values are much smaller than those in Table 9 for Y_0 and Z_0^1. This indicates that our UQ model can identify on average the improved approximations provided by the LaDBSDE scheme compared to the DBSDE scheme.

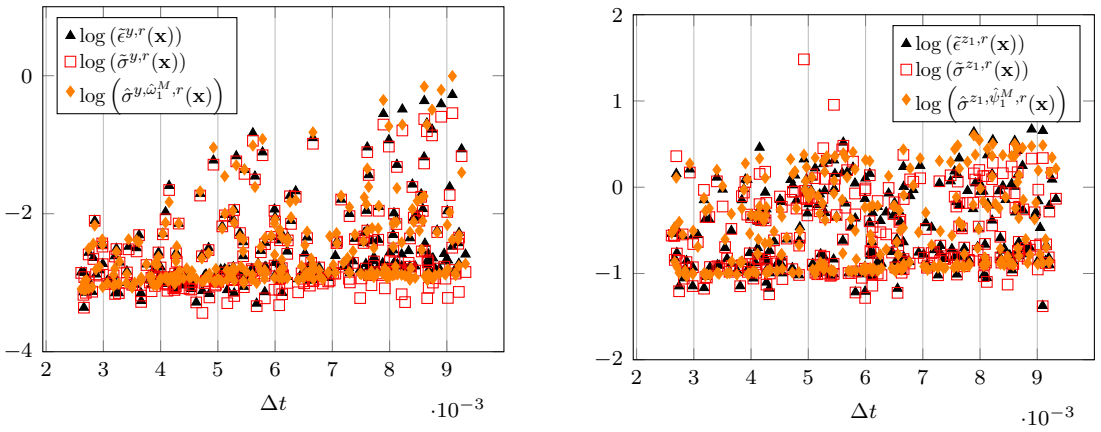
4.4 Practical implications of the UQ model

In this section, we study what sources of uncertainty can be captured by our UQ model and we demonstrate its applicability to downstream tasks.

We start by analyzing the sources of uncertainty that our UQ model can effectively capture. It can be expected that the estimated STD captures uncertainty due to the optimization heuristic as well as the uncertainty due to data sampling. However, it is less clear about the uncertainty stemming from the discretization error, as it might bias the approximations provided by the DBSDE scheme. To illustrate the behavior of the relative RMSE, ensemble STD, and estimated STD values across varying Δt values, we display these measures in Figure 19 using the testing data in dataset $\mathcal{D}$ from Example 2. Note that we use the relative estimated STD from the first trained UQ model (out of our ensemble of 10 models considered for evaluation). As Δt decreases, the bias from the discretization error decreases, and the relative estimated STD improves in approximating the relative RMSE. For larger values of Δt, the bias grows, but the STD also increases. Therefore, the trend of the STD remains consistent with the RMSE, indicating that the relative estimated STD remains reasonable across different values of Δt for approximating the relative RMSE. The same is observed for the relative ensemble STD. To measure the strength and direction of the monotonic relationship between the relative RMSE and estimated STD values across Δt values, we consider Spearman's rank correlation (ς). This metric is calculated as

$$\varsigma\left(\tilde{\epsilon}^r(\mathbf{x}), \hat{\sigma}^r(\mathbf{x})\right) = 1 - \frac{6\sum_{i=1}^{M^{test}}\left(rank(\tilde{\epsilon}^r(\mathbf{x}))_i - rank(\hat{\sigma}^r(\mathbf{x}))_i\right)}{M^{test}\left((M^{test})^2 - 1\right)},$$

where, e.g. $rank(\tilde{\epsilon}^r(\mathbf{x}))_i$ is the assigned rank to $\tilde{\epsilon}^r(\mathbf{x}_i)$. Note that $\varsigma\left(\tilde{\epsilon}^r(\mathbf{x}), \tilde{\sigma}^r(\mathbf{x})\right)$ is calculated similarly. The rank correlation values are displayed in Table 11. The high positive rank corre-



(a) Dataset $\mathcal{D}^y$.

(b) Dataset $\mathcal{D}^z$.

Figure 19: Relative RMSE, ensemble STD and estimated STD values from the UQ model for increasing value of Δt for $\mathcal{D}$ using the testing sample in Example 2.

UQ approach	Rank correlation	Dataset $\mathcal{D}$
Ensemble for Y_0	$\varsigma\left(\tilde{\epsilon}^{y,r}(\mathbf{x}), \tilde{\sigma}^{y,r}(\mathbf{x})\right)$	0.9308
UQ model for Y_0	$\varsigma\left(\tilde{\epsilon}^{y,r}(\mathbf{x}), \hat{\sigma}^{y,\hat{\omega}_1^M,r}(\mathbf{x})\right)$	0.8455
Ensemble for Z_0^1	$\varsigma\left(\tilde{\epsilon}^{z_1,r}(\mathbf{x}), \tilde{\sigma}^{z_1,r}(\mathbf{x})\right)$	0.9778
UQ model for Z_0^1	$\varsigma\left(\tilde{\epsilon}^{z_1,r}(\mathbf{x}), \hat{\sigma}^{z_1,\hat{\psi}_1^M,r}(\mathbf{x})\right)$	0.9352

Table 11: Rank correlation between the relative RMSE, ensemble STD, and estimated STD values for $\mathcal{D}$ using the testing sample in Example 2.

lation values indicate that the relative estimated STD from our UQ model can reflect multiple sources of uncertainty, including the uncertainty caused by the discretization error.
Next, we aim to determine whether our UQ model can detect the enhanced performance of the LaDBSDE scheme over the DBSDE scheme for each parameter set, rather than just considering the average performance as shown before. For this purpose, the accuracy score (acc) for the testing sample of datasets $\mathcal{D}$ and $\tilde{\mathcal{D}}$ of Example 2 is considered. We define binary labels to calculate it. For the relative RMSE, we consider

$$\ell^{\tilde{\epsilon}^r}(\mathbf{x}_i) = \begin{cases} 1 & \text{if } \tilde{\epsilon}^{r,LaDBSDE}(\mathbf{x}_i) < \tilde{\epsilon}^{r,DBSDE}(\mathbf{x}_i), \\ 0 & \text{otherwise,} \end{cases}$$

for the parameter set $\mathbf{x}_i$. Similarly, we define binary labels $\ell^{\tilde{\sigma}^r}(\mathbf{x})$ and $\ell^{\hat{\sigma}^r}(\mathbf{x}_i)$ for the relative ensemble STD and estimated STD, respectively. The accuracy score between the labels of the relative RMSE and estimated STD values represents the number of parameter sets in which the smallest relative RMSE and estimated STD values are achieved from the same scheme, divided by the total number of parameter sets, i.e.

$$acc\left(\ell^{\tilde{\epsilon}^r}(\mathbf{x}), \ell^{\hat{\sigma}^r}(\mathbf{x})\right) = \frac{1}{M^{test}} \sum_{i=1}^{M^{test}} \mathbb{1}_{\ell^{\tilde{\epsilon}^r}(\mathbf{x}_i)=\ell^{\hat{\sigma}^r}(\mathbf{x}_i)}.$$

Similarly, we calculate the accuracy score between the labels of relative RMSE and ensemble STD values $acc\left(\ell^{\tilde{\epsilon}^r}(\mathbf{x}), \ell^{\tilde{\sigma}^r}(\mathbf{x})\right)$ and report them in Table 12. The accuracy score of 1 for Z_0^1 implies that the relative estimated STD from our UQ model illustrates enhanced performance when comparing the DBSDE and LaDBSDE schemes across the entire testing sample. This observation is valid only for approximately 80% of the testing sample for Y_0.

UQ approach	Accuracy score	Datasets $\mathcal{D}$ and $\tilde{\mathcal{D}}$
Ensemble for Y_0	$acc\left(\ell^{\tilde{\epsilon}^{y,r}}(\mathbf{x}), \ell^{\tilde{\sigma}^{y,r}}(\mathbf{x})\right)$	0.8320
UQ model for Y_0	$acc\left(\ell^{\tilde{\epsilon}^{y,r}}(\mathbf{x}), \ell^{\hat{\sigma}^{y,\hat{\omega}_1^M,r}}(\mathbf{x})\right)$	0.7891
Ensemble for Z_0^1	$acc\left(\ell^{\tilde{\epsilon}^{z_1,r}}(\mathbf{x}), \ell^{\tilde{\sigma}^{z_1,r}}(\mathbf{x})\right)$	1.0000
UQ model for Z_0^1	$acc\left(\ell^{\tilde{\epsilon}^{z_1,r}}(\mathbf{x}), \ell^{\hat{\sigma}^{z_1,\hat{\psi}_1^M,r}}(\mathbf{x})\right)$	1.0000

Table 12: Accuracy score between the binary labels of the relative RMSE, ensemble STD, and estimated STD values from $\mathcal{D}$ and $\tilde{\mathcal{D}}$ using the testing sample in Example 2.

Moreover, as the RMSE increases due to propagated errors with increasing N (from a certain value of N depending on the parameter set values), it is of interest to determine whether the UQ model can identify the value of N at which the algorithm attains the smallest RMSE based on the estimated STD. To investigate this, we generate a dataset $\mathcal{D}^{\mathbf{N}}$ similar to $\mathcal{D}$ in Table 6 with a fixed maturity $T = 0.3$, and each sampled parameter set is solved for $\mathbf{N} = \{2, 8, 32, 128\}$. The dataset $\mathcal{D}^{\mathbf{N}}$ consists of 2560 parameter sets, resulting in a total number of samples $M = 10240$. We choose $M^{train} = 8192$ and $M^{valid} = M^{test} = 1024$. Note that $\mathfrak{n} = 2$ since $\mathbf{x}_i = (b_i, N)$, $N \in \mathbf{N}$. We use the same hyperparameters for the UQ model as for dataset $\mathcal{D}$ and train only one model. To evaluate the accuracy score, we define the binary multi-label

$$\boldsymbol{\ell}^{\tilde{\epsilon}^r}(\mathbf{x}_i) = \begin{cases} \{1,0,0,0\} & \text{if } N^{min,\tilde{\epsilon}^r}(\mathbf{x}_i) = 2, \\ \{0,1,0,0\} & \text{if } N^{min,\tilde{\epsilon}^r}(\mathbf{x}_i) = 8, \\ \{0,0,1,0\} & \text{if } N^{min,\tilde{\epsilon}^r}(\mathbf{x}_i) = 32, \\ \{0,0,0,1\} & \text{if } N^{min,\tilde{\epsilon}^r}(\mathbf{x}_i) = 128, \end{cases}$$

for the relative RMSE, where $N^{min,\tilde{\epsilon}^r}(\mathbf{x}_i) = \arg\min_{N \in \mathbf{N}} \tilde{\epsilon}^r(b_i, N)$. The binary multi-labels for the relative ensemble STD $\boldsymbol{\ell}^{\tilde{\sigma}^r}(\mathbf{x}_i)$ and estimated STD $\boldsymbol{\ell}^{\hat{\sigma}^r}(\mathbf{x}_i)$ are defined similarly. The accuracy score values between these multi-labels for the testing sample of $\mathcal{D}^{\mathbf{N}}$ are presented in Table 13. We observe that the accuracy score values between the multi-labels of the relative RMSE and

UQ approach	Accuracy score	Dataset $\mathcal{D}^{\mathbf{N}}$
Ensemble for Y_0	$acc\left(\boldsymbol{\ell}^{\tilde{\epsilon}^{y,r}}(\mathbf{x}), \boldsymbol{\ell}^{\tilde{\sigma}^{y,r}}(\mathbf{x})\right)$	0.6680
UQ model for Y_0	$acc\left(\boldsymbol{\ell}^{\tilde{\epsilon}^{y,r}}(\mathbf{x}), \boldsymbol{\ell}^{\hat{\sigma}^{y,\hat{\omega}_1^M,r}}(\mathbf{x})\right)$	0.4805
Ensemble for Z_0^1	$acc\left(\boldsymbol{\ell}^{\tilde{\epsilon}^{z_1,r}}(\mathbf{x}), \boldsymbol{\ell}^{\tilde{\sigma}^{z_1,r}}(\mathbf{x})\right)$	0.7773
UQ model for Z_0^1	$acc\left(\boldsymbol{\ell}^{\tilde{\epsilon}^{z_1,r}}(\mathbf{x}), \boldsymbol{\ell}^{\hat{\sigma}^{z_1,\hat{\psi}_1^M,r}}(\mathbf{x})\right)$	0.5469

Table 13: Accuracy score between the multi-labels of the relative RMSE, ensemble STD, and estimated STD values from $\mathcal{D}^{\mathbf{N}}$ using the testing sample in Example 2.

estimated STD values $acc\left(\boldsymbol{\ell}^{\tilde{\epsilon}^r}(\mathbf{x}), \boldsymbol{\ell}^{\hat{\sigma}^r}(\mathbf{x})\right)$ are approximately 0.5. This indicates that the relative estimated STD correctly predicted the value of N with the smallest relative RMSE for around 50% of the parameter sets in the testing sample.
The accuracy score serves as a restrictive metric, requiring each predicted label ($\boldsymbol{\ell}^{\tilde{\sigma}^r}(\mathbf{x})$ or $\boldsymbol{\ell}^{\hat{\sigma}^r}(\mathbf{x})$) to exactly match the true label ($\boldsymbol{\ell}^{\tilde{\epsilon}^r}(\mathbf{x})$). Hence, it doesn't tolerate partial errors. For instance, if the N value with the smallest relative RMSE coincides with the one having the second smallest relative estimated STD, the prediction is counted as incorrect. This rigid evaluation fails to consider the order of predicted labels. For this purpose, we consider the mean reciprocal rank

(MRR) metric, since there is only one relevant label per sample. It measures the effectiveness of a model in ranking a list of predicted labels based on their relevance to the only true label. In our case, the true label is $N^{min,\tilde{\epsilon}^r}(\mathbf{x})$. The predicted ones are denoted by $\mathbf{N}^{sort,\tilde{\sigma}^r}(\mathbf{x})$ and $\mathbf{N}^{sort,\hat{\sigma}^r}(\mathbf{x})$, the ascending sorted $\mathbf{N}$ values for the parameter set $\mathbf{x}$ based on the value of relative ensemble STD and estimated STD, respectively. Hence, $MRR\left(N^{min,\tilde{\epsilon}^r}(\mathbf{x}), \mathbf{N}^{sort,\hat{\sigma}^r}(\mathbf{x})\right)$ is given by

$$MRR\left(N^{min,\tilde{\epsilon}^r}(\mathbf{x}), \mathbf{N}^{sort,\hat{\sigma}^r}(\mathbf{x})\right) = \frac{1}{256}\sum_{i=1}^{256}\frac{1}{pos\left(N^{min,\tilde{\epsilon}^r}(\mathbf{x}_i), \mathbf{N}^{sort,\hat{\sigma}^r}(\mathbf{x}_i)\right)},$$

where $pos\left(N^{min,\tilde{\epsilon}^r}(\mathbf{x}_i), \mathbf{N}^{sort,\hat{\sigma}^r}(\mathbf{x}_i)\right)$ gives the position where the true label $N^{min,\tilde{\epsilon}^r}(\mathbf{x}_i)$ is found in the list of predicted labels $\mathbf{N}^{sort,\hat{\sigma}^r}(\mathbf{x}_i)$. The mean reciprocal rank values are reported in Table 14. We observe that, on average, our UQ model can show that the smallest relative RMSE

UQ approach	Mean reciprocal rank	Dataset $\mathcal{D}^{\mathbf{N}}$
Ensemble for Y_0	$MRR\left(N^{min,\tilde{\epsilon}^{y,r}}(\mathbf{x}), \mathbf{N}^{sort,\tilde{\sigma}^{y,r}}(\mathbf{x})\right)$	0.8119
UQ model for Y_0	$MRR\left(N^{min,\tilde{\epsilon}^{y,r}}(\mathbf{x}), \mathbf{N}^{sort,\hat{\sigma}^{y,\hat{\omega}_1^M,r}}(\mathbf{x})\right)$	0.6849
Ensemble for Z_0^1	$MRR\left(N^{min,\tilde{\epsilon}^{z_1,r}}(\mathbf{x}), \mathbf{N}^{sort,\tilde{\sigma}^{z_1,r}}(\mathbf{x})\right)$	0.8812
UQ model for Z_0^1	$MRR\left(N^{min,\tilde{\epsilon}^{z_1,r}}(\mathbf{x}), \mathbf{N}^{sort,\hat{\sigma}^{z_1,\hat{\psi}_1^M,r}}(\mathbf{x})\right)$	0.7614

Table 14: Mean reciprocal rank between the N value with the smallest relative RMSE and the ascending sorted $\mathbf{N}$ values based on the relative ensemble STD and estimated STD values from $\mathcal{D}^{\mathbf{N}}$ using the testing sample in Example 2.

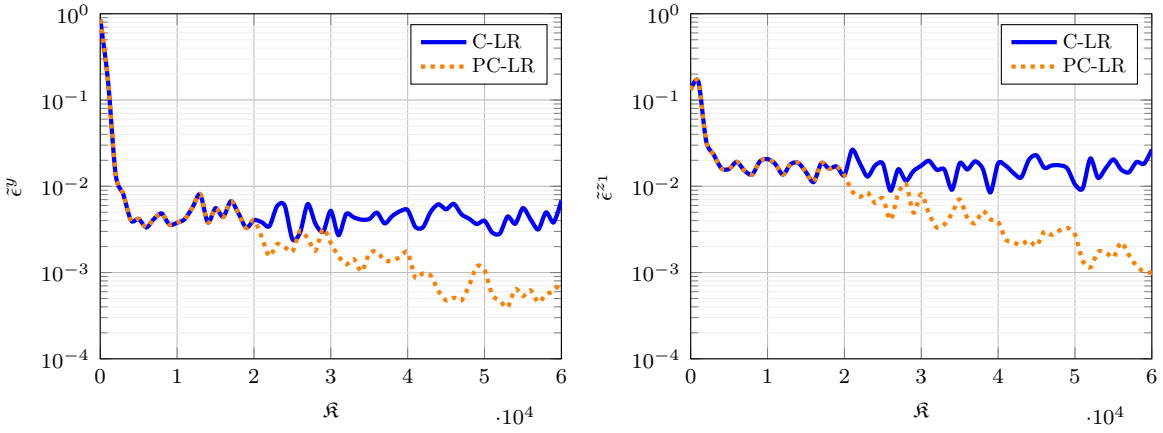
is achieved for the N value of either the first or second smallest relative estimated STD.

5 Conclusions

In this work, we investigate the sources of uncertainty in the deep learning-based BSDE schemes and develop a UQ model based on heteroscedastic nonlinear regression to estimate the uncertainty. We apply the UQ model to the pioneering scheme developed in [11] and the one in [31]. The STD of the approximate solution captures the uncertainty, which is usually estimated by performing multiple runs of the algorithm with different datasets. This approach is quite computationally expensive, especially in high-dimensional cases. Our UQ model estimates the STD much cheaper, namely using a single run of the algorithm. Under the assumption of normally distributed errors with zero mean and the STD depending on the parameter set of the discretized BSDE, we employ a DNN to learn two functions that estimate the mean and STD of the approximate solution. The DNN is trained using a dataset of i.i.d. samples, consisting of various parameter sets of the discretized BSDE and their corresponding approximated solutions from a single run of the algorithm. The network parameters are optimized by minimizing the negative log-likelihood. The STD is thus estimated much cheaper. Furthermore, the estimated mean can be leveraged to initialize the algorithm, improving the optimization process. Our numerical results demonstrate that the proposed UQ model provides reliable estimates of the mean and STD of the approximate solution for both considered schemes, even in high-dimensional cases. The estimated STD captures various sources of uncertainty, showcasing its capability in quantifying the uncertainty. Moreover, the UQ model illustrates the improved performance of the LaDBSDE scheme compared to the DBSDE scheme based on the corresponding estimated STD values. Finally, it can also identify the hyperparameters that yield a well-performing scheme.

Appendix A Impact of the sources of uncertainty for the Burgers type BSDE

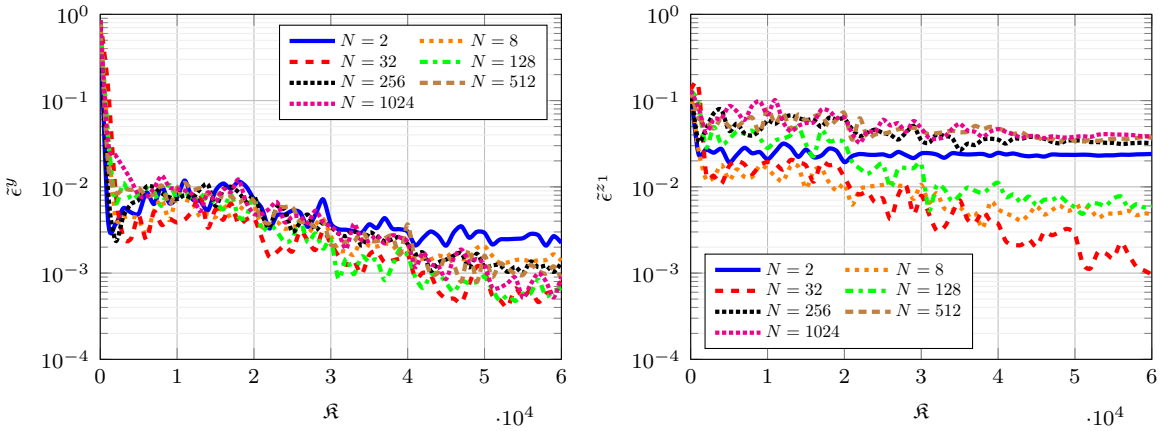
In this section, we visualize the effect of different errors on the RMSE for Example 2. The impact of the optimization error is shown in Figure 20 using C-LR and PC-LR approaches. For



(a) RMSE values for Y_0. (b) RMSE values for Z_0^1.

Figure 20: RMSE values are plotted for Example 2 using different learning rate approaches, where $T = 0.25$ and $b = 25$.

the discretization error, see Figure 21. The effect of the optimization error and propagated errors



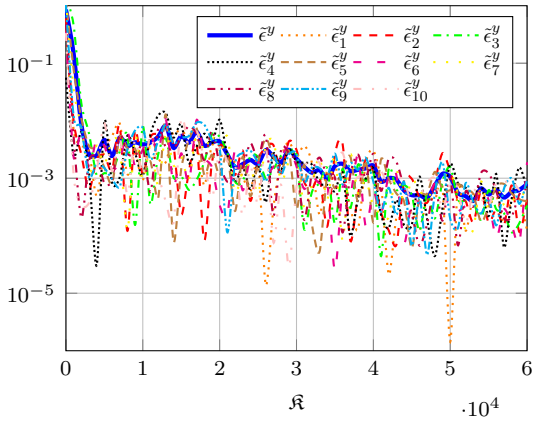
(a) RMSE values for Y_0. (b) RMSE values for Z_0^1.

Figure 21: RMSE values are plotted for Example 1 using $N \in \{2, 8, 32, 128, 256, 512, 1024\}$, where $T = 0.25$ and $b = 25$.

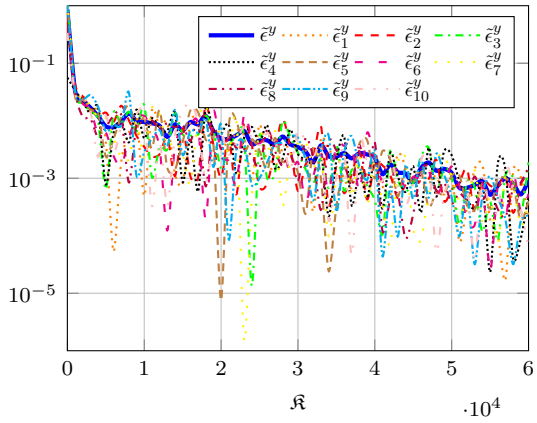
over time are displayed in Figure 22.

Appendix B Normality assumption of the error distribution

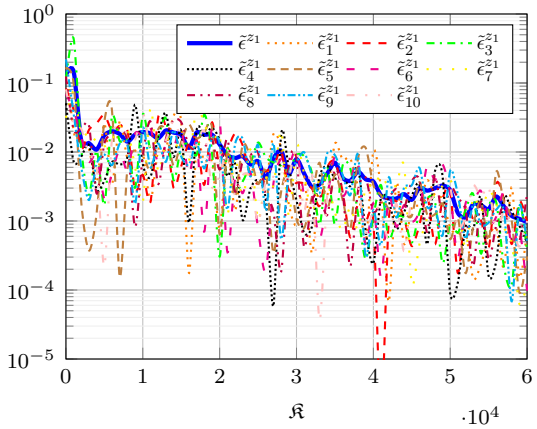
In this section, we conduct a test to assess the normality of the error distribution in (13) for Example 1. For the parameter values $T = 0.33, K = 100, S_0 = 100, a = 0.05, b = 0.2, R = 0.03$ and $\delta = 0$, the exact solution is $(Y_0, Z_0) = (5.0679, 11.1420)$. Using $N = 16$, $\mathfrak{K} = 30000$, $\alpha = 1\text{e}{-2}$ and conducting $Q = 500$ independent runs of the DBSDE algorithm, we display the



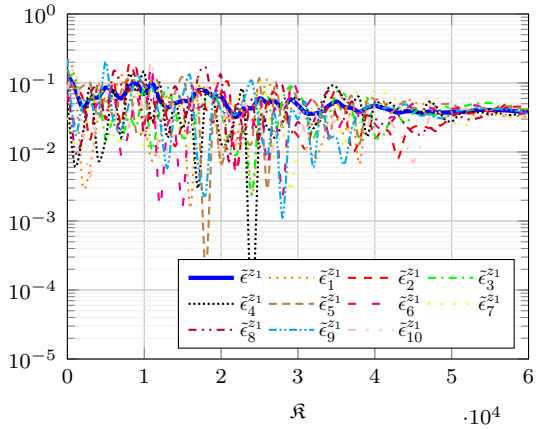
(a) RMSE values and the absolute error for each of $Q = 10$ DBSDE runs for Y_0 with $N = 32$.



(b) RMSE values and the absolute error for each of $Q = 10$ DBSDE runs for Y_0 with $N = 1024$.



(c) RMSE values and the absolute error for each of $Q = 10$ DBSDE runs for Z_0^1 with $N = 32$.



(d) RMSE values and the absolute error for each of $Q = 10$ DBSDE runs for Z_0^1 with $N = 1024$.

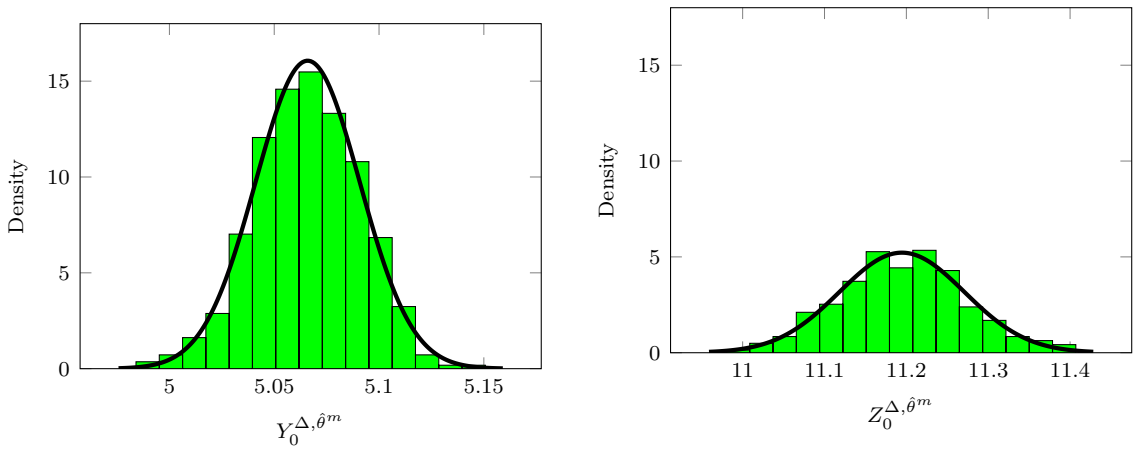
Figure 22: RMSE values and the absolute errors from each of $Q = 10$ DBSDE runs are plotted for Example 2 using $N \in \{32, 1024\}$, where $T = 0.25$ and $b = 25$.

empirical distribution of the approximations in Figure 23. The observed empirical distributions exhibit a Gaussian shape. To further assess the normality, we perform the Shapiro-Wilk [47] and D'Agostino and Pearson's [10] tests for the assessment of normality. The p-values obtained from these tests are presented in Table 15. With a significance level of 0.05, we conclude that there is

	Shapiro-Wilk	D'Agostino-Pearson
$Y_0^{\Delta,\hat{\theta}^m}$	0.5014	0.5368
$Z_0^{\Delta,\hat{\theta}^m}$	0.4391	0.6489

Table 15: p-value of the statistical tests in Example 1 for parameter set $T = 0.33, K = 100, S_0 = 100, a = 0.05, b = 0.2, R = 0.03$ and $\delta = 0$.

no evidence to reject the assumption of normal distribution for the approximate solutions $Y_0^{\Delta,\hat{\theta}^m}$ and $Z_0^{\Delta,\hat{\theta}^m}$.



(a) Empirical distribution of $Y_0^{\Delta,\hat{\theta}^m}$, where the calculated parameters are $(\tilde{\mu}^y, \tilde{\sigma}^y) = (5.0659, 0.0248)$.

(b) Empirical distribution of $Z_0^{\Delta,\hat{\theta}^m}$, where the calculated parameters are $(\tilde{\mu}^z, \tilde{\sigma}^z) = (11.1946, 0.0764)$.

Figure 23: Empirical distribution of the approximate solution (13) in Example 1 for parameter set $T = 0.33, K = 100, S_0 = 100, a = 0.05, b = 0.2, R = 0.03$ and $\delta = 0$. The curve represents a fitted normal distribution to the data.

References

[1] M. Abdar, F. Pourpanah, S. Hussain, D. Rezazadegan, L. Liu, M. Ghavamzadeh, P. Fieguth, X. Cao, A. Khosravi, U. R. Acharya, V. Makarenkov, and S. Nahavandi, *A review of uncertainty quantification in deep learning: Techniques, applications and challenges*, Inf. Fusion, 76 (2021), pp. 243–297, https://doi.org/10.1016/j.inffus.2021.05.008.

[2] S. L. Beal and L. B. Sheiner, *Heteroscedastic nonlinear regression*, Technometrics, 30 (1988), pp. 327–338, https://doi.org/10.1080/00401706.1988.10488406.

[3] C. Beck, S. Becker, P. Cheridito, A. Jentzen, and A. Neufeld, *Deep splitting method for parabolic pdes*, SIAM J. Sci. Comput., 43 (2021), pp. A3135–A3154, https://doi.org/10.1137/19M1297919.

[4] C. Beck, A. Jentzen, and B. Kuckuck, *Full error analysis for the training of deep neural networks*, Infin. Dimens. Anal. Quantum Probab. Relat. Top., 25 (2022), p. 2150020, https://doi.org/10.1142/S021902572150020X.

[5] C. Bender and J. Zhang, *Time discretization and markovian iteration for coupled FBSDEs*, Ann. Appl. Probab., 18 (2008), pp. 143–177, https://doi.org/10.1214/07-aap448.

[6] B. Bouchard and N. Touzi, *Discrete-time approximation and monte-carlo simulation of backward stochastic differential equations*, Stoch. Process Their Appl., 111 (2004), pp. 175–206, https://doi.org/10.1016/j.spa.2004.01.001.

[7] J.-F. Chassagneux, J. Chen, N. Frikha, and C. Zhou, *A learning scheme by sparse grids and picard approximations for semilinear parabolic pdes*, IMA J. Numer. Anal., (2022), p. drac066, https://doi.org/10.1093/imanum/drac066.

[8] D. Crisan and K. Manolarakis, *Solving backward stochastic differential equations using the cubature method: application to nonlinear pricing*, SIAM J. Financial Math., 3 (2012), pp. 534–571, https://doi.org/10.1137/090765766.

[9] G. Cybenko, *Approximation by superpositions of a sigmoidal function*, Math. Control Signal Systems, 2 (1989), pp. 303–314, https://doi.org/10.1007/BF02551274.

[10] R. D'Agostino and E. S. Pearson, *Tests for departure from normality. empirical results for the distributions of b^2 and $\sqrt{b^1}$*, Biometrika, 60 (1973), pp. 613–622, https://doi.org/10.1093/biomet/60.3.613.

[11] W. E, J. Han, and A. Jentzen, *Deep learning-based numerical methods for high-dimensional parabolic partial differential equations and backward stochastic differential equations*, Commun. Math. Stat., 5 (2017), pp. 349–380, https://doi.org/10.1007/s40304-017-0117-6.

[12] Y. Fu, W. Zhao, and T. Zhou, *Efficient spectral sparse grid approximations for solving multi-dimensional forward backward SDEs*, Discrete Contin. Dyn. Syst. - B, 22 (2017), pp. 3439–3458, https://doi.org/10.3934/dcdsb.2017174.

[13] M. Fujii, A. Takahashi, and M. Takahashi, *Asymptotic expansion as prior knowledge in deep learning method for high dimensional bsdes*, Asia-Pac. Financ. Mark., 26 (2019), pp. 391–408, https://doi.org/10.1007/s10690-019-09271-7.

[14] Y. Gal and Z. Ghahramani, *Dropout as a bayesian approximation: Representing model uncertainty in deep learning*, in Proc. 33-rd Int. Conf. Mach. Learn., vol. 48, PMLR, 2016, pp. 1050–1059, https://proceedings.mlr.press/v48/gal16.html.

[15] M. Germain, H. Pham, and X. Warin, *Approximation error analysis of some deep backward schemes for nonlinear pdes*, SIAM J. Sci. Comput., 44 (2022), pp. A28–A56, https://doi.org/10.1137/20M1355355.

[16] X. Glorot and Y. Bengio, *Understanding the difficulty of training deep feedforward neural networks*, in Proc. AISTATS 13-th Int. Conf. Artif. Intell. Stat., vol. 9, PMLR, 2010, pp. 249–256, https://proceedings.mlr.press/v9/glorot10a.html.

[17] E. Gobet and C. Labart, *Solving BSDE with adaptive control variate*, SIAM J. Numer. Anal., 48 (2010), pp. 257–277, https://doi.org/10.1137/090755060.

[18] E. Gobet, J.-P. Lemor, and X. Warin, *A regression-based monte carlo method to solve backward stochastic differential equations*, Ann. Appl. Probab., 15 (2005), pp. 2172–2202, https://doi.org/10.1214/105051605000000412.

[19] E. Gobet, J. G. López-Salas, P. Turkedjiev, and C. Vázquez, *Stratified regression monte-carlo scheme for semilinear PDEs and BSDEs with large scale parallelization on GPUs*, SIAM J. Sci. Comput., 38 (2016), pp. C652–C677, https://doi.org/10.1137/16m106371x.

[20] J. Han, A. Jentzen, and W. E, *Solving high-dimensional partial differential equations using deep learning*, Proc. Natl. Acad. Sci. U.S.A., 115 (2018), pp. 8505–8510, https://doi.org/10.1073/pnas.1718942115.

[21] J. Han and J. Long, *Convergence of the deep bsde method for coupled fbsdes*, Probab. Uncertain. Quant. Risk, 5 (2020), https://doi.org/10.1186/s41546-020-00047-w.

[22] D. Hendrycks and K. Gimpel, *A baseline for detecting misclassified and out-of-distribution examples in neural networks*, 2018, https://arxiv.org/abs/1610.02136.

[23] K. Hornik, M. Stinchcombe, and H. White, *Multilayer feedforward networks are universal approximators*, Neural Netw., 2 (1989), pp. 359–366, https://doi.org/10.1016/0893-6080(89)90020-8.

[24] E. Hüllermeier and W. Waegeman, *Aleatoric and epistemic uncertainty in machine learning: An introduction to concepts and methods*, Mach. Learn., 110 (2021), pp. 457–506, https://doi.org/10.1007/s10994-021-05946-3.

[25] C. Huré, H. Pham, and X. Warin, *Deep backward schemes for high-dimensional nonlinear PDEs*, Math. Comp., 89 (2020), pp. 1547–1579, https://doi.org/10.1090/mcom/3514.

[26] A. Jentzen and A. Riekert, *Strong overall error analysis for the training of artificial neural networks via random initializations*, Commun. Math. Stat., (2023), https://doi.org/10.1007/s40304-022-00292-9.

[27] H. Jiang, B. Kim, M. Guan, and M. Gupta, *To trust or not to trust a classifier*, Adv. Neural Inf. Process. Syst., 31 (2018).

[28] Y. Jiang and J. Li, *Convergence of the deep bsde method for fbsdes with non-lipschitz coefficients*, Probab. Uncertain. Quant. Risk, 6 (2021), pp. 391–408, https://doi.org/10.3934/puqr.2021019.

[29] L. Kapllani, *The effect of the number of neural networks on deep learning schemes for solving high dimensional nonlinear backward stochastic differential equations*, in M. Ehrhardt, M. Günther (eds) Progress in Industrial Mathematics at ECMI 2021. ECMI 2021. Mathematics in Industry(), vol. 39, Springer, Cham, 2022, https://doi.org/10.1007/978-3-031-11818-0_10.

[30] L. Kapllani and L. Teng, *Multistep schemes for solving backward stochastic differential equations on gpu*, J. Math. Ind., 12 (2022), https://doi.org/10.1186/s13362-021-00118-3.

[31] L. Kapllani and L. Teng, *Deep learning algorithms for solving high-dimensional nonlinear backward stochastic differential equations*, Discrete Contin. Dyn. Syst. - B, (2023), https://doi.org/10.3934/dcdsb.2023151.

[32] N. E. Karoui, S. Peng, and M. C. Quenez, *Backward stochastic differential equations in finance*, Math. Financ., 7 (1997), pp. 1–71, https://doi.org/10.1111/1467-9965.00022.

[33] A. Kendall and Y. Gal, *What uncertainties do we need in bayesian deep learning for computer vision?*, Adv. Neural Inf. Process. Syst., 30 (2017).

[34] D. P. Kingma and J. Ba, *Adam: A method for stochastic optimization*, 2017, https://arxiv.org/abs/1412.6980.

[35] M. A. Kupinski, J. W. Hoppin, E. Clarkson, and H. H. Barrett, *Ideal-observer computation in medical imaging with use of markov-chain monte carlo techniques*, J. Opt. Soc. Am. A, 20 (2003), pp. 430–438, https://doi.org/10.1364/JOSAA.20.000430.

[36] B. Lakshminarayanan, A. Pritzel, and C. Blundell, *Simple and scalable predictive uncertainty estimation using deep ensembles*, Adv. Neural Inf. Process. Syst., 30 (2017).

[37] J.-P. Lemor, E. Gobet, and X. Warin, *Rate of convergence of an empirical regression method for solving generalized backward stochastic differential equations*, Bernoulli, 12 (2006), pp. 889–916, https://doi.org/10.3150/bj/1161614951.

[38] J. Ma, J. Shen, and Y. Zhao, *On numerical approximations of forward-backward stochastic differential equations*, SIAM J. Numer. Anal., 46 (2008), pp. 2636–2661, https://doi.org/10.1137/06067393x.

[39] A. Mobiny, P. Yuan, S. K. Moulik, N. Garg, C. C. Wu, and H. V. Nguyen, *Dropconnect is effective in modeling uncertainty of bayesian deep networks*, Sci. Rep., 11 (2021), p. 5458, https://doi.org/10.1038/s41598-021-84854-x.

[40] D. A. Nix and A. S. Weigend, *Estimating the mean and variance of the target probability distribution*, in Proceedings of 1994 IEEE International Conference on Neural Networks (ICNN'94), vol. 1, IEEE, 1994, pp. 55–60, https://doi.org/10.1109/ICNN.1994.374138.

[41] P. Oberdiek, G. Fink, and M. Rottmann, *Uqgan: A unified model for uncertainty quantification of deep classifiers trained via conditional gans*, Adv. Neural Inf. Process. Syst., 35 (2022), pp. 21371–21385.

[42] P. Oberdiek, M. Rottmann, and H. Gottschalk, *Classification uncertainty of deep neural networks based on gradient information*, in L. Pancioni, F. Schwenker, E. Trentin (eds) Artificial Neural Networks in Pattern Recognition. ANNPR 2018. Lecture Notes in Computer Science(), vol. 11081, Springer, Cham, 2018, https://doi.org/10.1007/978-3-319-99978-4_9.

[43] Y. Ovadia, E. Fertig, J. Ren, Z. Nado, D. Sculley, S. Nowozin, J. Dillon, B. Lakshminarayanan, and J. Snoek, *Can you trust your model's uncertainty? evaluating predictive uncertainty under dataset shift*, Adv. Neural Inf. Process. Syst., 32 (2019).

[44] M. Raissi, *Forward-backward stochastic neural networks: Deep learning of high-dimensional partial differential equations*, 2018, https://arxiv.org/abs/1804.07010.

[45] M. Ruijter and C. Oosterlee, *Numerical fourier method and second-order taylor scheme for backward SDEs in finance*, Appl. Numer. Math., 103 (2016), pp. 1–26, https://doi.org/10.1016/j.apnum.2015.12.003.

[46] M. J. Ruijter and C. W. Oosterlee, *A fourier cosine method for an efficient computation of solutions to BSDEs*, SIAM J. Sci. Comput., 37 (2015), pp. A859–A889, https://doi.org/10.1137/130913183.

[47] S. S. Shapiro and M. B. Wilk, *An analysis of variance test for normality (complete samples)*, Biometrika, 52 (1965), pp. 591–611, https://doi.org/10.2307/2333709.

[48] A. Takahashi, Y. Tsuchida, and T. Yamada, *A new efficient approximation scheme for solving high-dimensional semilinear pdes: control variate method for deep bsde solver*, J. Comput. Phys., 454 (2022), p. 110956, https://doi.org/10.1016/j.jcp.2022.110956.

[49] Z. Wang and S. Tang, *Gradient convergence of deep learning-based numerical methods for bsdes*, Chin. Ann. Math., B, 42 (2021), pp. 199–216, https://doi.org/10.1007/s11401-021-0253-x.

[50] Y. Wen, P. Vicol, J. Ba, D. Tran, and R. Grosse, *Flipout: Efficient pseudo-independent weight perturbations on mini-batches*, 2018, https://arxiv.org/abs/1803.04386.

[51] G. Zhang, *A sparse-grid method for multi-dimensional backward stochastic differential equations*, J. Comput. Math., 31 (2013), pp. 221–248, https://doi.org/10.4208/jcm.1212-m4014.

[52] J. Zhang, *A numerical scheme for BSDEs*, Ann. Appl. Probab., 14 (2004), pp. 459–488, https://doi.org/10.1214/aoap/1075828058.

[53] W. Zhao, L. Chen, and S. Peng, *A new kind of accurate numerical method for backward stochastic differential equations*, SIAM J. Sci. Comput., 28 (2006), pp. 1563–1581, https://doi.org/10.1137/05063341x.

[54] W. Zhao, Y. Fu, and T. Zhou, *New kinds of high-order multistep schemes for coupled forward backward stochastic differential equations*, SIAM J. Sci. Comput., 36 (2014), pp. A1731–A1751, https://doi.org/10.1137/130941274.

[55] W. Zhao, G. Zhang, and L. Ju, *A stable multistep scheme for solving backward stochastic differential equations*, SIAM J. Numer. Anal., 48 (2010), pp. 1369–1394, https://doi.org/10.1137/09076979x.