Delft University of Technology

Faculty of Electrical Engineering, Mathematics and Computer Science

A Novel Method for Solving High-Dimensional Backward Stochastic Differential Equations Using Malliavin Calculus and Deep Learning

Bálint Négyesi

Supervisors:

Prof. dr. ir. Cornelis W. Oosterlee & Kristoffer Andersson

Masters of Science in Applied Mathematics

September, 2020

A Novel Method for Solving High-Dimensional Backward Stochastic Differential Equations Using Malliavin Calculus and Deep Learning

by

Bálint Négyesi

to obtain the degree of Master of Science
in Applied Mathematics at the Delft University of Technology,
to be defended on Thursday September 24, 2020 at 11:00 AM.

Student number:	4932668
Future correspondence:	balint.negyesi@gmail.com

Supervisors:
Prof. Dr. Ir. Cornelis W. Oosterlee
Kristoffer Andersson

Thesis committee:	
Prof. Dr. Ir. Cornelis W. Oosterlee	Numerical Analysis, TU Delft
Kristoffer Andersson	Scientific Computing, CWI
Dr. Cor Kraaikamp	Applied Probability, TU Delft
Dr. Ir. G. N. Joris C. Bierkens	Statistics, TU Delft

September, 2020, Delft.

Abstract

Backward stochastic differential equations (BSDE) are known to be a powerful tool in mathematical modeling due to their inherent connection with second-order parabolic partial differential equations (PDE) established by the non-linear Feynman–Kac relations. The fundamental power of BSDEs lies in the fact that with them one does not merely obtain the solution of the corresponding PDE but also its spatial gradient through the control process Z. Classical numerical methods tackling the system face the so-called curse of dimensionality and cannot be used to solve high-dimensional problems. In recent years, multiple approaches have been developed to overcome this computational burden, building on deep learning and showing remarkable empirical success even beyond 10 dimensions. However, such Deep BSDE methods struggle with giving accurate approximations for the Z-process throughout the whole time horizon. In this thesis we propose a novel approach aimed to give better estimations for the control problem, exploiting the natural dynamics of the Z-process given by Malliavin calculus. The proposed methods use deep learning parametrizations taking advantage of the universal approximation capability of neural networks. The Malliavin derivatives are estimated through the Malliavin chain rule. Two discrete numerical methods are developed which are called One-Step Malliavin (OSM) and Multi-Step Malliavin (MSM) schemes respectively. An error analysis is carried out proving the consistency of the algorithms and showing first-order convergence under certain assumptions. Numerical experiments are presented to demonstrate the efficiency of the Malliavin formulation compared to other Deep BSDE solvers.

Keywords: backward stochastic differential equations, Deep BSDE, Malliavin calculus, deep learning, neural networks, PDE, high-dimensions, curse of dimensionality.

Contents

List of Algorithms

List of Tables

List of Figures

Az ég beborul, egészen besötétedik. Felvonyítanak a kutyák, meglapul a nyúl, fut a szarvascsorda, fut, riadtan menekül. És ebbe' a félelmetes és felfoghatalan alkonyatban még a madarak is, a madarak is megzavarodnak és a fészkükre szállnak riadtan. És akkor néma csend. Minden élőlény elnémul. Vajon megindulnak a hegyek? Ránk dől a mennyboltozat? Beomlik a Föld? Nem tudjuk mi lesz, nem tudjuk mi lesz, mert beállt a teljes napfogyatkozás. De... de nincs ok a félelemre. Még nincs vége. Mert a Nap izzó gömbjének túloldalán lassan kiúszik a Hold. És a Nap most újból felragyog. És a Földre lassan visszatér a fény és áradni kezd újból a meleg. És mindenkit megindultság fog el, hogy felszabadul a sötétség súlya alól.

Tarr Béla – Krasznahorkai László, Werckmeister harmóniák
(részlet)

The sky darkens, then goes all dark. The dogs howl, rabbits hunch down, the deer run in panic, run, stampede in fright. And in this awful, incomprehensible dusk, even the birds... the birds too are confused and go to roost. And then... Complete silence. Everything that lives is still. Are the hills going to march off? Will heaven fall upon us? Will the Earth open under us? We don't know. We don't know, for a total eclipse has come upon us... But... but no need to fear. It's not over. For across the sun's glowing sphere, slowly, the Moon swims away. And the sun once again bursts forth, and to the Earth slowly there comes again light, and warmth again floods the Earth. Deep emotion pierces everyone. They have escaped the weight of darkness.

Béla Tarr – László Krasznahorkai, Werckmeister Harmonies
(opening scene)

Acknowledgments

First and foremost, I would like to express my gratitude towards my two supervisors, Kees Oosterlee and Kristoffer Andersson for their time, empathy and patience in putting up with all my impossibly chaotic, highly non-linear and footnote-rich[1] explanations about my progress. Their remarks, careful inspections and suggestions surely made this work significantly better. Kees, I would like to thank you for – on top of being an excellent, understanding and attentive supervisor – all the great music recommendations you gave me during the last nine months. They made some parts of this work much more enjoyable. (Ed Kuepper is really cool!) Kristoffer, thanks for stopping by my office with that Deep BSDE paper last December and for having brought my attention to this intriguing field. Moreover, I would also like to thank Adam Andersson whose initial encouragement in my original Malliavin related idea might have been small but it surely was indispensable for persisting with the project.

This thesis concludes my traineeship at the Scientific Computing group in Centrum Wiskunde & Informatica. I would like to thank all my (ex-)colleagues for our fruitful discussions and most importantly for all the entertaining games of squash. This work also puts an end to my two years as an Applied Mathematics master student in Delft. Thanks to everyone who in any way contributed to the experience. It was fun.

[1] said so

Introduction

The concept of backward stochastic differential equations (BSDE) was first introduced by Bismut in [1] for a linear stochastic control problem. The field has been an area of intense research ever since because – as shown in [2] – it turns out that BSDEs provide a natural probabilistic formulation to a wide range of physical phenomena through their inherent connections with second-order parabolic partial differential equations (PDE). The intriguing peculiarity of BSDEs is that their solution is a pair of random processes with which they do not merely provide the solution to a corresponding PDE by their Y-process but also its spatial gradient at all points over spacetime in the so-called control, Z-process. Hence, these backward equations by construction yield sensitivities simultaneously to the solution, an attribute that is of fundamental relevance in several applications.

Driven by the possibility of probabilistic mesh-free methods they provide for PDEs, several numerical approaches have been developed to tackle the BSDE problem. Classical numerical methods of that kind include a discretization of the continuous time window and subsequently approximate recursive conditional expectations backwards in time. Such approaches have proven to be successful up to 6 and 7 dimensions – see, e.g., [3], [4], [5] or [6]. However, they come with one significant drawback. Namely, through the choice of a finite set of basis functions – such as polynomials or piecewise step functions – they face the so-called *curse of dimensionality*, meaning that their computational complexity scales exponentially in the number of input dimensions. Recently, a new class of BSDE solvers were proposed in the literature in order to overcome this computational obstacle. The group of so-called *Deep BSDE* methods are founded on the exceptional practical achievements deep learning has reached over the past decades in a wide range of engineering applications. The majority of such Deep BSDE approaches were inspired by the pioneering paper of Han et. al in [7], where the backward equation is reformulated in the form of a forward Euler–Maruyama discretization, using a sequence of neural networks parametrizing the Z-process in the BSDE. The authors of [7] then suggest to train the sequence of neural networks on a terminal loss produced by the forward discretization. Encouraged by the experimental successes shown by this algorithm, another approach was proposed in [8], [9] and [10] where both processes in the backward equation are parametrized by deep neural networks. Thereupon, similarly to classical conditional expectation approaches, the neural networks are optimized in a backward induction through a sequence of loss functions formulated on a forward Euler–Maruyama discretization of the backward equation. We call these two models the Forward and Backward Deep BSDE solvers respectively.

Both algorithms have shown remarkable empirical success in solving BSDEs beyond 10 dimensions for equations of highly complex structures. These results incite hope that such deep learning based formulations may indeed be the answer to the curse of dimensionality. However, Deep BSDE models also exhibit serious downsides. In fact, the Forward Deep BSDE method only manages to solve the equation at $t = 0$ and its accuracy severely deteriorates at further points in time, for both the Y- and Z-processes. In exchange for performing multiple sequential optimization steps, the Backward Deep BSDE method manages to mitigate the errors stemming in the Y-process at $t > 0$, nonetheless it fails to give estimates of the same accuracy for the Z-process throughout the whole time horizon. Consequently, both schemes struggle to cope with the Z part of the solution, which in fact is the biggest challenge in the numerical approximation of BSDEs. A possible explanation for this limitation is that both of the methods are solely formulated on the BSDE driving the evolution of the Y-process, whereas neither of them poses direct training on the control process. Hence, the estimations for the Z-process are only trained implicitly through the dynamics of Y.

In this thesis, we propose a novel method to counteract this phenomenon inspired by the logic of classical recursive conditional expectation schemes. We split up the approximation tasks of the Y- and Z-processes and estimate each by separate neural network Least-Squres Monte Carlo (LSMC) regressions. Additionally, using the natural dynamics of the control process provided by Malliavin calculus, we show that the Z-process itself satisfies a linear BSDE depending on the Malliavin derivatives of the solution pair of the BSDE. Building on this well-known theoretical result, we derive two discrete numerical schemes where the Z-process is approximated by explicit conditional expectations arising from

Malliavin calculus. The two proposed schemes are called the One-Step Malliavin (OSM) and Multi-Step Malliavin (MSM) schemes respectively and they only differ in their corresponding discretizations of the underlying time horizon. In both algorithms, the Malliavin derivatives of the solution pair of the BSDE are estimated through the Malliavin chain rule formula, enabled by neural networks being differentiable universal function approximators.

The thesis is structured as follows. In the first three chapters we provide a theoretical introduction to the key concepts of the main idea. Namely, in chapter 1 we introduce backward stochastic differential equations and prove their well-posedness under general assumptions on the underlying randomness. Thereafter, we cover forward backward stochastic differential equations (FBSDE) and state the generalized Feynman–Kac relations establishing the connection between BSDEs and PDEs. This is followed by an introduction on Malliavin calculus given in chapter 2, where we mostly focus on the Malliavin derivative operator. We emphasize the importance of the so-called Malliavin chain rule in the context of this work. Thereafter the properties of the Malliavin derivatives of an FBSDE system are discussed, where we show that the Malliavin derivatives of the solution pair of a BSDE satisfy a linear BSDE themselves. This theoretical result is the basis of the proposed algorithms since the solution of the Malliavin BSDE provides a continuous version of the Z-process in the original BSDE. Finally, to conclude the theoretical introduction, an overview on deep learning is presented in chapter 3. In this part, we introduce the concept of fully-connected feedforward deep neural networks. We discuss their well-known universal approximation capability given by Theorem 3.2.2 which states that the class of even shallow neural networks is dense over compact subsets of Sobolev spaces.

The theoretical introduction is followed by a discussion on numerical analysis. First, in chapter 4 classical numerical methods to solve FBSDE systems are covered. We derive recursive conditional expectation schemes for the backward equation and explain the concept of the Least-Squares Monte Carlo (LSMC) regression which is the numerical tool of this work to approximate conditional expectations. In the final part of the chapter, we describe the class of Deep BSDE methods in more details, elaborate on their major drawbacks and comment on why they cannot be used to give accurate control estimates throughout the whole spacetime. Thereupon, motivated by this observation, we propose the two novel discrete schemes OSM and MSM in chapter 5. Subsequently, an error analysis is carried out for both proposed algorithms in chapter 6. We derive error bounds which depend linearly on the mesh-size of the discretized time interval, on top of the regression errors induced by neural network estimations. Under certain assumptions, we show that one, by allowing for multistepping, gains an $\mathcal{O}(1/N)$ order of convergence in the approximation errors of the Z-process through the cumulative regression error of the Y-process. The resulting error figures are compared to ones established for standard Deep BSDE methods. Afterwards numerical experiments are presented in chapter 7, where the practical performance of the proposed schemes is demonstrated. We show that they indeed manage to overcome some of the drawbacks of the Deep BSDE methods, giving more accurate control estimates in case of numerical examples up to 10 dimensions. A particular example is highlighted where, for an equation of complex structure, the proposed schemes give an order of magnitude better convergence than the Backward Deep BSDE method. Ultimately, our findings are summarized in the Discussion where we conclude the results of the work and lay out possible directions for future research.

Chapter 1

Backward Stochastic Differential Equations

In the following chapter we introduce the key concept of this work, backward stochastic differential equations (BSDE). We first give an intuition on how BSDEs can be interpreted compared to standard forward stochastic differential equations (SDE), also covering how are they inherently different from them. After this heuristic motivation, we establish the well-posedness of a general non-linear BSDE problem and give a proof on the existence of a unique solution under certain regularity conditions. As in later stages the class of linear BSDEs is going to be crucial for our main algorithms – see Equation 2.19 in particular –, we give a representation formula for the solution of such equations. Finally, we introduce the concept of forward-backward stochastic differential equations (FBSDE), define the final problem setup for the rest of the analysis and put the work into context through the so-called generalized Feynman–Kac relations, with which we demonstrate that solving BSDEs is (in some probabilistic sense) equivalent to solving a rather wide class of second-order parabolic partial differential equations (PDE).

1.1 Preliminaries

Let us fix some finite $0 \leq T < \infty$. During the whole thesis we are concerned with a filtered probability space $(\Omega, \mathcal{F}, \mathbb{F}, \mathbb{P})$, where $\mathbb{F} := \{\mathcal{F}_t\}_{t\in[0,T]}$ and $\mathcal{F}_t := \sigma\{W_s : 0 \leq s \leq t \leq T\}$ is the natural filtration generated by a d-dimensional Brownian motion augmented with $\mathbb{P}$-null sets of Ω. We denote $\mathcal{F} = \mathcal{F}_T$. In what follows, (in)equalities between random $\mathbb{F}$-measurable quantities are always meant in the $\mathbb{P}$-a.s. sense.

Furthermore, throughout the whole work we heavily use the following notations, definitions of certain spaces.

Definition 1.1.1 (Notations, spaces)**.** Let us introduce the following notations

- $|X| := \sqrt{\mathrm{Tr}\left[X^T X\right]}$ for $X \in \mathbb{R}^{n\times d}$. We use the same notation for scalars and vectors as well, therefore it is worth to notice that for the latter two this norm reduces to the standard Euclidean norm;
- $\langle y|z\rangle : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ the inner product;
- $y \odot z, y \oslash z : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}^d$ the element-wise multiplication and division respectively;
- $\mathbb{L}^p_{\mathcal{G}}(\mathbb{R}^n)$: for any $\mathcal{G} \subseteq \mathcal{F}_T$ the space of all $\mathcal{G}$-measurable random variables $\xi : \Omega \to \mathbb{R}^n$ such that $\mathbb{E}\left[|\xi|^p\right] < \infty$;
- $\mathbb{L}^p_{\mathbb{F}}(\mathbb{R}^n)$: the space of all progressively $\mathcal{F}$-measurable random processes $\Phi : [0,T] \times \Omega \to \mathbb{R}^d$, such that for each $t \in [0,T]$: $\mathbb{E}\left[|\Phi_t|^p\right] < \infty$;
- $\mathbb{L}^\infty_{\mathbb{F}}(\mathbb{R}^n)$: the space of all progressively $\mathcal{F}$-measurable bounded random processes Φ, i.e. $\Phi : \Omega \times [0,T] \to \mathbb{R}^n$ for which $\exists C \in \mathbb{R}$ such that $\mathbb{P}\left[\sup_{t\in[0,T]} |\Phi_t| > C\right] = 0$;
- $\mathbb{H}^p_{\mathbb{F}}(\mathbb{R}^{n\times d})$: the space of all progressively $\mathbb{F}$-measurable random processes $\Phi : [0,T] \times \Omega \to \mathbb{R}^{n\times d}$, such that $\mathbb{E}\left[\left(\int_0^T |\Phi_t|^2 \mathrm{d}t\right)^{p/2}\right] < \infty$; with slight abuse of terminology we often call this latter norm the *integral norm*;

- $\mathbb{S}^p_{\mathbb{F}}(\mathbb{R}^d)$: the subspace in $\mathbb{H}^p_{\mathbb{F}}(\mathbb{R}^{d\times 1})$ of all continuous processes;
- $\widehat{\mathbb{S}}^p_{\mathbb{F}}(\mathbb{R}^d)$: the space of all progressively $\mathbb{F}$-measurable, continuous random processes $\Phi : [0,T]\times\Omega \to \mathbb{R}^d$, such that $\mathbb{E}\left[\sup_{t\in[0,T]}|\Phi_t|^p\right] < \infty$; with slight abuse of terminology we often call this latter norm the *supremum norm*;
- $\|\cdot\|_\beta$: the norm, which for any $\beta > 0$ and $\Phi \in \mathbb{H}^2_{\mathbb{F}}(\mathbb{R}^{n\times d})$ is defined by $\|\Phi\|^2_\beta := \mathbb{E}\left[\int_0^T e^{\beta t}|\Phi_t|^2 \mathrm{d}t\right]$;
- $\mathbb{H}^2_{\mathbb{F},\beta}(\mathbb{R}^{n\times d})$: the metric space $(\mathbb{H}^2_{\mathbb{F}}(\mathbb{R}^{n\times d}), \|\cdot\|_\beta)$ equipped with this norm.

With slight abuse of the notation, we often use $\mathbb{H}^p_{\mathbb{F}}(\mathbb{R}^d) := \mathbb{H}^p_{\mathbb{F}}(\mathbb{R}^{1\times d})$ when $n = 1$.

Moreover, in the rest of the chapter we rely on the following widely known theorems of functional analysis and stochastic calculus which are stated without proofs.

Theorem 1.1.1 (Banach's Fixed-Point Theorem)
Let (X, d) be a non-empty complete metric space and $\Phi : X \to X$ a contraction mapping defined on it, i.e. there exists $K \in [0, 1)$ such that $\forall x, y \in X$

$$d(\Phi(x), \Phi(y)) \le K d(x, y). \tag{1.1}$$

Then Φ admits a unique fixed-point x^* such that $\Phi(x^*) = x^*$.

Proof. See, e.g., [11]. □

Theorem 1.1.2 (Burkholder–Davis–Gundy Inequality)
Let $\sigma \in \mathbb{H}^2_{\mathbb{F}}(\mathbb{R}^d)$. Then, for any $p > 0$ there exist universal constants $k_p, K_p \in \mathbb{R}$ depending only on p and d, such that the following inequalities hold

$$k_p\mathbb{E}\left[\left(\int_0^T |\sigma_t|^2\mathrm{d}t\right)^{p/2}\right] \le \mathbb{E}\left[\left|\sup_{t\in[0,T]}\int_0^t \sigma_s \mathrm{d}W_s\right|^p\right] \le K_p\mathbb{E}\left[\left(\int_0^T |\sigma_t|^2\mathrm{d}t\right)^{p/2}\right]. \tag{1.2}$$

Proof. See, e.g., [12]. □

Theorem 1.1.3 (Martingale Representation Theorem)
Let $\xi \in \mathbb{L}^2_{\mathcal{F}_T}(\mathbb{R})$. Then there exists a unique $Z \in \mathbb{H}^2_{\mathbb{F}}(\mathbb{R}^d)$ such that

$$\xi = \mathbb{E}[\xi] + \int_0^T Z_s \mathrm{d}W_s. \tag{1.3}$$

Consequently, for any square-integrable $\mathbb{F}$-martingale M, there exists a unique $Z \in \mathbb{H}^2_{\mathbb{F}}(\mathbb{R}^d)$ such that

$$M_t = M_0 + \int_0^t Z_s \mathrm{d}W_s. \tag{1.4}$$

Proof. See, e.g., [13]. □

1.2 (Forward) Stochastic Differential Equations

A forward stochastic differential equation (SDE) is an equation given by

$$X_t = \eta + \int_0^t \mu(t, X_t)\mathrm{d}t + \int_0^t \sigma(t, X_t)\mathrm{d}W_t, \qquad 0 \le t \le T, \tag{1.5}$$

where X is a d-dimensional random process, $\mu : [0,T]\times\mathbb{R}^d \to \mathbb{R}^d$, $\sigma : [0,T]\times\mathbb{R}^d \to \mathbb{R}^{d\times d}$ are deterministic functions called the *drift* and *diffusion* coefficients respectively, and η is an $\mathcal{F}_0$-measurable random variable. We frequently make use of the above equation's differential form, which can be written in the following way

$$\mathrm{d}X_t = \mu(t, X_t)\mathrm{d}t + \sigma(t, X_t)\mathrm{d}W_t, \qquad X_0 = \eta, \qquad 0 \le t \le T. \tag{1.6}$$

It is important to highlight that due to the non-trivial nature of SDEs there exist multiple concepts for what we consider a solution to the equation above. For the purpose of this work, in the following we are only considering so-called *strong solutions*, meaning that we say a measurable process X is a solution to the SDE above, if Equation 1.5 is satisfied $\mathbb{P}$-a.s. For a further discussion on other kinds of solutions see, e.g., [13].

In order to have a basic idea on when does such a solution exist, let us establish the following assumption

Assumption 1.2.1 (Unique Solution of SDEs)
Let the initial condition η and the coefficients $\mu : [0,T]\times\mathbb{R}^d \to \mathbb{R}^d$, $\sigma : [0,T]\times\mathbb{R}^d \to \mathbb{R}^{d\times d}$ in Equation 1.5 be such that

1. $\eta \in \mathbb{L}^2_{\mathcal{F}_0}(\mathbb{R}^d)$;
2. *they admit linear growth in x, i.e. there exists a constant C such that $\forall (t,x) \in [0,T]\times\mathbb{R}^d$*
$$|\mu(t,x)| + |\sigma(t,x)| \leq C\left(1+|x|\right); \tag{1.7}$$
3. *μ, σ are uniformly Lipschitz continuous in the spatial variable, i.e. there exists a constant L such that $\forall x_1, x_2 \in \mathbb{R}^d, t \in [0,T]$*
$$|\mu(t,x_1) - \mu(t,x_2)| + |\sigma(t,x_1) - \sigma(t,x_2)| \leq L|x_1 - x_2|. \tag{1.8}$$

Theorem 1.2.1 (Existence of Unique Solution – SDEs)
Under Assumption 1.2.1, the SDE above admits to a unique continuous solution $X \in \widehat{\mathbb{S}}^2_{\mathbb{F}}(\mathbb{R}^d)$.

Proof. See, e.g., [13]. □

From now on, we often refer to the solution of SDEs satisfying the assumption above as *standard Itô-processes*. An important attribute of such processes is that they provide a probabilistic representation to the solution of a certain class of partial different differential equations (PDE) established by the well-known Feynman–Kac formula.

Theorem 1.2.2 (Feynman–Kac)
Consider the following, boundary valued linear PDE problem

$$\frac{\partial u}{\partial t}(t,x) + \langle \mu | \nabla u\rangle (t,x) + \frac{1}{2}\operatorname{Tr}\left[\sigma\sigma^T \operatorname{Hess} u\right](t,x) - \langle v|u\rangle(t,x) - f^0(t,x) = 0$$
$$u(T,x) = g(x).$$

Additionally, let X be the solution of a corresponding SDE given by

$$X_t = \eta + \int_0^t \mu(s,X_s)\mathrm{d}s + \int_0^t \sigma(s,X_s)\mathrm{d}W_s,$$

and let the conditions of Assumption 1.2.1 be satisfied. Then the forward SDE has a unique solution, furthermore, if the solution of the PDE problem exists, it can be written as the conditional expectation

$$u(t,x) = \mathbb{E}\left[\int_t^t e^{\int_t^s v(r,X_r)\mathrm{d}r} f^0(s,X_s)\mathrm{d}s + e^{\int_t^T v(r,X_r)\mathrm{d}r} g(X_T)\middle| X_t = x\right]. \tag{1.9}$$

Proof. The proof is a simple consequence of Itô's lemma, we refer the more interested reader to, e.g., [12]. □

1.2.1 Arithmetic and Geometric Brownian Motions

To conclude this section, let us finally mention two special cases when the solution of Equation 1.5 adheres to a closed-form expression of the underlying Brownian motion. These are the so-called Arithmetic (ABM) and Geometric Brownian Motions (GBM) whose dynamics read as

$$\mathrm{d}X_t^{\mathrm{ABM}} = \mu 1_d \mathrm{d}t + \sigma I_d \mathrm{d}W_t, \qquad X_0^{\mathrm{ABM}} = x_0, \qquad 0 \leq t \leq T, \tag{1.10}$$
$$\mathrm{d}X_t^{\mathrm{GBM}} = \mu X_t^{\mathrm{GBM}} \mathrm{d}t + \sigma \operatorname{diag}\left(X_t^{\mathrm{GBM}}\right)\mathrm{d}W_t, \qquad X_0^{\mathrm{GBM}} = x_0, \qquad 0 \leq t \leq T, \tag{1.11}$$

where $\mu, \sigma \in \mathbb{R}$, 1_d denotes a d-dimensional vector full of ones, I_d is the identity matrix and $\operatorname{diag} : \mathbb{R}^d \to \mathbb{R}^{d\times d}$ maps a d-dimensional vector to a diagonal matrix whose main diagonal is the vector itself. Notice that in above, by slight abuse of notation, we redefined the coefficients μ, σ in Equation 1.5 with scalars which – although may be confusing at first – is in accordance with the standard notation of the literature. From this point on, when – and only then – talking about ABM or GBM, μ and σ always correspond to the scalars in Equation 1.10 and Equation 1.11.

Applying Itô's lemma, it is straightforward to show that the solutions of such equations admit to

$$X_t^{\mathrm{ABM}} = x_0 + \mu t 1_d + \sigma W_t, \tag{1.12}$$
$$X_t^{\mathrm{GBM}} = x_0 \odot e^{\left(\mu - \frac{\sigma^2}{2}\right)1_d t + \sigma W_t}, \tag{1.13}$$

where the exponential function is also taken element-wise. These expressions are of great use since, as we shall see in chapter 6 and chapter 7, they allow one to use analytically accurate approximations for diffusion dynamics of the form Equation 1.10 and Equation 1.11 respectively.

1.3 General Formulation

The concept of backward stochastic differential equations (BSDE) was first introduced by Bismut [1] in 1973 for a linear stochastic control problem. Later Pardoux and Peng [2], [14] generalized the formulation and its solvability to non-linear settings and proved their connection with second-order quasi-linear parabolic partial differential equations. The field has been an area of intense research ever since then, as BSDEs turn out to be a natural formulation for a wide range of PDEs and stochastic control problems. In the following section we build up an intuition on what BSDEs mean and formulate the problem in a general setting.

As SDEs can be thought of as a non-linear extension to stochastic integration, BSDEs are an extension to SDEs built on the martingale representation theorem. By the martingale representation Theorem 1.1.3, we can easily see that any martingale induced by taking appropriate conditional expectations of an $\xi \in \mathbb{L}^2_{\mathcal{F}_T}(\mathbb{R})$ random variable: $Y_t := \mathbb{E}[\xi|\mathcal{F}_t]$, admits to

$$Y_t := \mathbb{E}[\xi|\mathcal{F}_t] = Y_0 + \int_0^t Z_s \mathrm{d}W_s. \tag{1.14}$$

By which we also have

$$Y_T = Y_0 + \int_0^T Z_s \mathrm{d}W_s. \tag{1.15}$$

Combining these latter two equations, we get

$$Y_t = \xi - \int_t^T Z_s \mathrm{d}W_s. \tag{1.16}$$

This is a linear SDE, where instead of fixing an initial condition in $t = 0$ we fix a random terminal condition at $t = T$.

It is important to notice that the solution of a BSDE is a pair of adapted random processes (Y, Z) which satisfy the equation above. Indeed, equations with the above dynamics in the absence of the process Z have one unique solution which reads as $Y_t = \xi$. However, since ξ is an $\mathcal{F}_T$-measurable random variable, allowing for such a solution would violate the natural requirements of adaptivity. Therefore – in rather laymen's terms – one can think of the process Z as the additional information in the solution, which *"subtracts the required amount of randomness"* [15, pg. 6] from the non-adapted terminal condition ξ.

Motivated by the example above, in the rest of this chapter we are concerned with non-linear BSDEs of the form

$$Y_t = \xi + \int_t^T f(s, Y_s, Z_s)\mathrm{d}s - \int_t^T Z_s \mathrm{d}W_s, \qquad 0 \le t \le T, \tag{1.17}$$

where ξ is an $\mathcal{F}_T$-measurable random variable called *terminal condition*, and $f : \Omega \times [0, T] \times \mathbb{R}^n \times \mathbb{R}^{n\times d} \to \mathbb{R}^n$ is an $\mathbb{F}$-measurable function called the *driver* or the *generator* of the BSDE. We often make use of this equation's differential form which reads

$$\mathrm{d}Y_t = -f(t, Y_t, Z_t)\mathrm{d}t + Z_t \mathrm{d}W_t, \qquad Y_T = \xi, \qquad 0 \le t \le T. \tag{1.18}$$

As argued above, adaptivity of processes satisfying the equations above is non-trivial, therefore it is worth to define the concept of a solution explicitly.

Definition 1.3.1 (Solution of the BSDE)**.** The solution of the BSDE is a pair of random processes (Y, Z) satisfying Equation 1.17, such that $Y \in \mathbb{S}^2_{\mathbb{F}}(\mathbb{R}^n)$ and $Z \in \mathbb{H}^2_{\mathbb{F}}(\mathbb{R}^{n\times d})$.

In order to establish the existence of a unique solution, we need to put certain restrictions on the terminal condition and the generator of the equation. These are collected in the following assumption.

Assumption 1.3.1 (Unique Solution of BSDEs)
Let the parameters in Equation 1.17 be such that

1. $\xi \in \mathbb{L}^2_{\mathcal{F}_T}(\mathbb{R}^n)$;
2. $f(t, y, z)$ *is uniformly Lipschitz continuous in* (y, z), *i.e. there exists* L *such that* $\forall(t, y_1, z_1)$, $(t, y_2, z_2) \in [0, T] \times \mathbb{R}^n \times \mathbb{R}^{n\times d}$

$$|f(\omega, t, y_1, z_1) - f(\omega, t, y_2, z_2)| \le L\left(|y_1 - y_2| - |z_1 - z_2|\right), \tag{1.19}$$

$\mathrm{d}t \times \mathrm{d}\mathbb{P}$-*a.s;*

3. $f(\cdot, 0, 0) \in \mathbb{H}^2_{\mathbb{F}}(\mathbb{R}^n)$.

We call the parameters (ξ, f) of Equation 1.17 *standard* when the assumption above is satisfied.

Finally, we remark that most of the results we present below would hold for a more general multidimensional setting, however, since this work is only concerned with scalar BSDEs for the ease of notation we assume $n = 1$. This means that – by slight abuse of notation – Z_t is always a $1 \times d$-dimensional row-vector.

1.4 Well-Posedness of BSDEs

In the following section we are considering the general non-linear BSDE setting and establish the well-posedness of the problem, thus we show the existence of a unique solution. The section is built up as follows. First we prove an a priori estimate for pairs of random processes satisfying equations like Equation 1.17. Using this estimate, we show the existence of a unique solution by defining a contraction mapping over the space $\mathbb{S}^2_{\mathbb{F}}(\mathbb{R}) \times \mathbb{H}^2_{\mathbb{F}}(\mathbb{R}^d)$ and applying Banach's fixed-point theorem. Consequently, we construct a way to estimate the unique solution via a sequence of Picard iterations. Finally, in the light of the latter Malliavin formulation in chapter 2, we study the class of linear BSDEs and prove a representation formula for the solution of such equations. Throughout the whole section, we are largely building on [16, Chapter 2] and [12, Chapter 5].

1.4.1 A Priori Estimate

In order to show the well-posedness of the problem, we first derive an error bound for the difference of the solutions of two BSDEs. It is worth to mention that in literature there exist multiple a priori estimates, stated in different error norms – see Remark 1.4.1 below. Hereby we use the one given by El Karoui et. al in [16], which provides the following useful a priori estimate.

Lemma 1.4.1 (A Priori Estimate)
Let $\{(f^i, \xi^i), i = 1, 2\}$ be two sets of standard parameters satisfying Assumption 1.3.1. Denote the solutions of the corresponding BSDEs given in Equation 1.17 by $\{(Y^i, Z^i), i = 1, 2\}$. Let L be the Lipschitz constant of f^1, and put $\Delta Y_t := Y^1_t - Y^2_t$, $\Delta Z_t := Z^1_t - Z^2_t$ and $\Delta_2 f_t := f^1(t, Y^2_t, Z^2_t) - f^2(t, Y^2_t, Z^2_t)$. Then for any triple of real numbers (μ, λ, β) admitting to $\mu > 0, \lambda^2 > L, \beta \geq L(2 + \lambda^2) + \mu^2$ the following inequalities hold

$$\|\Delta Y\|^2_\beta \leq T\left[e^{\beta T}\mathbb{E}\left[|\Delta Y_T|^2\right] + \frac{1}{\mu^2}\|\Delta_2 f\|^2_\beta\right], \tag{1.20}$$

$$\|\Delta Z\|^2_\beta \leq \frac{\lambda^2}{\lambda^2 - L}\left[e^{\beta T}\mathbb{E}\left[|\Delta Y_T|^2\right] + \frac{1}{\mu^2}\|\Delta_2 f\|^2_\beta\right]. \tag{1.21}$$

Proof. We closely follow the proof given in [16]. In what follows C always denotes a constant – depending only on T and d –, whose value may vary from line to line.

We are considering the BSDEs of the form for $i = 1, 2$

$$Y^i_t = \xi^i + \int_t^T f^i(s, Y^i_s, Z^i_s)\mathrm{d}s - \int_t^T Z^i_s \mathrm{d}W_s. \tag{1.22}$$

Using the triangular inequality it follows that

$$|Y^i_t| \leq |\xi^i| + \int_0^T |f^i(s, Y^i_s, Z^i_s)|\mathrm{d}s + \sup_{t\in[0,T]}\left|\int_t^T Z^i_s \mathrm{d}W_s\right|. \tag{1.23}$$

As the right hand side is independent from t, it follows that

$$\sup_{t\in[0,T]}|Y^i_t| \leq |\xi^i| + \int_0^T |f^i(s, Y^i_s, Z^i_s)|\mathrm{d}s + \sup_{t\in[0,T]}\left|\int_t^T Z^i_s \mathrm{d}W_s\right|. \tag{1.24}$$

Since, by assumption $(\xi^i, f^i), i = 1, 2$ are both standard parameters – i.e. f is Lipschitz continuous –, the first two terms are $\mathbb{L}^2_{\mathcal{F}_T}(\mathbb{R})$ random variables. To show that so is the last term, we use the algebraic inequality $(a+b)^2 \leq 2(a^2+b^2)$ and get

$$\begin{aligned}\left(\sup_{t\in[0,T]}\left|\int_t^T Z_s^i \mathrm{d}W_s\right|\right)^2 &= \left(\sup_{t\in[0,T]}\left|\int_0^T Z_s^i \mathrm{d}W_s - \int_0^t Z_s^i \mathrm{d}W_s\right|\right)^2 \\ &\leq 2\left|\int_0^T Z_s^i \mathrm{d}W_s\right|^2 + 2\sup_{t\in[0,T]}\left|\int_0^t Z_s^i \mathrm{d}W_s\right|^2 \\ &\leq 4\sup_{t\in[0,T]}\left|\int_0^t Z_s^i \mathrm{d}W_s\right|^2.\end{aligned} \tag{1.25}$$

Using the Burkholder-Davis-Gundy inequality Theorem 1.1.2, we conclude that

$$\mathbb{E}\left[\left(\sup_{t\in[0,T]}\left|\int_t^T Z_s^i \mathrm{d}W_s\right|\right)^2\right] \leq C\mathbb{E}\left[\int_0^T \left|Z_s^i\right|^2 \mathrm{d}s\right] < \infty, \tag{1.26}$$

as $Z^i \in \mathbb{H}^2_{\mathbb{F}}(\mathbb{R}^d)$ by the definition of a solution. Therefore, we derive that $\sup_{t\in[0,T]}\left|Y_t^i\right| \in \mathbb{L}^2_{\mathcal{F}_T}(\mathbb{R})$, and by similar reasoning – due to the additive closedness of Banach spaces – we also gather $\sup_{t\in[0,T]}|\Delta Y_t| \in \mathbb{L}^2_{\mathcal{F}_T}$.

Let us now apply Itô's lemma to the process $\widetilde{Y}_s := e^{\beta s}|\Delta Y_s|^2$. It follows that

$$\mathrm{d}\left(e^{\beta s}|\Delta Y_s|^2\right) = \left[\beta e^{\beta s}\Delta Y_s - 2e^{\beta s}\left(f^1(s,Y_s^1,Z_s^1) - f^2(s,Y_s^2,Y_s^2)\right)\Delta Y_s + |\Delta Z_s|^2\right]\mathrm{d}s \\ + 2e^{\beta s}\Delta Y_s \Delta Z_s \mathrm{d}W_s. \tag{1.27}$$

Integrating between t and T gives

$$e^{\beta t}|\Delta Y_t|^2 + \beta\int_t^T e^{\beta s}|\Delta Y_s|^2\mathrm{d}s + \int_t^T e^{\beta s}|\Delta Z_s|^2\mathrm{d}s \\ = e^{\beta T}|\Delta Y_T|^2 + 2\int_t^T e^{\beta s}\Delta Y_s(f_s^1 - f_s^2)\mathrm{d}s - 2\int_t^T e^{\beta s}\Delta Y_s\Delta Z_s \mathrm{d}W_s, \tag{1.28}$$

where we used the notations $f_s^i := f^i(s,Y_s^i,Z_s^i)$. By the Lipschitz continuity of each driver, we have that

$$\left|f_s^1 - f_s^2\right| \leq \left|f_s^1 - f_s^2\right| + |\Delta_2 f_s| \leq L\left(|\Delta Y_s| + |\Delta Z_s|\right) + |\Delta_2 f_s|. \tag{1.29}$$

Substituting this into the second term of the right-hand side in Equation 1.28 yields

$$\begin{aligned}2\int_t^T e^{\beta s}\Delta Y_s(f_s^1 - f_s^2)\mathrm{d}s &\leq 2\int_t^T e^{\beta s}\Delta Y_s\left|f_s^1 - f_s^2\right|\mathrm{d}s \\ &\leq 2\int_t^T e^{\beta s}\Delta Y_s\left[L\left(|\Delta Y_s| + |\Delta Z_s|\right) + |\Delta_2 f_s|\right]\mathrm{d}s.\end{aligned} \tag{1.30}$$

Using the algebraic inequality $2y(Lz+t) \leq \frac{Lz^2}{\lambda^2} + \frac{t^2}{\mu^2} + y^2\left(\mu^2 + L\lambda^2\right)$ for $\mu, \lambda > 0$ implies[1]

$$\int_t^T e^{\beta s}2\Delta Y_s\left[L|\Delta Z_s| + |\Delta_2 f_s|\right] \leq \int_t^T e^{\beta s}\left[\frac{L|\Delta Z_s|^2}{\lambda^2} + \frac{|\Delta_2 f_s|^2}{\mu^2} + (\mu^2 + L\lambda^2)|\Delta Y_s|^2\right]\mathrm{d}s. \tag{1.31}$$

Substituting this back into the original inequality Equation 1.28, we conclude

$$e^{\beta t}|\Delta Y_t|^2 + \beta\left[\int_t^T e^{\beta s}|\Delta Y_s|^2\mathrm{d}s + \frac{1}{\beta}\int_t^T e^{\beta s}|\Delta Z_s|^2\mathrm{d}s\right] \\ \leq e^{\beta T}|\Delta Y_T|^2 + \left[L(2+\lambda^2) + \mu^2\right]\int_t^T e^{\beta s}|\Delta Y_s|^2\mathrm{d}s + \frac{L}{\lambda^2}\int_t^T e^{\beta s}|\Delta Z_s|^2\mathrm{d}s \\ + \frac{1}{\mu^2}\int_t^T e^{\beta s}|\Delta_2 f_s|^2\mathrm{d}s - 2\int_t^T e^{\beta s}\Delta Y_s\Delta Z_s\mathrm{d}W_s. \tag{1.32}$$

[1] with $y = \Delta Y_s, z = \Delta Z_s$ and $t = \Delta_2 f_s$.

Choosing μ, λ, β such that $\beta \geq L(2+\lambda^2)+\mu^2$ and $\lambda^2 > L$, this inequality reduces to

$$e^{\beta t}|\Delta Y_t|^2 \leq e^{\beta T}|\Delta Y_T|^2 + \frac{1}{\mu^2}\int_t^T e^{\beta s}|\Delta_2 f_s|^2 \mathrm{d}s - 2\int_t^T e^{\beta s}\Delta Y_s \Delta Z_s \mathrm{d}W_s. \tag{1.33}$$

Integration over the whole time horizon then gives

$$\int_0^T e^{\beta t}|\Delta Y_t|^2 \mathrm{d}t \leq T\left[e^{\beta T}|\Delta Y_T|^2 + \frac{1}{\mu^2}\int_0^T e^{\beta s}|\Delta_2 f_s|^2 \mathrm{d}s\right] - 2\int_0^T \int_t^T e^{\beta s}\Delta Y_s \Delta Z_s \mathrm{d}W_s \mathrm{d}t, \tag{1.34}$$

where we applied the Fubini theorem.

Since $\sup_{t\in[0,T]}|\Delta Y_t| \in \mathbb{L}^2_{\mathcal{F}_T}(\mathbb{R})$, $\Delta Y \in \mathbb{H}^2_{\mathbb{F}}(\mathbb{R})$ and also $\Delta Z \in \mathbb{H}^2_{\mathbb{F}}(\mathbb{R}^d)$, by the Cauchy-Schwarz inequality we get

$$\mathbb{E}\left[\sqrt{\int_0^T |e^{\beta s}\Delta Y_s \Delta Z_s|^2 \mathrm{d}s}\right] \leq e^{\beta T}\mathbb{E}\left[\int_0^T |\Delta Y_s|^2 \mathrm{d}s\right]\mathbb{E}\left[\int_0^T |\Delta Z_s|^2 \mathrm{d}s\right] < \infty. \tag{1.35}$$

Consequently $\{e^{\beta s}\Delta Y_s \Delta Z_s\}_{s\in[0,T]} \in \mathbb{H}^1_{\mathbb{F}}(\mathbb{R}^d)$, which implies that the stochastic integral above has zero expectation. Therefore, taking expectations of the inequality above proves the upper bound for ΔY

$$\|\Delta Y\|^2_\beta \leq T\left[e^{\beta T}|\Delta Y_T|^2 + \frac{1}{\mu^2}\|\Delta f\|^2_\beta\right]. \tag{1.36}$$

Substituting this back into the inequality Equation 1.32, after identical steps as above, gives the upper bound for the control process

$$\|\Delta Z\|^2_\beta \leq \frac{\lambda^2}{\lambda^2 - L}\left[e^{\beta T}|\Delta Y_T|^2 + \frac{1}{\mu^2}\|\Delta_2 f\|^2_\beta\right], \tag{1.37}$$

concluding the proof. □

Using this a priori estimate, we can now prove the well-posedness of the BSDE problem in Equation 1.17. This is usually done by defining a contraction mapping over the product space $\mathbb{S}^2_{\mathbb{F}}(\mathbb{R})\times\mathbb{H}^2_{\mathbb{F}}(\mathbb{R}^d)$ and arguing that – due to Banach's fixed point theorem – the contraction has a unique fixed point. In what follows, we present the proof of El Karoui et. al given in [16, Theorem 2.1].

Theorem 1.4.1 (Existence of Unique Solutions – BSDEs)
Given standard parameters (f,ξ) satisfying Assumption 1.3.1, there exists a unique pair of random processes such that $(Y,Z)\in\mathbb{S}^2_{\mathbb{F}}(\mathbb{R})\times\mathbb{H}^2_{\mathbb{F}}(\mathbb{R}^d)$, and they satisfy

$$Y_t = \xi + \int_t^T f(s,Y_s,Z_s)\mathrm{d}s - \int_t^T Z_s \mathrm{d}W_s \tag{1.38}$$

$\mathbb{P}$-a.s.

Proof. Most of the work has already been done by Lemma 1.4.1. We use a fixed-point argument for the mapping $\Phi : (y,z)\mapsto(Y,Z)$ implicitly defined by the following equation

$$Y_t = \xi + \int_t^T f(s,y_s,z_s)\mathrm{d}s - \int_t^T Z_s \mathrm{d}W_s, \tag{1.39}$$

where $(y,z)\in\mathbb{S}^2_{\mathbb{F}}(\mathbb{R})\times\mathbb{H}^2_{\mathbb{F}}(\mathbb{R}^d)$.

The fact that (f,ξ) are standard parameters implies that $f(t,y_t,z_t)\in\mathbb{H}^2_{\mathbb{F}}(\mathbb{R})$, and consequently $X = \xi + \int_0^T f(s,y_s,z_s)\mathrm{d}s$ is a square-integrable $\mathcal{F}_T$-measurable random variable and $M_t := \mathbb{E}[X|\mathcal{F}_t] = \mathbb{E}\left[\xi + \int_0^T f(s,y_s,z_s)\mathrm{d}s \middle| \mathcal{F}_t\right]$ is a continuous square-integrable martingale. Then, by the martingale representation Theorem 1.1.3, we have that there exists a unique integrable process $\widetilde{Z}\in\mathbb{H}^2_{\mathbb{F}}(\mathbb{R}^d)$ such that

$$M_t = M_0 + \int_0^t \widetilde{Z}_s \mathrm{d}W_s. \tag{1.40}$$

Notice that the image of the mapping Φ satisfies

$$\begin{aligned} Y_t &= M_t - \int_0^t f(s, y_s, z_s)\mathrm{d}s \\ Z_t &= \widetilde{Z}_t. \end{aligned} \tag{1.41}$$

By the linearity of the conditional expectations we also have $Y_t = \mathbb{E}\left[\int_t^T f(s, y_s, z_s)\mathrm{d}s + \xi|\mathcal{F}_t\right]$, which naturally implies that $Y \in \mathbb{S}^2_{\mathbb{F}}(\mathbb{R})$. We therefore conclude that Φ is a mapping from the space $\mathbb{S}^2_{\mathbb{F}}(\mathbb{R}) \times \mathbb{H}^2_{\mathbb{F}}(\mathbb{R}^d)$ onto itself.

Let us now consider two elements of $\mathbb{S}^2_{\mathbb{F}}(\mathbb{R}) \times \mathbb{H}^2_{\mathbb{F}}(\mathbb{R}^d)$ which we will denote by $\{(y^i, z^i), i = 1, 2\}$. Let $\{(Y^i, Z^i), i = 1, 2\}$ be the corresponding mappings defined by Equation 1.39. Applying Lemma 1.4.1 with $\beta = \mu^2$ we have that – notice that in this case $f^1 = f^2, \xi^1 = \xi^2$ and thus $L = 0, \Delta Y_T = 0$ –

$$\begin{aligned} \|\Delta Y\|^2_\beta &\leq \frac{T}{\beta}\mathbb{E}\left[\int_0^T e^{\beta s}\left|f(s, y^1_s, z^1_s) - f(s, y^2_s, z^2_s)\right|^2 \mathrm{d}s\right], \\ \|\Delta Z\|^2_\beta &\leq \frac{1}{\beta}\mathbb{E}\left[\int_0^T e^{\beta s}\left|f(s, y^1_s, z^1_s) - f(s, y^2_s, z^2_s)\right|^2 \mathrm{d}s\right]. \end{aligned} \tag{1.42}$$

Since the driver of the BSDE is Lipschitz continuous we have that

$$\left|f(s, y^1_s, z^1_s) - f(s, y^2_s, z^2_s)\right| \leq L\left[|\Delta y_s| + |\Delta z_s|\right], \tag{1.43}$$

where $\Delta y := y^1 - y^2, \Delta z := z^1 - z^2$. Using the algebraic inequality $(a+b)^2 \leq 2(a^2 + b^2)$ once again, we conclude that

$$\|\Delta Y\|^2_\beta + \|\Delta Z\|^2_\beta \leq \frac{2(1+T)L}{\beta}\left[\|\Delta y\|^2_\beta + \|\Delta z\|^2_\beta\right]. \tag{1.44}$$

Let us now choose μ^2 such that $\mu^2 = \beta > 2(1+T)L$. This clearly implies that the mapping $\Phi : \mathbb{S}^2_{\mathbb{F}}(\mathbb{R}) \times \mathbb{H}^2_{\mathbb{F}}(\mathbb{R}^d) \to \mathbb{S}^2_{\mathbb{F}}(\mathbb{R}) \times \mathbb{H}^2_{\mathbb{F}}(\mathbb{R}^d)$ (implicitly) defined by Equation 1.39 is not just a mapping from a metric space – spaces equipped with the norm $\|(y, z)\|^2_\beta := \|y\|^2_\beta + \|z\|^2_\beta$ – onto itself, but it also holds that $\exists K \in [0, 1)$ such that $\left\|\Phi(y^1, z^1) - \Phi(y^2, z^2)\right\|_\beta \leq K\left\|(y^1, z^1) - (y^2, z^2)\right\|_\beta$, $\forall (y^1, z^2), (y^2, z^2) \in \mathbb{S}^2_{\mathbb{F}}(\mathbb{R}) \times \mathbb{H}^2_{\mathbb{F}}(\mathbb{R}^d)$. Therefore Φ is a contraction mapping, and consequently by Banach's fixed-point theorem Theorem 1.1.1 there exists a unique fixed point $\mathbf{x}^* := (Y^*, Z^*) \in \mathbb{S}^2_{\mathbb{F}}(\mathbb{R}) \times \mathbb{H}^2_{\mathbb{F}}(\mathbb{R}^d)$ such that $\Phi(\mathbf{x}^*) = \mathbf{x}^*$. For this unique fixed-point we have

$$Y^*_t = \xi + \int_t^T f(s, Y^*_s, Z^*_s)\mathrm{d}s - \int_t^T Z^*_s \mathrm{d}W_s, \tag{1.45}$$

$Y^* \in \mathbb{S}^2_{\mathbb{F}}(\mathbb{R}), Z^* \in \mathbb{H}^2_{\mathbb{F}}(\mathbb{R}^d)$, meaning that (Y^*, Z^*) is the solution of the original BSDE indeed. This concludes the proof. □

The proof naturally establishes a way to construct the solution which – following the derivation of [16] – we collect in the following corollary.

Corollary 1.4.1 (Picard Iterations)
Let β satisfy $2(T+1)L < \beta$ for a BSDE with a driver with Lipschitz constant L. Let (Y^k, Z^k) be a Picard sequence defined recursively by

$$\begin{aligned} Y^0 &= 0, Z^0 = 0, \\ Y^{k+1}_t &= \xi + \int_t^T f(s, Y^k_s, Z^k_s)\mathrm{d}s - \int_t^T Z^k_s \mathrm{d}W_s, \end{aligned} \tag{1.46}$$

where Y^0, Z^0 are constant zero random processes taking values in $\mathbb{R}$ and $\mathbb{R}^{1\times d}$ respectively. Then the sequence (Y^k, Z^k) converges to the solution (Y, Z) $\mathrm{d}\mathbb{P} \times \mathrm{d}t$-a.s.

Proof. For the mapping $\Phi' : (Y^k, Z^k) \mapsto (Y^{k+1}, Z^{k+1})$ defined implicitly by Equation 1.46 we have that

$$\begin{aligned} k = 1: \quad & \left\|Y^2 - Y^1\right\|^2_\beta + \left\|Z^2 - Z^1\right\|^2_\beta && \leq \frac{2(1+T)L}{\beta}\left[\left\|Y^1 - Y^0\right\|^2_\beta + \left\|Z^1 - Z^0\right\|^2_\beta\right] \\ k = 2: \quad & \left\|Y^3 - Y^2\right\|^2_\beta + \left\|Z^3 - Z^2\right\|^2_\beta && \leq \frac{2(1+T)L}{\beta}\left[\left\|Y^2 - Y^1\right\|^2_\beta + \left\|Z^2 - Z^1\right\|^2_\beta\right] \\ & && \leq \left(\frac{2(1+T)L}{\beta}\right)^2\left[\left\|Y^1 - Y^0\right\|^2_\beta + \left\|Z^1 - Z^0\right\|^2_\beta\right] \\ k \geq 3: \quad & && \dots \end{aligned}$$

by Equation 1.44. Introducing the notation $\varepsilon := \frac{2(1+T)L}{\beta}$, $K := \left\|Y^1 - Y^0\right\|_\beta^2 + \left\|Z^1 - Z^0\right\|_\beta^2$ we therefore have

$$\left\|Y^{k+1} - Y^k\right\|_\beta^2 + \left\|Z^{k+1} - Z^k\right\|_\beta^2 \leq \varepsilon^k K. \tag{1.47}$$

By the assumption $2(T+1)L < \beta$ we get $\varepsilon < 1$, which implies

$$\sum_{k=0}^{\infty} \left\|(\Delta Y^k, \Delta Z^k)\right\|_{\beta'}^2 := \sum_{k=0}^{\infty} \left[\left\|Y^{k+1} - Y^k\right\|_\beta^2 + \left\|Z^{k+1} - Z^k\right\|_\beta^2\right] = K \sum_{k=0}^{\infty} \varepsilon^k = K \frac{1}{1-\varepsilon} < +\infty, \tag{1.48}$$

by the summation formula of geometric series. Therefore the Picard series indeed converges, concluding our proof. □

Remark 1.4.1 (Supremum Norm)
In the above results we were using the norm defined by

$$\|(Y,Z)\|_\beta^2 := \mathbb{E}\left[\int_0^T e^{\beta s}|Y_s|^2 \mathrm{d}s\right] + \mathbb{E}\left[\int_0^T e^{\beta s}|Z_s|^2 \mathrm{d}s\right]$$

and gave upper bounds for convergence in metric spaces equipped with this norm, implying the existence of a unique solution in $\mathbb{S}_{\mathbb{F}}^2(\mathbb{R}) \times \mathbb{H}_{\mathbb{F}}^2(\mathbb{R}^d)$. A careful inspection of the proof of Theorem 1.4.1 shows that this result can be expanded to the space $\widehat{\mathbb{S}}_{\mathbb{F}}^2(\mathbb{R}) \times \mathbb{H}_{\mathbb{F}}^2(\mathbb{R}^d)$, where $\widehat{\mathbb{S}}_{\mathbb{F}}^2(\mathbb{R})$ is the space of all progressively $\mathbb{F}$-measurable, continuous random processes $\Phi : \Omega \times [0,T] \to \mathbb{R}$ such that $\mathbb{E}\left[\sup_{t\in[0,T]} |\Phi_t|^2\right] < \infty$. Similarly, a(nother) priori estimate can be proven to upper bound the norm

$$\|(Y,Z)\|^2 := \mathbb{E}\left[\sup_{0\leq t\leq T} |Y_t|^2 + \int_0^T |Z_s|^2 \mathrm{d}s\right], \tag{1.49}$$

leading to a different contraction mapping but the same convergence results. Consequently one can prove in a similar fashion that the Picard sequence above also converges for $\sup_{t\in[0,T]} \left|Y_t^k - Y_t\right|$ to 0, $\mathbb{P}$-a.s. For a proof on the latter, we refer the more interested reader to, e.g., [12].

1.4.2 Representation of Linear BSDEs

Later in this work, for the algorithms proposed on the Malliavin derivative's BSDE introduced in chapter 2, the class of so-called linear BSDEs – i.e. where the driver f is a linear function of Y, Z – plays a fundamental role. Therefore it is of interest to see how the solution of such linear BSDEs looks like. In what follows, we present a representation formula for the solution of a linear BSDE built on the formulation of [12], which shall be useful in the alternative representations of the Malliavin derivative of the Y-process – see section 2.6 in particular.

Proposition 1.4.1 (Representation Formula – Linear BSDEs)
Let $\xi \in \mathbb{L}_{\mathcal{F}_T}^2(\mathbb{R}), \gamma \in \mathbb{H}_{\mathbb{F}}^2(\mathbb{R}), \alpha \in \mathbb{L}_{\mathbb{F}}^\infty(\mathbb{R}), \beta \in \mathbb{L}_{\mathbb{F}}^\infty(\mathbb{R}^d)$ bounded, progressively measurable processes. Then, if the pair of processes (Y,Z) satisfies the linear BSDE

$$Y_t = \xi + \int_t^T \left[\gamma_s + \alpha_s Y_s + Z_s \beta_s\right] \mathrm{d}s - \int_t^T Z_s \mathrm{d}W_s, \tag{1.50}$$

the solution admits the following representation

$$Y_t = \Gamma_t^{-1} \mathbb{E}\left[\Gamma_T \xi + \int_t^T \Gamma_s \gamma_s \mathrm{d}s \middle| \mathcal{F}_t\right], \tag{1.51}$$

where

$$\Gamma_t := 1 + \int_0^t \alpha_s \Gamma_s \mathrm{d}s + \int_0^t \beta_s \Gamma_s \mathrm{d}W_s \tag{1.52}$$

is defined by a linear forward SDE.

Proof. We are closely following the proof given in [12]. First of all, let us highlight that the boundedness of α, β implies that the driver defined as $f(t, y, z) := \gamma_s + \alpha_s y + z\beta_s$ is uniformly Lipschitz continuous and therefore (ξ, f) are standard parameters satisfying Assumption 1.3.1. By the results above, we therefore have that the linear BSDE above does indeed have a unique solution (Y, Z).

Then the representation formula is a simple consequence of Itô's lemma. Applying the multidimensional Itô's lemma for the process $\widetilde{Y}_t := \Gamma_t Y_t$ yields

$$\mathrm{d}\widetilde{Y}_t = -\Gamma_t \gamma_t \mathrm{d}t + \Gamma_t \left[Y_t \beta_t^T + Z_t\right] \mathrm{d}W_t. \tag{1.53}$$

Introducing the notations

$$\widetilde{Y}_t := \Gamma_t Y_t, \quad \widetilde{Z} := \Gamma_t \left[Y_t \beta_t + Z_t\right], \quad \widetilde{\xi} := \Gamma_T \xi, \quad \widetilde{\gamma}_t := \Gamma_t \gamma_t,$$

we immediately get

$$\widetilde{Y}_t = \widetilde{\xi} + \int_t^T \widetilde{\gamma}_s \mathrm{d}s - \int_t^T \widetilde{Z}_s \mathrm{d}W_s. \tag{1.54}$$

This is a BSDE with a *"constant"* driver $\widetilde{f}(t, Y_t, Z_t) := \widetilde{\gamma}_t$, i.e. the driver does not depend on (Y, Z). From the assumptions it follows that $(\widetilde{\xi}, \widetilde{f})$ are also standard parameters, implying that this latter BSDE has a unique solution $(\widetilde{Y}, \widetilde{Z}) \in \mathbb{S}^2_{\mathbb{F}}(\mathbb{R}) \times \mathbb{H}^2_{\mathbb{F}}(\mathbb{R}^d)$ too. From this, we derive that the last term's Itô-integral has zero expectation. Consequently, taking conditional expectations gives

$$\widetilde{Y}_t = \mathbb{E}\left[\widetilde{\xi} + \int_t^T \widetilde{\gamma}_s \mathrm{d}s \middle| \mathcal{F}_t\right] \implies Y_t = \Gamma_t^{-1} \mathbb{E}\left[\Gamma_T \xi + \int_t^T \Gamma_s \gamma_s \mathrm{d}s \middle| \mathcal{F}_t\right],$$

concluding the proof. □

1.5 Forward-Backward Stochastic Differential Equations

In this work, we consider a special class of BSDEs, namely a system of so-called Forward-Backward Stochastic Differential Equations (FBSDE) where the randomness in the backward equation is coming from a forward SDE. These systems, in the most general setting, can be described by the following set of equations

$$X_t = x_0 + \int_0^t \mu(s, X_s, Y_s, Z_s)\mathrm{d}s + \int_0^t \sigma(s, X_s, Y_s, Z_s)\mathrm{d}W_s, \qquad 0 \le t \le T, \tag{1.55a}$$

$$Y_t = g(X_T) + \int_t^T f(s, X_s, Y_s, Z_s)\mathrm{d}s - \int_t^T Z_s \mathrm{d}W_s, \qquad 0 \le t \le T. \tag{1.55b}$$

Notice that in the formulation above we allowed for the so-called *coupling*, meaning that the solution of the forward equation depends on the solutions of the backward equation. This poses a challenge in terms of constructing proofs for the existence of unique solutions, as our results in the previous sections are not applicable due to the interdependence.

Nevertheless, in the rest of the work, we only consider the above system in a decoupled manner, i.e. we do not allow for the forward process to depend on (Y, Z):

$$X_t = x_0 + \int_0^t \mu(s, X_s)\mathrm{d}s + \int_0^t \sigma(s, X_s)\mathrm{d}W_s, \qquad 0 \le t \le T, \tag{1.56a}$$

$$Y_t = g(X_T) + \int_t^T f(s, X_s, Y_s, Z_s)\mathrm{d}s - \int_t^T Z_s \mathrm{d}W_s, \qquad 0 \le t \le T, \tag{1.56b}$$

where $f : [0, T] \times \mathbb{R}^d \times \mathbb{R} \times \mathbb{R}^{1\times d} \to \mathbb{R}$ is a measurable function. In this setting, solving the equation above can simply be divided into two parts: solving the SDE of the forward equation first, and then plugging the resulting randomness in the backward part. This means that if the coefficients in Equation 1.56a are such that μ, σ, x_0 satisfy Assumption 1.2.1 then the forward part of the equation has a unique solution provided by Theorem 1.2.1. However, in the setting above the backward equation's driver and terminal condition also depend on the solution of the forward SDE, therefore the results obtained before cannot be directly applied for the existence of a unique solution for the backward equation in the system above. In order to guarantee the well-posedness of the FBSDE problem, we need to make further assumptions.

Assumption 1.5.1 (Unique Solutions of FBSDEs)
Let the parameters of Equation 1.56a and Equation 1.56b be such that

1. x_0, μ, σ *satisfy Assumption 1.2.1;*

2. $f : [0,T] \times \mathbb{R}^d \times \mathbb{R} \times \mathbb{R}^{1\times d} \to \mathbb{R}$ *is uniformly Lipschitz continuous with respect to* y *and* z*, i.e. there exists* L *such that* $\forall (t,x,y_1,z_1),(t,x,y_2,z_2) \in [0,T] \times \mathbb{R}^d \times \mathbb{R} \times \mathbb{R}^{1\times d}$

$$|f(t,x,y_1,z_1) - f(t,x,y_2,z_2)| \leq L\left(|y_1 - y_2| + |z_1 - z_2|\right); \tag{1.57}$$

3. f *and* g *admit at most polynomial growth in* x*, i.e. for* $p \geq \frac{1}{2}$ *there exists* C *such that* $\forall (t,x,y,z) \in [0,T] \times \mathbb{R}^d \times \mathbb{R} \times \mathbb{R}^{1\times d}$

$$|f(t,x,y,z)| + |g(x)| \leq C\left(1 + |x|^p\right). \tag{1.58}$$

If $g(\cdot), f(\cdot,\cdot,\cdot,\cdot)$ satisfy Assumption 1.5.1 above then we call them *standard parameters*. The following theorem establishes the existence of a unique solution under the assumption of standard parameters.

Theorem 1.5.1 (Existence of Unique Solutions – FBSDEs)
Let the parameters of Equation 1.56a and Equation 1.56b satisfy Assumption 1.5.1. Then the system of FBSDEs has a unique solution, namely a triple of random processes $(X,Y,Z) \in \widehat{\mathbb{S}}^2_{\mathbb{F}}(\mathbb{R}^d) \times \widehat{\mathbb{S}}^2_{\mathbb{F}}(\mathbb{R}) \times \mathbb{H}^2_{\mathbb{F}}(\mathbb{R}^d)$.

Proof. See, e.g., [16]. □

1.5.1 A Word on Markovianity

In the rest of this work we are relying on another important property of the decoupled FBSDE system above, which we have so far swept under the carpet. This is related to Markovianity. By the nature of SDEs it easily follows that the solutions of the following equations

$$X^{s,x}_t = x + \int_s^t \mu(r, X^{s,x}_r)\mathrm{d}r + \int_s^t \sigma(r, X^{s,x}_r)\mathrm{d}W_r, \qquad s \leq t \leq T, \tag{1.59}$$

$$X^{0,x_0}_t = x_0 + \int_0^t \mu(r, X^{0,x_0}_r)\mathrm{d}r + \int_0^t \sigma(r, X^{0,x_0}_r)\mathrm{d}W_r, \qquad 0 \leq t \leq T, \tag{1.60}$$

coincide $\mathbb{P}$-a.s. for $x \coloneqq x_0 + \int_0^s \mu(r, X^{0,x_0}_r)\mathrm{d}r + \int_0^s \sigma(r, X^{0,x_0}_r)\mathrm{d}W_r$. This shows that the unique solution of an SDE satisfying Assumption 1.2.1 is Markovian. In the above formulation we could have defined the backward equation in the following manner

$$Y^{s,x}_t = g(X^{s,x}_T) + \int_t^T f(r, X^{s,x}_r, Y^{s,x}_r, Z^{s,x}_r)\mathrm{d}r - \int_t^T Z^{s,x}_r \mathrm{d}W_r, \qquad s \leq t \leq T. \tag{1.61}$$

Now, by the shifting properties of Brownian motion it can be shown that the solution of the above BSDE inherits the measurability properties of the Markov SDE's solution $X^{s,x}_t$ in Equation 1.59 – see, e.g., [16, Proposition 4.2]. Therefore we conclude that for BSDEs of the form Equation 1.61, the solutions $\left(Y^{t,x}_t, Z^{t,x}_t\right)$ are deterministic functions of $X^{t,x}_t$ and time:

$$Y^{t,x}_t = u(t, X^{t,x}_t), \quad Z^{t,x}_t = v(t, X^{t,x}_t), \tag{1.62}$$

for some $u : [0,T] \times \mathbb{R}^d \to \mathbb{R}, v : [0,T] \times \mathbb{R}^d \to \mathbb{R}^d$.

In the light of the observations above, we now simplify our notation for the rest of this work. Instead of considering a Markov FBSDE system of the form Equation 1.59 and Equation 1.61, we only consider forward diffusions starting off from a fixed, known initial condition x_0 at $t = 0$. In order to ease the notation, we drop the superscripts describing the initial state of the randomness and work with equations as before, where $X_t = X^{0,x_0}_t, Y_t = Y^{0,x_0}_t, Z_t = Z^{0,x_0}_t$. Nevertheless, it is essential to keep in mind that the reason why we can later specify deterministic mappings of the form $u : (t, X_t) \mapsto Y_t$ is inherently linked to Markovianity.

1.5.2 Generalized Feynman–Kac Relations

As we have just seen, due to Markovianity, the solution of the BSDE is – for each point in time – a deterministic mapping of the solution of the forward SDE. The following theorem describes in more depth, what the nature of this mapping is. Recall that the solution of a linear, second-order parabolic PDE is equivalent to the solution of a forward SDE given by the Feynman–Kac lemma Theorem 1.2.2. The fundamental power of BSDEs lies in the generalization of this theorem, which provides a similar connection between second-order, quasi-linear parabolic PDEs and BSDEs. Hereby, in order to avoid going too deep in functional analysis, we only state the theorem under rather strict smoothness assumptions.

Theorem 1.5.2 (Generalized Feynman–Kac Theorem)
Consider the following quasi-linear parabolic PDE of the form

$$\begin{aligned}\frac{\partial u}{\partial t}+\frac{1}{2}\operatorname{Tr}\left[\sigma\sigma^T\operatorname{Hess}u(t,x)\right]+\mu(t,x)\nabla u(t,x)+f(t,x,u,\nabla u)(t,x)=0,\\ u(T,x)=g(x),\end{aligned}\tag{1.63}$$

and the system of decoupled FBSDEs

$$\begin{aligned}X_t&=x+\int_0^t\mu(s,X_s)\mathrm{d}s+\int_0^t\sigma(s,X_s)\mathrm{d}W_s,\\ Y_t&=g(X_T)+\int_t^T f(s,X_s,Y_s,Z_s)\mathrm{d}s-\int_t^T Z_s\mathrm{d}W_s.\end{aligned}\tag{1.64}$$

Assume that the solution of the PDE satisfies $u\in C^{1,2}([0,T]\times\mathbb{R}^d)$, and that the conditions of Assumption 1.5.1 are satisfied. Then the solution of the PDE problem coincides with the solutions of the corresponding Markovian BSDE

$$u(t,X_t)=Y_t,\quad(\nabla u)(t,X_t)=\sigma^{-1}(t,X_t)Z_t,\tag{1.65}$$

$\mathbb{P}$-a.s.

Proof. The proof is a straightforward consequence of applying Itô's lemma to the solution $u(t,X_t)$

$$\mathrm{d}u(t,X_t)=\left[\frac{\partial u}{\partial t}+\mu\nabla u+\frac{\sigma\sigma^T}{2}\operatorname{Hess}u\right](t,X_t)\,\mathrm{d}t+(\sigma\nabla u)(t,X_t)\,\mathrm{d}W_t\tag{1.66}$$

$$=-f(t,X_t,u(t,X_t),(\nabla u)(t,X_t))\mathrm{d}t+(\sigma\nabla u)(t,X_t)\mathrm{d}W_t,\tag{1.67}$$

with $u(T,X_T)=g(X_T)$ since $u(t,x)$ satisfies the PDE for each t,x. □

It is important to mention that the assumptions of this theorem can be further relaxed if we allow for a weaker concept of the solution of the PDE, the so-called *viscosity solutions*. As our future assumptions – see Assumption 2.5.1 in particular – shall require the problem to possess certain smoothness properties, we omit these difficulties here and refer the more interested reader to, e.g., [2].

Chapter 2

Malliavin Calculus in the Context of BSDEs

In the following chapter we give a high-level introduction to the concept of Malliavin calculus, mostly focused on the derivative operator in the context of FBSDEs. After fixing some additional notations, we start by defining the derivative operator in the Malliavin sense. Thereafter, we present the main properties of Malliavin differentiation, out of which we emphasize the importance of the Malliavin chain rule stated in Lemma 2.3.1. After this, we turn to FBSDEs. We first show that under certain regularity conditions the Malliavin derivative of a forward SDE satisfies a linear forward SDE itself. Subsequently, we present a similar result for BSDEs and show that the Malliavin derivatives of the solutions of BSDEs satisfy a linear BSDE themselves. Moreover, we shall also see that the Z-process can be interpreted as a Malliavin derivative of the Y-process which is of fundamental importance with respect to the upcoming algorithmic proposals in chapter 5, as it establishes the regularity and continuity of the control process. Finally, we conclude the chapter by stating certain alternative representations for the solution of Malliavin derivative's linear BSDE.

Since Malliavin calculus is not the main subject but rather a tool in this work, in order to avoid getting lost in stochastic theory we keep the discussion brief. Throughout the whole chapter we restrict the presentation to the statement of results without proofs. We refer the more interested reader to one of the great introductions [17] or [18] for more details on the theory of Malliavin calculus.

2.1 Preliminaries

As in the previous chapter, we are concerned with a filtered probability space $(\Omega, \mathcal{F}, \mathbb{F}, \mathbb{P})$, where $\mathbb{F} \coloneqq \{\mathcal{F}_t\}_{t\in[0,T]}$ is the natural filtration generated by a d-dimensional Brownian motion. On top of the notations we have already introduced in section 1.1, we introduce the following additional definitions.

Definition 2.1.1 (Notations, spaces)**.** We use the following notations.

1. $C_p^k(\mathbb{R}^d; \mathbb{R}^n)$: set of k times continuously differentiable functions of the form $f : \mathbb{R}^d \to \mathbb{R}^n$ such that all the partial derivatives of f have at most polynomial growth of order p;
2. $C_b^k(\mathbb{R}^d; \mathbb{R}^n)$: set of k times continuously differentiable functions of the form $f : \mathbb{R}^d \to \mathbb{R}^n$ such that all the partial derivatives of f are bounded;
3. $L^2([0,T]; R^n)$: the Hilbert space of real L^2-integrable functions over a finite time interval $[0,T]$, i.e. $h : [0,T] \to \mathbb{R}^n$ such that $\int_0^T |h(t)|^2 \mathrm{d}t < \infty$, equipped with the inner product $\langle h|g\rangle_{L^2} \coloneqq \int_0^T h(t)g(t)\mathrm{d}t$. We denote the norm of these spaces by $\|x\|_{L^2([0,T];\mathbb{R}^n)} \coloneqq \sqrt{\langle x|x\rangle}$;
4. $\partial_i f$: for any multivariate deterministic scalar function of the form $f(x) : \mathbb{R}^n \to \mathbb{R}$ we denote $\partial_i f \coloneqq \frac{\partial f}{\partial x_i}$;
5. $\nabla_x f, \nabla_y f, \nabla_z f$: for deterministic multivariate functions of the form $f(t,x,y,z) : \mathbb{R} \times \mathbb{R}^n \times \mathbb{R}^m \times \mathbb{R}^p \to \mathbb{R}$ we put $\nabla_x f \coloneqq \left(\frac{\partial f}{\partial x_1}, \dots, \frac{\partial f}{\partial x_n}\right)$, $\nabla_y f \coloneqq \left(\frac{\partial f}{\partial y_1}, \dots, \frac{\partial f}{\partial y_m}\right)$ and $\nabla_z f \coloneqq \left(\frac{\partial f}{\partial z_1}, \dots, \frac{\partial f}{\partial z_p}\right)$.

In what follows, we may sometimes rely on the Markovian notation introduced in subsection 1.5.1. The need for this becomes apparent during the presentation of the variational processes corresponding to the solutions of SDEs and BSDEs – see Equation 2.16 and Equation 2.27 in particular.

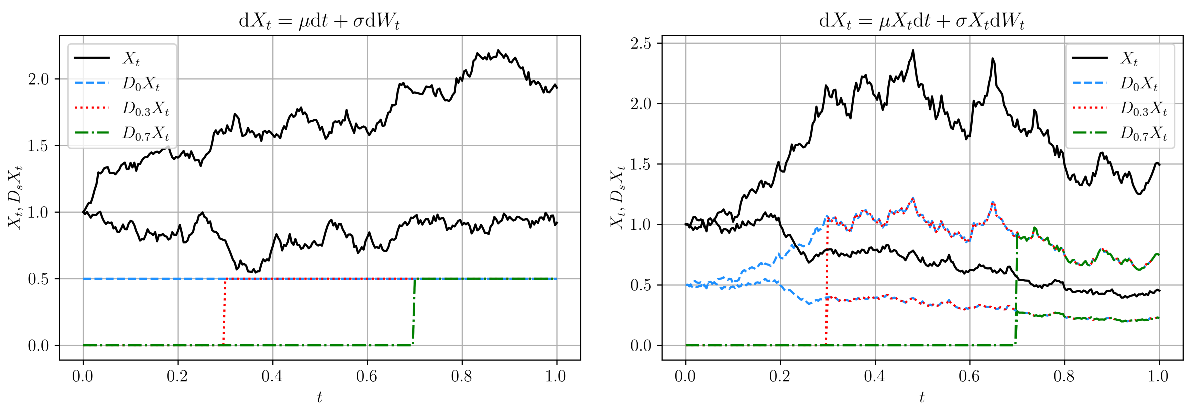


Figure 2.1: Malliavin Derivative Illustration. Two realizations of the driving Brownian motion. Left: Arithmetic Brownian Motion ($\mu = 0, \sigma = 0.5$), right: Geometric Brownian Motion ($\mu = 0.2, \sigma = 0.5$).

2.2 An Overview of Malliavin Calculus

In the following section we give a brief introduction to the concepts of Malliavin calculus mostly focusing on the derivative operator in the case where the randomness is generated by a Brownian motion – see section 1.1. In order to ease the notation, we restrict the presentation to the one-dimensional case, nevertheless most results below also hold for multidimensional random variables. We only recall the most fundamental concepts of Malliavin calculus and refer to [17, Chapter 1.2] for more details. Our goal is to introduce a concept of differentiation for square integrable random variables $X : \Omega \to \mathbb{R}$, hence we want to interpret the *"derivative"* of $X(\omega)$ with respect to the randomness ω.

In order to do this, let us first recall the definition of *isonormal Gaussian random processes*. Considering a separable real Hilbert space H, the stochastic processes $W(h)$ which are centered Gaussians such that $\forall h, g \in H$: $\mathbb{E}\left[W(h)W(g)\right] = \langle h|g\rangle_H$ are called isonormal Gaussian. It is known that $h \mapsto W(h)$ defines a linear isometry from H onto a closed subspace of $\mathbb{L}^2_{\mathcal{F}_T}(\mathbb{R})$ which we denote by $\mathcal{H}_1$.

From now on we only consider a special case of isonormal Gaussian processes given by the following form

$$W(h) := \int_0^T h(s)\mathrm{d}W_s, \tag{2.1}$$

where $h \in L^2([0,T];\mathbb{R})$ and $\{W_s\}_{s\in[0,T]}$ is the Brownian motion generating the filtration.[1] It can easily be checked that the mapping above truly defines an isonormal Gaussian random variable, indeed by Itô-isometry we have

$$\mathbb{E}\left[W(h)W(g)\right] := \mathbb{E}\left[\left(\int_0^T h(s)\mathrm{d}W_s\right)\left(\int_0^T g(s)\mathrm{d}W_s\right)\right]\mathbb{E}\left[\int_0^T h(s)g(s)\mathrm{d}s\right] = \langle h|g\rangle_{L^2}. \tag{2.2}$$

Therefore the closed subspace of such random variables $\mathcal{H}_1 := \left\{W(h)\colon h \in L^2([0,T])\right\} \subset \mathbb{L}^2_{\mathcal{F}_T}(\mathbb{R})$.

Let us now define *smooth random variables* of the form

$$F = f(W(h_1), \dots, W(h_n)), \tag{2.3}$$

where $n \geq 1$, $f \in C_p^\infty(\mathbb{R}^n;\mathbb{R})$, $\forall i : h_i \in L^2([0,T];\mathbb{R})$ and $W(h_i)$ is the isonormal Gaussian process defined by Equation 2.1. We denote the set of such smooth random variables by $\mathcal{S} := \left\{F\colon F = f(W(h_1),\dots,W(h_n)), f \in C_p^\infty(\mathbb{R}^n;\mathbb{R}), h_i \in L^2([0,T]), \forall i\right\}$. Similarly, we use the notation $\mathcal{P}$ for random variables of the form Equation 2.3, where f is a polynomial. It is worth to notice that $\mathcal{S}$ is a dense subset of $L^2_{\mathcal{F}_T}(\mathbb{R})$ and additionally $\mathcal{P} \subset \mathcal{S}$.

Given the notations above, let us now define the Malliavin derivative of smooth random variables.

[1]It is important to notice that $W(h)$ and W_s are not the same quantities. The former is a random variable defined by Equation 2.1 whereas the latter is the Brownian motion at time s. This, although may be confusing at first, is in accordance with the standard notations of the literature.

Definition 2.2.1 (Malliavin Derivative)**.** The Malliavin derivative $D_sF : \Omega \to \mathbb{R}^n$ of a random variable $F \in \mathcal{S}$ of the form Equation 2.3 is, for each $0 \leq s \leq T$, defined as

$$D_sF := \sum_{i=1}^{n} \partial_i f(W(h_1), \dots, W(h_n))h_i(s). \tag{2.4}$$

For instance, we have $D_sW(h) = h$.

The definition above only establishes the concept of Malliavin differentiation for random variables of the form Equation 2.3. However, it can be extended to a wider range of random processes. In fact, for $F \in \mathcal{S}, p > 1$ one can define the following norm

$$\|F\|_{1,p} := \left(\mathbb{E}\left[|F|^p + \left(\int_0^T |D_sF|^2 \mathrm{d}s\right)^{p/2}\right]\right)^{1/p}. \tag{2.5}$$

It can be shown – see [17, Proposition 1.2.1] – that the operator D defined in Definition 2.2.1 has a closed extension with respect to the norm Equation 2.5. We denote this closure by $\mathbb{D}^{1,p}$ and for the rest of the chapter focus on random variables $X \in \mathbb{D}^{1,p}$. In particular, for $p = 2$ we have that $\mathbb{D}^{1,2}$ is a Hilbert space with the scalar product

$$\langle F|G\rangle_{\mathbb{D}^{1,2}} := \mathbb{E}[FG] + \mathbb{E}[\langle DF|DG\rangle_{L^2}]. \tag{2.6}$$

Since these results also hold in the general multidimensional case, from this point on we also use the notation $\mathbb{D}^{d,p}$ for the space of d-dimensional Malliavin differentiable random variables.

Finally, it needs to be mentioned that Definition 2.2.1 can straightforwardly be extended to *higher-order Malliavin derivatives* – see, e.g., [19, Pg. 13] – which are usually denoted by D^kF for the k'th Malliavin derivative of random variables $F \in \mathcal{S}$. However, in the rest of this work we only deal with first-order Malliavin derivatives.

2.3 Malliavin Chain Rule and Properties of the Derivative Operator

It is well-known from the theory of real calculus that the classical differentiation operator exhibits certain advantageous properties, such as the chain rule. In the following section we demonstrate that the Malliavin differentiation operator given by Definition 2.2.1 admits to a similar formula, a result which is essential for the upcoming algorithmic formulations in chapter 5.

First, let us highlight that by Definition 2.2.1 it follows that for any $F, G \in \mathbb{D}^{1,p}$ and $a, b \in \mathbb{R}$: $D_s(aF + bG) = aD_sF + bD_sG$, i.e. the Malliavin derivative is indeed a linear operator. Furthermore, as a consequence of the following integration-by-parts formula – proven in [17, Lemma 1.2.1] –

$$\mathbb{E}[\langle DF|h\rangle_{L^2}] = \mathbb{E}[FW(h)], \tag{2.7}$$

we also have that for any $F, G \in \mathbb{D}^{1,p}$, $D_s(FG) = D_sFG + FD_sG$. These results naturally raise the question whether a "chain rule like" result also holds for the Malliavin derivative operator. Turns out that it indeed does, established by the following lemma.

Lemma 2.3.1 (Malliavin Chain Rule)
Let $\varphi \in C_b^1(\mathbb{R}^d; \mathbb{R})$ and fix $p \geq 1$. Let $F := (F_1, \dots, F_d) \in \mathbb{D}^{1,p}$. Then $\varphi(F) \in \mathbb{D}^{1,p}$ and for each $0 \leq s \leq T$

$$D_s\varphi(F) = \sum_{i=1}^{d} \partial_i\varphi(F)D_sF_i. \tag{2.8}$$

Proof. See, e.g., [17, Proposition 1.2.3]. □

We remark that the chain rule can be extended to Lipschitz continuous functions of $F \in \mathbb{D}^{1,p}$ – see [17, Proposition 1.2.4]. Note that for the continuously differentiable case, this is a direct consequence of the Mean Value Theorem, which implies that continuously differentiable functions whose derivatives are bounded, are also Lipschitz continuous.

As a result, one can derive closed-form analytical expressions for the Malliavin derivative in some special cases. In fact, with a careful inspection[2] of Equation 2.1 and Definition 2.2.1 we gather that $D_sW_t = \mathbb{1}_{s\leq t}(t)$. This, together with the Malliavin Chain Rule, implies that for any function φ satisfying the conditions of Lemma 2.3.1 $D_s\varphi(W_t) = \nabla\varphi(W_t)\mathbb{1}_{s\leq t}(t)$. As a direct consequence, for Arithmetic and Geometric Brownian Motions introduced in subsection 1.2.1 we have that

$$D_sX_t^{\text{ABM}} = \sigma I_d\mathbb{1}_{s\leq t}(t), \tag{2.9}$$

$$D_sX_t^{\text{GBM}} = \sigma\,\text{diag}\,(X_t)\,\mathbb{1}_{s\leq t}(t), \tag{2.10}$$

by their analytical solutions given in Equation 1.12 and Equation 1.13 respectively. These expressions shall be of great use in chapter 7 as they allow one to use analytically accurate approximations for the Malliavin derivatives of certain forward diffusions. For an illustration on Malliavin differentiation, we refer to Figure 2.1 where the evolution of ABM and GBM are depicted in the one-dimensional case together with their corresponding Malliavin derivatives, for two realizations of the underlying Brownian motion.

As we shall later see in chapter 4, conditional expectations play a crucial role in the numerical analysis of FBSDE systems. Therefore, to conclude the section, let us finally state the following useful result which – similarly to Leibniz's rule – establishes the interchangeability of Malliavin differentiation and conditional expectations.

Proposition 2.3.1 (Malliavin Derivative of Conditional Expectations)
Let $F\in\mathbb{D}^{1,p}$. Then for all $0\leq s,t\leq T$ $\mathbb{E}\left[F|\mathcal{F}_t\right]\in\mathbb{D}^{1,p}$ and

$$D_s\mathbb{E}\left[F|\mathcal{F}_t\right] = \mathbb{E}\left[D_sF|\mathcal{F}_t\right]\mathbb{1}_{s\leq t}(t). \tag{2.11}$$

In particular, if $F\in\mathcal{F}_t$ we have $D_sF = 0$ for all $s\in(t,T]$.

Proof. See [17, Proposition 1.2.8]. □

2.4 SDE's Malliavin Derivative

Having introduced the concept of Malliavin differentiation, we can now discuss how this concept relates to the FBSDE system Equation 1.56 introduced in the previous chapter. We start by first explaining how the Malliavin derivatives of the solutions of SDEs look like.

In order to do this, let us define the flow and its inverse of an SDE, denoted by $\nabla_xX^{(\tau,x)}$ and $\nabla_xX^{-1^{(\tau,x)}}$ respectively, as the solutions of following linear SDEs

$$\nabla_xX_t^{(\tau,x)} := I_d + \int_\tau^t \nabla_x\mu\left(r, X_r^{(\tau,x)}\right)\nabla_xX_r^{(\tau,x)}\mathrm{d}r + \int_\tau^t\sum_{j=1}^d\nabla_x\sigma^j\left(r,X_r^{(\tau,x)}\right)\nabla_xX_r^{(\tau,x)}\mathrm{d}W_r^j \tag{2.12a}$$

$$\begin{aligned}\nabla_xX_t^{-1^{(\tau,x)}} := I_d + \int_\tau^t\nabla_xX_r^{-1^{(\tau,x)}}\left[\nabla_x\mu\left(r,X_r^{(\tau,x)}\right) - \sum_{j=1}^d\nabla_x\sigma^j\left(r,X_r^{(\tau,x)}\right)\nabla_x\sigma^j\left(r,X_r^{(\tau,x)}\right)\right]\mathrm{d}r\\ + \int_\tau^t\sum_{j=1}^d\nabla_xX_r^{-1^{(\tau,x)}}\nabla_x\sigma^j\left(r,X_r^{(\tau,x)}\right)\mathrm{d}W_r^j,\end{aligned} \tag{2.12b}$$

where $\nabla_x\mu:[0,T]\times\mathbb{R}^d\to\mathbb{R}^{d\times d}$, $\nabla_x\sigma^j:[0,T]\times\mathbb{R}^d\to\mathbb{R}^{d\times d}$ are deterministic functions corresponding to the spatial derivatives of the coefficients in the SDE in Equation 1.56a; σ^j denotes the j'th column of the diffusion coefficient, and $\mathrm{d}W_r^j$ the j'th coordinate of the d-dimensional Brownian increment. The following lemma provides a natural interpretation to the processes above, as the Jacobian matrix of the solution with respect to the initial condition x: $\nabla_x\left[X_t^{(\tau,x)}\right]_{ij} = \frac{\partial\left[X_t^{(\tau,x)}\right]_i}{\partial x_j}$.

Lemma 2.4.1 (Flow of SDEs)
Let the conditions of Assumption 1.2.1 be in power. Additionally assume that μ and σ are twice continuously differentiable, with bounded derivatives. Let $\nabla_xX^{(\tau,x)},\nabla_xX^{-1^{(\tau,x)}}$ be the processes defined in Equation 2.12. Then for every $p>1$ there exists a constant C such that

$$\mathbb{E}\left[\sup_r\left|\nabla_xX_r^{(\tau,x)}\right|^p\right] + \mathbb{E}\left[\sup_r\left|\nabla_xX_r^{-1^{(\tau,x)}}\right|^p\right] < C. \tag{2.13}$$

[2]See, e.g., [20] for detailed steps.

Furthermore, for all $0 \leq t, s \leq T$

$$\mathbb{E}\left[\left|\nabla_x X_r^{(\tau,x)} - \nabla_x X_s^{(\tau,x)}\right|^2 + \left|\nabla_x X_r^{-1^{(\tau,x)}} - \nabla_x X_s^{-1^{(\tau,x)}}\right|^2\right] \leq C|r-s|, \tag{2.14}$$

and $\nabla_x X_r^{(\tau,x)} \nabla_x X_r^{-1^{(\tau,x)}} = I_d$ *almost surely.*

Proof. See [21, Lemma 2.4]. □

The lemma above provides a direct way to represent the Malliavin derivative of the solution of an SDE, as stated by the following theorem.

Theorem 2.4.1 (Malliavin Derivative of SDEs)
Consider Equation 1.56a and take any $p \geq 1$. *Let the conditions of Assumption 1.2.1 be in power. Additionally assume that* μ *and* σ *are twice continuously differentiable, with bounded derivatives. Then for any* $0 \leq t \leq T$ $X_t \in \mathbb{D}^{d,p}$, *and its Malliavin derivative for each* $0 \leq s \leq T$ *admits to a version satisfying the following linear SDE*

$$D_s X_t = \sigma(s, X_s) + \int_s^t \nabla_x \mu(r, X_r) D_s X_r \mathrm{d}r + \int_s^t \nabla_x \sigma(r, X_r) D_s X_r \mathrm{d}W_r, \quad 0 \leq s \leq t \leq T, \tag{2.15}$$

with $D_s X_t = 0$ *when* $0 \leq t < s \leq T$. *Additionally, the version of the Malliavin derivative solving the SDE above satisfies the following representation formula*

$$D_s X_t = \nabla_x X_t^{(\tau,x)} \nabla_x X_s^{-1^{(\tau,x)}} \sigma\left(s, X_s^{(\tau,x)}\right) \mathbb{1}_{s\leq t}(t) \mathbb{1}_{\tau \leq s}(s) = \nabla_x X_t^{(s,x)} \sigma(s, X_s) \mathbb{1}_{s \leq t}(t) \tag{2.16}$$

almost surely. Consequently, there exists a constant C *such that for any* $0 \leq s \leq u < r \leq T$

$$\sup_s \mathbb{E}\left[\sup_r |D_s X_r|^2\right] \leq C, \tag{2.17}$$

$$\mathbb{E}\left[|D_s X_r - D_s X_u|^2\right] \leq C|r-u|. \tag{2.18}$$

Proof. See [17, Theorem 2.2.1, Lemma 2.2.2]. □

Let us now interpret this result. By Equation 2.15 we see that the Malliavin derivative of the solution of an SDE admits to a version which solves a linear SDE itself. Motivated by this, from now on, throughout the whole work we define the Malliavin derivative of the SDE as its version solving Equation 2.15. Additionally, it can also be seen that the Malliavin derivative $D_s X$ – for any $s \in [0, T]$ – is bounded in the supremum norm and is also continuous. These results shall be crucial in the error analysis presented later in chapter 6.

2.5 BSDE's Malliavin Derivative

The Malliavin differentiability of SDEs naturally raises a question whether similar propositions can be made about BSDEs and Malliavin calculus. Turns out that we can. Below we state the theorem which establishes that the solution pair of the BSDE part of an FBSDE is not just Malliavin differentiable, but their Malliavin derivatives solve a linear BSDE themselves. This property proves to be fundamental since, as we shall soon see, the Z-process can be expressed as $Z_t = D_t Y_t$. Therefore, one by solving the BSDE of the Malliavin derivative does not merely solve the natural dynamics of the control process, but also obtains certain regularity properties – and continuity in particular – provided by the Malliavin derivative.

We remark that the results below hold true even in the case of general, non-Markovian BSDE problems – see, e.g., [16], [22] or [23]. Nevertheless, since we are only concerned with the FBSDE system given in Equation 1.56, we state the original result of Pardoux and Peng proven in [2]. For this, we first have to make further assumptions about the FBSDE system, which are eventually all linked to the Malliavin differentiability of its solutions.

Assumption 2.5.1 (Malliavin Differentiability of FBSDEs)
Let the conditions of Assumption 1.5.1 be satisfied. Additionally, assume that

1. μ *and* σ *are twice continuously differentiable in* x *with bounded derivatives;*

2. g *is twice continuously differentiable with bounded derivatives;*

3. f is twice continuously differentiable in (x, y, z) with bounded derivatives – uniformly in time.

It is worth to highlight that comparing Assumption 1.5.1 to Assumption 2.5.1 we see that the additional assumptions are all smoothness criteria, which essentially ensure the Malliavin differentiability of the solution triple through the Malliavin chain rule Lemma 2.3.1 and the continuity of their Malliavin derivatives. We remark that for pure Malliavin differentiability of the solution pair it is sufficient to assume continuous differentiability for both g and f and refer to [23] for a more detailed discussion on the topic. Notwithstanding, in order to assure convergence of the upcoming numerical schemes in chapter 7, for the rest of the work we only consider the stronger set of conditions stated in Assumption 2.5.1. With this in mind, we can finally state the main result of this chapter.

Theorem 2.5.1 (Malliavin Differentiability of BSDEs)
Consider a BSDE in Equation 1.56b. Let the conditions of Assumption 2.5.1 be satisfied. Then for any $0 \leq t \leq T$: $(Y_t, Z_t) \in \mathbb{D}^{1,2} \times \mathbb{D}^{d,2}$, and their Malliavin derivatives for each $0 \leq s \leq T$ admit to a version solving the following linear BSDE

$$\begin{aligned} D_sY_t = \nabla_x g(X_T)D_sX_T + \int_t^T & [\nabla_x f(r, X_r, Y_r, Z_r)D_sX_r + \nabla_y f(r, X_r, Y_r, Z_r)D_sY_r \\ & + \nabla_z f(r, X_r, Y_r, Z_r)D_sZ_r]\, \mathrm{d}r - \int_t^T D_sZ_r \mathrm{d}W_r, \quad 0 \leq s \leq t \leq T, \end{aligned} \tag{2.19}$$

with $D_sY_t = 0, D_sZ_t = 0$ when $0 \leq t < s \leq T$. Furthermore, the above BSDE gives a version of the Malliavin derivative for which

$$Z_s = D_sY_s \tag{2.20}$$

almost surely.

Proof. The proof can be split in two parts. One first needs to prove that $Z_t \in \mathbb{D}^{d,2}$, which is the more difficult and technical part as it requires some additional properties on the Malliavin derivative of stochastic integrals. From there, by Assumption 2.5.1, it follows that $Y_t \in \mathbb{D}^{1,2}$. Using these results, the second part of the proof then simply follows by a Picard iteration argument similar to Corollary 1.4.1. We refer to [2, Proposition 2.2] for more details. □

Let us now interpret the results provided by Theorem 2.5.1. We see that, by the concept of solutions, $\{(D_sY_t, D_sZ_t)\}_{s,t\in[0,T]}$ solving Equation 2.19 satisfy

$$\sup_{0\leq s\leq T} \left[\mathbb{E}\left[\sup_{0\leq t\leq T} |D_sY_t|^2 \right] + \mathbb{E}\left[\int_s^T |D_sZ_t|^2 \mathrm{d}t \right] \right] < \infty. \tag{2.21}$$

From now on – motivated by Equation 2.20 – we define the Z-process in the solution of Equation 1.56b as the version of the Malliavin derivative D_tY_t satisfying Equation 2.19. Consequently – since D_sY is the solution of the linear SDE in Equation 2.20 and thus $\mathbb{E}\left[\sup_{t\in[0,T]} |D_sY_t|^2\right] < \infty$ –, we have that the control process is bounded under the supremum norm

$$\mathbb{E}\left[\sup_t |Z_t|^2 \right] < \infty. \tag{2.22}$$

Moreover, the theorem above also provides a unique way to establish the continuity of the Z-process. This was first proven by Pardoux and Peng in [2, Lemma 2.4] for the case of Markovian FBSDEs. It was later generalized by Hu et. al in [22] for the general non-Markovian case, where they proved Kolmogorov continuity of the Z-process under relaxed conditions. In order to keep the discussion as general as possible, hereby we state the latter result.

Theorem 2.5.2 (Continuity of the Z-process, Hu et. al [22])
Let the assumptions of Theorem 2.5.1 be satisfied. Then there exists a universal constant C – independent of s, t, r – such that the Malliavin derivative above satisfies for any $0 \leq s \leq t \leq r \leq T$

$$\mathbb{E}\left[|D_sY_r - D_sY_t|^2 \right] \leq C|r - t|. \tag{2.23}$$

Additionally, if we define $Z_t = D_tY_t$ as the version solving Equation 2.19, we also have

$$\mathbb{E}\left[|Z_t - Z_s|^2 \right] \leq C|t - s|. \tag{2.24}$$

Proof. See [22, Theorem 2.6, Part (b)]. □

Summarizing the results provided by Theorem 2.5.1 and Theorem 2.5.2 we therefore have that the unique solution pair of Equation 1.56b satisfies

$$\mathbb{E}\left[\sup_t |Y_t|^2\right] + \mathbb{E}\left[\sup_t |Z_t|^2\right] < \infty. \tag{2.25}$$

This is where the formulation given by Malliavin calculus shines in its full glory. It provides regularity and continuity for the control process under Assumption 2.5.1 and essentially enables us to bound errors of numerical schemes under the norm defined by the left-hand side of Equation 2.25 – as we shall later see in chapter 6.

To conclude the discussion on the Malliavin differentiability of BSDEs, let us finally introduce the concepts of variational processes corresponding to the solutions of BSDEs. We define $\{(\nabla_x Y_t^{(s,x)}, \nabla_x Z_t^{(s,x)})\}_{0\leq s\leq t\leq T}$ as the unique pair of random processes satisfying the following linear BSDE

$$\begin{aligned}\nabla_x Y_t^{(s,x)} &= \nabla_x g\left(X_T^{(s,x)}\right)\nabla_x X_T^{(s,x)} \\ &+ \int_t^T \left[\nabla_x f\left(r, X_r^{(s,x)}, Y_r^{(s,x)}, Z_r^{(s,x)}\right)\nabla_x X_r^{(s,x)} + \nabla_y f\left(r, X_r^{(s,x)}, Y_r^{(s,x)}, Z_r^{(s,x)}\right)\nabla_x Y_r^{(s,x)}\right. \\ &\left. + \nabla_z f\left(r, X_r^{(s,x)}, Y_r^{(s,x)}, Z^{(s,x)}\right)\nabla_x Z_r^{(s,x)}\right]\mathrm{d}r - \int_r^T \nabla_x Z_r^{(s,x)}\mathrm{d}W_r.\end{aligned} \tag{2.26}$$

It can be shown that under the conditions of Assumption 2.5.1 this problem is indeed well-posed. Just as in case of SDEs – see representation formula Equation 2.16 –, similar relationships hold in between the Malliavin derivatives of the solutions of the BSDE and their corresponding variational processes. These relations are described in the following lemma.

Lemma 2.5.1 (Variational Process and the Malliavin Derivative)
Let Assumption 2.5.1 be in power. Then for any $0 \leq s \leq t \leq T$ we have

$$D_s Y_t = \nabla_x Y_t^{(s,x)} \nabla_x X_s^{-1^{(s,x)}} \sigma\left(s, X_s^{(s,x)}\right). \tag{2.27}$$

Consequently, by Theorem 2.5.1,

$$Z_t^{(t,x)} = \nabla_x Y_t^{(t,x)} \sigma(s,x) \tag{2.28}$$

also holds. Moreover the unique solution pair $\left(\nabla_x Y^{(t,x)}, \nabla_x Z^{(t,x)}\right)$ to Equation 2.26 are the gradients of the solutions of the corresponding Markovian BSDE in Equation 1.61 with respect to the initial condition x. Finally, the variational process is bounded, i.e. there exists a constant C such that $\forall (t,x) \in [0,T]\times\mathbb{R}^d$

$$\left|\nabla_x u(r, X_r^{(t,x)})\right| = \left|\nabla_x Y_r^{(t,x)}\right| \leq C. \tag{2.29}$$

Proof. For the representation formulas and the connection with the Malliavin derivative see [2, Lemma 2.4-2.5, Corollary 2.11]. For the boundedness of the variational process see [24, Corollary 3.2]. □

These results shall be crucial with respect to the error analysis provided in chapter 6 – see Lemma 6.2.2 in particular.

2.6 A Few Words on Alternative Representations

In order to conclude the chapter, let us finally have a few words on alternative representations of the Z-process given by the Malliavin derivative. In fact, as it can be seen, Theorem 2.5.1 is not directly applicable as one needs to gather estimations for the Malliavin derivatives $(D_s Y, D_s Z)$ at each point in time. To overcome this burden and avoid solving the Malliavin BSDE directly, there have been several representation formulas proven in the literature, providing a *"Feynman–Kac like"* formula for the continuous version of the Z-process. These representations allow one to circumvent solving the Malliavin BSDE directly and yet gather continuous approximations for the control. Hereby we state two of them without proofs.

The first representation exploits the fact that Equation 2.19 in Theorem 2.5.1 is a linear BSDE. Therefore tailoring Proposition 1.4.1 to the conditions of the Malliavin BSDE, one can show that the following holds.

Proposition 2.6.1 (Representation Formula 1 – Malliavin Derivative)
Under the conditions of Assumption 2.5.1, the Malliavin derivative solving Equation 2.19 can be represented as

$$D_s Y_t = \mathbb{E}\left[\rho_{s,T} D_s Y_T + \int_s^T \rho_{s,r} \nabla_x f(r, X_r, Y_r, Z_r) D_s X_r \mathrm{d}r \middle| \mathcal{F}_t\right], \tag{2.30}$$

where the coefficients $\rho_{s,r}$ solve the following forward SDE

$$\rho_{s,r} \ := \ \exp\left(\int_s^r \nabla_z f(\tau, X_\tau, Y_\tau, Z_\tau)\mathrm{d}W_\tau + \int_s^r \left[\nabla_y f(\tau, X_\tau, Y_\tau, Z_\tau) - \frac{1}{2}(\nabla_z f(\tau, X_\tau, Y_\tau, Z_\tau))^2\right]\mathrm{d}\tau\right). \tag{2.31}$$

In particular, we have

$$Z_s = \mathbb{E}\left[\rho_{s,T} D_s Y_T + \int_s^T \rho_{s,r} \nabla_x f(r, X_r, Y_r, Z_r) D_s X_r \mathrm{d}r \middle| \mathcal{F}_s\right]. \tag{2.32}$$

$$\tag{2.33}$$

Proof. See, e.g., [16, Proposition 5.5] for the general – not necessarily Markovian – case. □

The other theorem provides a similar interpretation for the control process by representing it with another conditional expectation. We state the theorem with the notation of [21].

Proposition 2.6.2 (Representation Formula 2 – Malliavin Derivative)
Under the conditions of Assumption 2.5.1, the continuous version of the Z-process given by the Malliavin derivative in Equation 2.19 can be represented as

$$Z_s = \mathbb{E}\left[g(X_T) H_{s,T} + \int_t^T f(r, X_r, Y_r, Z_r) H_{s,r} \mathrm{d}r \middle| \mathcal{F}_s\right], \tag{2.34}$$

where the so-called Malliavin weights $H_{s,r}$ solve the following forward SDE

$$H_{s,r} := \frac{1}{r-s}\int_s^r \sigma^{-1}(\tau, X_\tau) D_s X_\tau \mathrm{d}W_\tau. \tag{2.35}$$

Proof. See [24, Theorem 3.1]. □

The message to take away both from Proposition 2.6.1 and Proposition 2.6.2 is that one can solve the control problem of the BSDE by solving the conditional expectations arising on the right-hand sides of Equation 2.30 and Equation 2.34. Notice that the arguments of these conditional expectations do not depend on $(D_s Y, D_s Z)$ anymore, only on the Malliavin derivative of the forward process. However, it is worth to mention that – through the coefficients $\rho_{s,r}$ and $H_{s,r}$ defined in Equation 2.31 and Equation 2.35 – one needs to solve another forward SDE which may pose additional challenges in numerical methods.

Chapter 3

A Few Words on Deep Learning

Machine learning has shown remarkable successes in a wide range of engineering problems all the way from image recognition and language processing to solving partial differential equations – see, e.g., [25], [26] or [27] – and, through the Feynman–Kac relations presented in Theorem 1.2.2, also BSDEs – see, e.g., [7]. Recently, the latter field has been an area of intense research, primarily because of deep neural networks showing excellent empirical capability in solving high-dimensional problems, and thus inciting hope that they could be used to overcome the *curse of dimensionality*. Motivated by these results, we decided to use deep learning for the approximation tool of this work. In what follows, we explain the basic concepts of neural network modeling focused on notions which are most relevant for the algorithms proposed later in chapter 5. We start the chapter by describing what neural networks are, after which we motivate their use with the so-called Universal Approximation Theorems. We remark that the fact that neural networks can approximate a function and also its derivatives arbitrarily well shall be a crucial input to get meaningful estimations of the Malliavin derivatives through the Malliavin chain rule in chapter 5. Thereupon, we explain the most popular optimization methods for tuning the parameter set of a neural network. Finally, as differentiability of such function approximators is crucial in the latter algorithms proposed in chapter 5, we briefly touch upon automatic differentiation, which provides an efficient and cheap way to estimate derivatives semi-analytically.

Since machine learning, in the context of this work, is just a numerical tool for obtaining a wide enough function class on which we can perform Monte Carlo regression, we keep the discussion as concise as possible and refer the more interested reader to one of the excellent, detailed introductions to the field [28], [29] or [30].

3.1 Deep Neural Networks

For our purpose *fully-connected feedforward deep neural networks* are simply a sequence of compositions of simple functions, which therefore can be collected in the following form

$$\mathcal{NN}\left(\mathbf{x}|\Theta\right) := a^{\text{out}} \circ A^{L+1}(\,|\theta^{L+1}) \circ a \circ A^{L}(\,|\theta^{L}) \circ a \cdots \circ a \circ A^{1}(\mathbf{x}|\theta^{1}), \tag{3.1}$$

with $\Theta := \left(\theta^1, \dots, \theta^{L+1}\right) \in \mathbb{R}^p$, where p is the number of parameters in the model. In Equation 3.1, a and a^{out} are (non-)linear *activations* applied elementwise on an input, and $A^\ell(\mathbf{y}|\theta^\ell), \ell \in \{1, \dots, L+1\}$ is an affine mapping $A^\ell(\mathbf{y}|\theta^\ell) : \mathbb{R}^{N^{\ell-1}} \to \mathbb{R}^{N^\ell}$, defined by

$$A^\ell\left(\mathbf{y}|\theta^\ell\right) = \left\{\mathbf{W}^\ell, \mathbf{b}^\ell\right\} := \mathbf{W}^\ell \mathbf{y} + \mathbf{b}^\ell, \tag{3.2}$$

where $\mathbf{W}^\ell$ is an $\mathbb{R}^{N^\ell \times N^{\ell-1}}$ matrix containing the so-called *weights* of the layer. Moreover, $\mathbf{b}^\ell$ is an N^ℓ-dimensional real vector called the *biases*. The affine transformations $A^\ell(\cdot|\theta^\ell), \ell = 1 \dots L$ are referred to as the *hidden layers* of the neural network model, whereas the last transformation is called the *output layer* of the network. The size of a given layer N^ℓ denotes how many computing units, so-called *artificial neurons* are contained in that layer.

The power of deep learning is given by the fact that through such a deep sequence of simple composition functions, highly complex patterns can be approximated with good accuracy. For these approximations to be able to extract non-linear patterns, it is crucial for the above activation function to be non-linear as well, otherwise the model would simply be a linear transformation of the input. Common choices for activations include the ReLU (Rectified Linear Unit), the ELU (Exponential Linear Unit) and

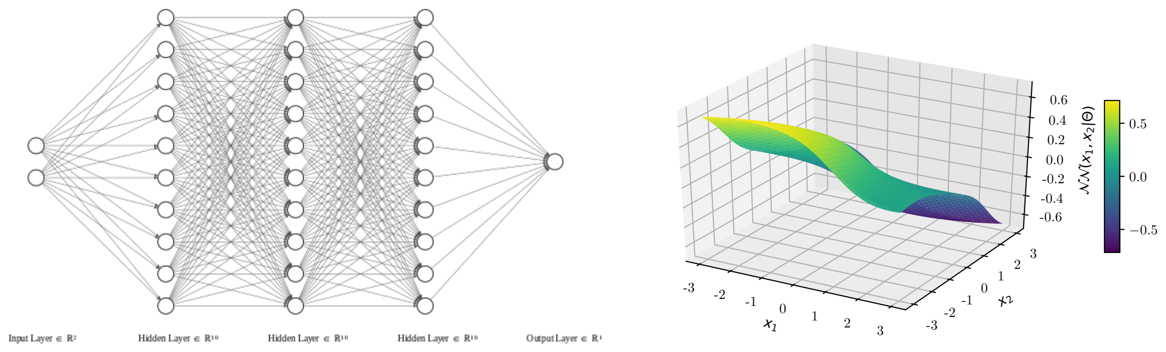


Figure 3.1: Deep Neural Networks Illustration. Left: network architecture[1], right: the corresponding mapping with randomly initialized parameter set.

the tanh hyperbolic tangent functions, which are defined by the mappings below ($x \in \mathbb{R}$)

$$a^{\text{ReLU}}(x) = \max\left[x, 0\right], \tag{3.3}$$

$$a^{\text{ELU}}(x) = \begin{cases} x, & x \geq 0, \\ \alpha\left(e^x - 1\right), & x < 0, \end{cases} \tag{3.4}$$

$$a^{\tanh}(x) = \tanh(x), \tag{3.5}$$

where $\alpha > 0$. Additionally, we remark that in all our forthcoming applications the output activation a^{out} is chosen as the identity function. In what follows, these activation functions play the central role. Nevertheless, it is worth to note that the choice of the activation function is an inherent part of neural network modeling, which itself has a fundamental impact on the properties of the function approximation. The most important of these properties is differentiability. As it can easily be seen from above, a neural network is itself differentiable, if and only if the applied non-linear activation function is differentiable.

3.2 Universal Approximation Theorem

The use of Deep Learning is most often motivated by the so called Universal Approximation Theorem (UAT). This (set of) theorem(s) establish(es) that the class of fully-connected feedforward neural networks introduced above are indeed capable of fitting any real valued continuous function defined on compact subsets of $\mathbb{R}^d$. The intuitive message one should take away from these theorems is that neural networks themselves, assuming the existence of a procedure for finding appropriate network parameters, span a wide function class. Although, these theorems are very often cited and held in front of machine learning research as a shield, it is also important to highlight that none of the current versions of the theorem is constructive: they do not provide, suggest, imply a procedure for optimizing the parameter space Θ.

The proposition has several – gradually relaxed – versions, hereby we only state two of the original formulations. In the theorems below, we denote the d-dimensional hypercube by $I_d \coloneqq [0,1]^d$ and the space of real-valued continuous functions defined over it by $C(I_d)$. Furthermore, we put $G : \mathbb{R}^d \to \mathbb{R}$ for the finite sums of the form

$$G(x) = \sum_{n=1}^{N} w_n a\left(y_n x + b_n\right), \tag{3.6}$$

where $b_n, w_n \in \mathbb{R}, x, y_n \in \mathbb{R}^d$. With these notations in hand, we are now able to state the two classical UAT theorems. The first, given by Cybenko in [31], proves the approximation capability for a class of so-called *sigmoid activations.*

Theorem 3.2.1 (Universal Approximation Theorem, Cybenko [31])
Consider the class of sigmoidal activation functions, i.e. which satisfy $a : \mathbb{R} \to \mathbb{R}, \lim_{x\to+\infty} a(x) = 1, \lim_{x\to-\infty} a(x) = 0$. *Let* a *be any such, sigmoidal function that is also continuous. Then the finite sums* G *of the form Equation 3.6 are dense in* $C(I_d)$. *In other words, given any* $f \in C(I_d)$ *and* $\varepsilon > 0$ *there is a sum* $G(x)$ *of the above form, for which*

$$|G(x) - f(x)| < \varepsilon, \quad \forall x \in I_d. \tag{3.7}$$

[1]Figure drawn with NN-SVG.

Proof. See [31, Theorem 2]. □

Comparing the form of these finite sums against the definition of fully-connected feedforward neural networks presented in Equation 3.1, it is easy to recognize that w_n can be thought of as the n'th column vector of the output's weight matrix; and y_n as the n'th row-vector in the weight matrix of the first (and only) hidden layer, which contains N-many neurons. It is crucial to notice, however, that the above theorem merely provides the universal approximation capability in the infinite limit without giving any bounds on the size of N.

The theorem has been extended multiple times. For instance Hornik et al. in [32] generalized it to non-sigmoid activations, and in a follow up paper in [33] showed that the networks are not just merely able to approximate a function but also its derivatives with arbitrary accuracy. Since this observation is crucial with respect to the algorithms proposed in chapter 5, we hereby state the theorem in its original form. However, in order to do this we first have to introduce the concepts of *Sobolev spaces* and *ℓ-finite* activations. This is done in the next two definitions.

Definition 3.2.1 (Sobolev Spaces)**.** Let us denote the L^p-norm restricted on a subset $U \in \mathbb{R}^d$ by $\|f\|_{p,U} := \left(\int_U |f(x)|^p \mathrm{d}x\right)^{1/p}$. Then the space of functions $f \in C^m(U)$ for which

$$\|f\|_{m,p,U} := \left(\sum_{|\alpha|\le m} \|D^\alpha f\|_{p,U}^p\right)^{1/p} < \infty \tag{3.8}$$

is called the $S_p^m(U)$ Sobolev space.[2] Respectively, the norm on the left-hand side above is called the Sobolev norm, and its natural metric is denoted by $d_p^m(f,g) := \|f-g\|_{m,p,U}$ for each $f, g \in S_p^m(U)$.

Definition 3.2.2 (ℓ-finite Activations)**.** Let $\ell \in \mathbb{N}$ be given. An activation function $a : \mathbb{R} \to \mathbb{R}$ is ℓ-finite if $a \in C^\ell(\mathbb{R})$ and $0 < \int_{\mathbb{R}} \left|D^\ell a(x)\right| \mathrm{d}x < \infty$.

With these two concepts in hand, we can now the state the universal approximation theorem in Sobolev spaces proven by Hornik et al. in [33].

Theorem 3.2.2 (Universal Approximation Theorem in Sobolev Spaces, Hornik et al. [33])
Let a be an ℓ-finite activation and U a compact subset of $\mathbb{R}^d$. Then for all $0 \le m \le \ell$ the finite sums of the form Equation 3.6 are d_p^m-dense in $S_p^m(U)$.

In particular, for any $\ell = 1$-finite activation we have that there exists a finite sum G such that for any $f \in S_2^m(U)$[3] and $\varepsilon > 0$

$$\int_U |f(x) - G(x)|^2 \mathrm{d}x + \int_U |\nabla_x f(x) - \nabla_x G(x)|^2 \mathrm{d}x < \varepsilon. \tag{3.9}$$

Proof. See [33, Corollary 6]. □

There is one important corollary to notice: these theorems stand for shallow, single hidden layer neural networks, however, they naturally generalize to networks of arbitrary depth – i.e. deep neural networks. In order to see this, apply the theorems above for the second hidden layer of the network to approximate any f. Subsequently apply them again to further hidden layers approximating the identity function. The output of such a network is f itself.

To conclude the discussion on the UAT property, let us finally remark that out of the activation functions introduced before, the tanh in Equation 3.5 satisfies both the conditions of Theorem 3.2.1 and Theorem 3.2.2 with $\ell = 1$ – on top of being smooth – and therefore is a good candidate for our latter applications. Although, this cannot be said about neither the ReLU in Equation 3.3 nor the ELU in Equation 3.4, it can be shown that they themselves also admit to the UAT property under weaker assumptions. In fact, Pinkus in [34] showed that the UAT property is equivalent to non-polynomial activations. Additionally, in a recent paper, Kidger and Lyons in [35] improved the conditions for the deep neural network version and further relaxed the bounds for the widths of deep neural networks still exhibiting universal approximation properties.

[2]In the definition above α is a multi-index and D^α denotes the multivariate differentiation operator.

[3]It is worth to notice the similarities between the norm below and the one of $\mathbb{D}^{1,2}$ introduced in the previous chapter – see Equation 2.6. In fact, $\|\cdot\|_{1,p}$ is often called the *Malliavin–Sobolev norm.*

3.3 Training Neural Networks

We have seen that, once the network architecture is defined, what determines the mapping of a certain input to an output are the parameters incorporated in the neural network model, i.e. each layer's weights and biases. For a neural network to approximate a certain function, these parameters need to be *optimized*, which – in the context of machine learning – is called the *training* of the network.

3.3.1 The Loss Function

In order to measure a neural network's performance, an abstract distance from the desired behaviour must be formulated which expresses how well is the network describing the target behaviour. This abstract measure is called the *loss function.* Loss functions serve as the objective functions to be minimized during the so-called *training* procedure, in which the network's optimal set of parameters – i.e. with which the network is generating outputs closest to the desired outputs – are sought. The loss function is inherently linked to the specific estimation problem and the properties of the target function. Nonetheless, in the scope of this work the most important problem setup is the so-called *regression* problem which we define in the following.

Regression Problem

In a regression problem we aim to estimate a deterministic, vector valued function, mapping a d-dimensional input to a $q = N^{L+1}$-dimensional output. For this, we gather training data sampled from some (unknown) multivariate joint distribution $(\mathcal{X}, \mathcal{Y}) \sim \mathcal{D}_{X,Y}$. Here $\mathcal{X}$ is most often referred to as the *input-*, and $\mathcal{Y}$ as the *label space* of the problem. The goal in a regression setting is then to approximate the conditional expectation $h(\mathbf{x}) = \mathbb{E}\left[Y|X = \mathbf{x}\right]$ for $(X, Y) \sim \mathcal{D}_{X,Y}$. This deterministic function $h(\mathbf{x})$ is called the *labeling function.*

The purpose of the loss function is to measure how well the current approximation is adhering to the desired output of the network. A common choice for this abstract distance is the *mean squared error*, which measures the $\mathbb{L}^2$-distance between predictions and true labels

$$\mathcal{L}(\Theta) := \mathbb{E}_{(X,Y)\sim\mathcal{D}_{X,Y}}\left[|\mathcal{NN}\left(X|\Theta\right) - Y|^2\right]. \tag{3.10}$$

In practice, one is often restricted to finite samples drawn from the true distribution $\mathcal{D}_{X,Y}$. In this case the above loss function is approximated by Monte Carlo integration, and for a sample $\{(\mathbf{x}_i, \mathbf{y}_i)\}_{i=1}^{M}$ the following empirical loss function is deployed instead

$$\mathcal{L}^{\text{emp}}\left(\Theta|\{(\mathbf{x}_i, \mathbf{y}_i)\}_{i=1}^{M}\right) := \frac{1}{M}\sum_{i=1}^{M} |\mathcal{NN}\left(\mathbf{x}_i|\Theta\right) - \mathbf{y}_i|^2. \tag{3.11}$$

Assuming that the empirical training sample is representative for the true distribution $\mathcal{D}_{X,Y}$ and that the empirical loss mean is a good estimator for the true loss measure, one – by minimizing the empirical loss function – obtains a function that exhibits *close* behaviour to the desired outputs $\mathbf{y}_i$. Hence, the ultimate goal of the optimization problem is to find the set of parameters for which the empirical loss measure is the smallest

$$\widehat{\Theta} \in \arg\min_{\Theta}\left[\mathcal{L}^{\text{emp}}\left(\Theta|\{(\mathbf{x}_i, \mathbf{y}_i)\}_{i=1}^{M}\right)\right], \tag{3.12}$$

given a finite training sample.

3.3.2 Stochastic Gradient Descent and Beyond

Having introduced loss functions and the intuition behind them, we now briefly cover how the *optimal* parameter set above is approximated. In order to minimize the loss function with respect to the parameters of the network, the most commonly used optimization technique in deep learning is Stochastic Gradient Descent (SGD) – and its offsprings. SGD is a first-order gradient based optimization method built upon the idea that a scalar function – just like the loss function above – while varying its inputs infinitesimally, can be decreased the most in the opposite direction to its gradient. Note that for such a method to be applicable, it is crucial that the networks themselves are differentiable mappings – *at least almost everywhere.*

The term *"stochastic"* comes from the fact that SGD is not performed on the whole training sample, but rather on a randomly drawn subsample of it, on which the estimation of the gradient of the loss function is obtained. Therefore – in case of a mean squared error loss –, the SGD step of the i'th iteration is of the form

$$\begin{aligned} \Theta^{(i+1)} &= \Theta^{(i)} - \eta \nabla_\Theta \widehat{\mathcal{L}}(\Theta^{(i)}), \\ \widehat{\mathcal{L}}\left(\Theta^{(i)}\right) &= \frac{1}{K}\sum_{k=1}^{K} \left\| \mathcal{NN}\left(\mathbf{x}_k | \Theta^{(i)}\right) - \mathbf{y}_k \right\|_2^2, \end{aligned} \tag{3.13}$$

where we loop over K-long chunks of a random bijection $\varphi : \{1, \dots, M\} \to \{1, \dots, M\}$ that is chosen uniformly for each loop. We call such chunks *mini-batches* and their loop over the bijection φ an *epoch* of the training. It is important to notice that the performance of this optimization algorithm inherently depends on the choice of $\eta \in (0, \infty)$, which is called the *learning rate* of the optimizer, another hyperparameter to be selected with care in a neural network estimation. We collect the detailed steps of an SGD optimization in algorithm 1.

Algorithm 1: Stochastic Gradient Descent

```
**Input:** $\eta$ – learning rate
**Input:** $\mathcal{L}(\cdot)$ – stochastic objective function to be minimized
**Input:** $\Theta^{(0)}$ – initial parameter vector
$i \leftarrow 0$ – initialize step counts
**while** $\Theta^{(i)}$ *not converged* **do**
    $i \leftarrow i + 1$ – update step count
    $\Theta^{(i)} \leftarrow \Theta^{(i-1)} - \eta \nabla_\Theta \mathcal{L}_i\left(\Theta^{(i-1)}\right)$ – update parameters
**end**
**Output:** $\Theta^{(i)}$ – final parameter estimation
```

Adam: An Extension to SGD

Adam is a descendant of SGD, which on top of estimating the first-order derivatives of the loss function, also calculates the moving averages of these derivatives in order to account for the variance stemming from noisy losses. Adam is one of the most frequently used optimization algorithms for training deep neural networks, and is the core optimization method used throughout chapter 7 in this work. Therefore, below we collect its explicit update rules in algorithm 2 and refer the more interested reader to the original paper in which it was proposed by Kingma and Ba [36] for more details. (We remark that in the formulation of algorithm 2 we use the standard notations $\odot, \oslash$ for element-wise multiplications and divisions respectively.) It is important to notice that on top of the usual learning rate Adam has additional hyperparameters to be chosen: the exponential decay rates and the numerical stability parameter. As suggested by the original paper [36], a good common choice for these latter parameters in practice are $\beta_1 = 0.9, \beta_2 = 0.999, \epsilon = 10^{-8}$.

3.4 Some Words on Specific Machine Learning Practices

The above architecture presented in Equation 3.1 has numerous extensions, most of which fall out of the scope of this work. Nonetheless, there are two important generalizations which we shall not leave unmentioned. In the following we briefly explain these concepts.

The first problem has to do with overfitting, a common issue in neural network modeling. As we have seen, a neural network model is a differentiable function approximator which is parametrized by a large parameter vector $\Theta := \left(\theta^1, \dots, \theta^{L+1}\right) \in \mathbb{R}^p$. One of the biggest challenges with respect to finding the optimal parameter set of a neural network is to avoid *"picking up on"* training data specific patterns, and rather enforce better generalization of the fitted models. Hence, just like in regular regression models, regularization approaches have been developed for neural networks, penalizing certain norms of the parameter vector Θ. However, as we shall later see in chapter 5, in our applications the underlying data is neither scarce nor noisy, and consequently overfitting poses a nearly insignificant and easily manageable problem[4] for the latter algorithms. Therefore, in the upcoming models regularization is not needed and consequently – instead of going into specifics – we refer the more interested reader to [28] for an introduction on such techniques.

[4]Quite simply by simulating more data.

Algorithm 2: Adam

```
Input: $\eta$ – learning rate
Input: $\beta_1, \beta_2$ – exponential decay rates for moment estimates
Input: $\epsilon$ – numerical stability parameter
Input: $\mathcal{L}(\cdot)$ – stochastic objective function to be minimized
Input: $\Theta^{(0)}$ – initial parameter vector
$m^{(0)} \leftarrow 0$ – initialize first-order moments
$v^{(0)} \leftarrow 0$ – initialize second-order moments
$i \leftarrow 0$ – initialize step counts
while $\Theta^{(i)}$ not converged do
    $i \leftarrow i + 1$ – update step count
    $g^{(i)} \leftarrow \nabla_\Theta \mathcal{L}^{(i)}(\Theta^{(i-1)})$ – collect gradients at time step $i$
    $m^{(i)} \leftarrow \beta_1 m^{(i-1)} + (1-\beta_1) g^{(i)}$ – update biased first-order moment estimates
    $v^{(i)} \leftarrow \beta_2 v^{(i-1)} + (1-\beta_2) g^{(i)} \odot g^{(i)}$ – update biased second-order moment estimates
    $\widehat{m}^{(i)} \leftarrow \frac{m^{(i)}}{1-\beta_1^i}$ – compute bias-corrected first-order moment estimates
    $\widehat{v}^{(i)} \leftarrow \frac{v^{(i)}}{1-\beta_2^i}$ – compute bias-corrected second-order moment estimates
    $\Theta^{(i)} \leftarrow \Theta^{(i-1)} - \eta \widehat{m}^{(i)} \oslash \left(\sqrt{\widehat{v}^{(i)}} + \epsilon\right)$ – update parameters
end
Output: $\Theta^{(i)}$ – final parameter estimation
```

3.4.1 Batch Normalization

Nevertheless, in the forthcoming algorithms we use another machine learning concept corresponding to the normalization of network inputs, which we thus cover briefly in this section. The technique is called *batch normalization* and it was first introduced in [37].

Batch normalization is a procedure in which the mean and the variance of each processed *mini-batch* – recall subsection 3.3.2 – are also computed during the training procedure. It is meant to reduce internal covariate shifts between the hidden layers, i.e. ”the change in the distribution of network activations due to the change in network parameters during training” [37, Pg. 2]. Batch normalization makes up for this phenomenon by incorporating normalization of the inputs within the network architecture. In fact, batch normalization extends the parameter space of a network by so-called *scales* and *offsets*. For the exact details of a batch normalization step, we refer to algorithm 3.

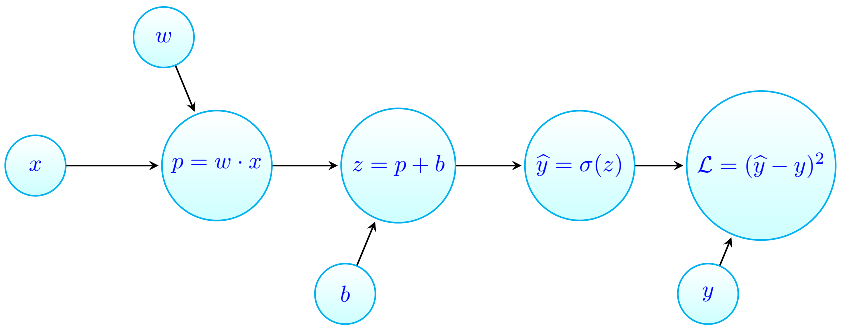
Having introduced batch normalization, we are now able to formulate the final neural network architecture used later in chapter 5. Hereby we extend the previous structure by inserting batch normalization before each hidden layer in Equation 3.1

$$\mathcal{NN}(\mathbf{x}|\Theta) := a^{\text{out}} \circ A^{L+1}(\,|\theta^{L+1}) \circ a \circ BN^L(\,|\theta^L) \circ A^L(\,|\theta^L) \circ a \cdots \circ a \circ A^1(\,|\theta^1) \circ BN^1(\mathbf{x}|\theta^1). \quad (3.14)$$

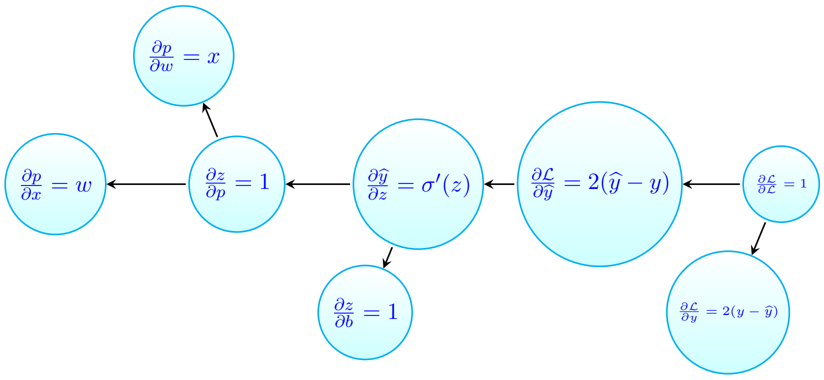
Notice that in above the scale and offset parameters of each batch normalization layer have been included in the parameter set of the corresponding hidden layer $\gamma^\ell, \beta^\ell \in \theta^\ell, \ell = 1, \dots, L$.

Algorithm 3: Batch-Normalization Transformation

```
Input: $\boldsymbol{\gamma}, \boldsymbol{\beta} \in \mathbb{R}^d$ – scale and offset parameters
Input: $\{\mathbf{x}\}_{i=1}^B$ – values of $\mathbf{x}$ over a mini-batch of size $B$
Result: $\{\mathbf{y}\}_{i=1}^B$ – batch-normalized input
$\boldsymbol{\mu}_B \leftarrow \frac{1}{M}\sum_{i=1}^M \mathbf{x}_i$ – collect batch mean
$\boldsymbol{\sigma}_B^2 \leftarrow \frac{1}{M}\sum_{i=1}^M (\mathbf{x}_i - \boldsymbol{\mu}_B) \odot (\mathbf{x}_i - \boldsymbol{\mu}_B)$ – collect batch variance
$\widehat{\mathbf{x}}_i \leftarrow (\mathbf{x}_i - \boldsymbol{\mu}_B) \oslash \left(\sqrt{\boldsymbol{\sigma}_B^2 + \varepsilon}\right)$ – normalization
$\mathbf{y}_i \leftarrow \boldsymbol{\gamma} \odot \widehat{\mathbf{x}}_i + \boldsymbol{\beta}$ – scale and shift normalize inputs
```


w
x
$p = w \cdot x$
$z = p + b$
$\widehat{y} = \sigma(z)$
$\mathcal{L} = (\widehat{y} - y)^2$
b
y

(a) Forward Propagation


$\frac{\partial p}{\partial w} = x$
$\frac{\partial p}{\partial x} = w$
$\frac{\partial z}{\partial p} = 1$
$\frac{\partial \widehat{y}}{\partial z} = \sigma'(z)$
$\frac{\partial \mathcal{L}}{\partial \widehat{y}} = 2(\widehat{y} - y)$
$\frac{\partial \mathcal{L}}{\partial \mathcal{L}} = 1$
$\frac{\partial z}{\partial b} = 1$
$\frac{\partial \mathcal{L}}{\partial y} = 2(y - \widehat{y})$

(b) Backpropagation

Figure 3.2: Automatic Differentiation Illustration

3.5 Automatic Differentiation

In order to conclude this high-level introduction to deep learning, we have to say a few words about how these concepts are implemented in practice. As we have seen above, the most commonly used optimization methods for tuning the parameter set of a neural network model rely on the ability to estimate the loss function's gradients with respect to the parameters in the model. However, we have also seen that the function approximator of a neural network depends on the parameters $\Theta := \left(\theta^1, \dots, \theta^{L+1}\right) \in \mathbb{R}^p$, where p is not seldom in the range of billions. The fact that gradient vectors of such size can be computed is not trivial. In fact, numerical differentiation – through, e.g., finite differences – of the loss function would not be feasible due to the large number of parameters. However, nor would symbolic differentiation be, because the hierarchical nature of the networks lead to huge and highly redundant symbolic expressions which would not fit in memory. In the following section we briefly explain how this computational burden is overcome, and introduce the concept of automatic differentiation, a half-numerical, half-symbolic differentiation method enabling us to calculate derivatives by a sequence of simple tensor multiplications. We refer the more interested reader to the survey paper [38] for more details.

3.5.1 Forward Propagation

In the context of neural networks, forward propagation is the hierarchical sequence of operations performed in Equation 3.14, ordered from right to left. It takes an input of the network and maps it to the output. In machine learning, such sequences of operations are executed by so-called *computational graphs* which are abstract representations of operations. Mathematically speaking, a computational graph is a directed, acyclic graph where each node corresponds to an operation and data flows through the vertices. In this way every node in the graph defines a function of multiple variables. The values of the variables are multidimensional arrays, i.e. *tensors*. Operations can be specified by three characteristics: a compute function which calculates the function's value with respect to its inputs, the set of input (or parent) nodes and the set of consumer nodes, i.e. nodes who take the node's value as an input. The key advantage of computational graphs is that with their usage one can break down rather complex calculations to a sequence of basic elementary operations that can easily be handled analytically.

For an example of a computational graph consider the one given in Figure 3.2a. This graph specifies a shallow, single hidden layer, single neuron neural network and calculates its mean squared error loss compared to a given set of training examples. This example is meant to demonstrate that every neural network can be represented by computational graphs which turns out to be a crucial observation with respect to the implementability of neural networks.

3.5.2 Backpropagation

With the use of computational graphs, we can now explain why differentiation of the loss function in the previous sections can be implemented in practice. The idea of automatic differentiation (or – in the context of neural networks – *backpropagation*) is the following: on top of representing the steps of forward propagation by a composition of simple, elementary operations – such as: additions, multiplications, exponentials, etc. –, simultaneously also store the derivatives of such elementary operations, which due to their simple structure can be computed analytically. Hence, besides building the original computational graph we also construct another graph in the background, which we call the *reverse graph*, where each node's operation is replaced with the derivatives of that node with respect to its parent nodes. Now, using the chain rule for any input value of the computational graph – having first forward propagated that input value through the original graph – we can roll back the output values through these expressions in the reverse graph and (in each node) get the derivative of the output with respect to that particular node. A demonstration of this idea is presented in Figure 3.2b. Here the reverse logic of the differentiating graph becomes apparent. Since the derivative of the loss function is known analytically (and so is the derivative of the activation function) after having forward propagated a certain input through the original graph, backpropagating the returned values through the differentiating graph yields the derivatives of the output (here $\mathcal{L}$) with respect to *every single* input. For instance, as it can be easily checked, the derivative of the loss with respect to the weight of the network in Figure 3.2a is given by

$$\frac{\partial \mathcal{L}}{\partial w} = \frac{\partial \mathcal{L}}{\partial \mathcal{L}} \frac{\partial \mathcal{L}}{\partial \widehat{y}} \frac{\partial \widehat{y}}{\partial z} \frac{\partial z}{\partial p} \frac{\partial p}{\partial w} = (\widehat{y} - y)\sigma'(z)x, \tag{3.15}$$

leading to a simple tensor multiplication in the reverse graph. These operations, with the help of modern matrix multiplication libraries, can be efficiently computed in parallel for a huge network, making the calculation of the gradient vector feasible in practice.

Chapter 4

An Overview of Existing Numerical Methods

If interested in the solutions of FBSDEs of the form Equation 1.56, one often has to rely on numerical methods to tackle them. In the following chapter we explain the most popular numerical methods in the literature, focusing on encouraging recent developments built on deep learning. We start off by describing a discretization procedure of the continuous problem and recall the well-known Euler–Maruyama method for forward SDEs. Thereafter, we turn to the backward equation and motivate a similar discrete approximation scheme for BSDEs. We emphasize that due to the adaptivity requirements, one cannot just perform backward Euler stepping over the reversed time horizon, and in fact is forced to approximate recursive conditional expectations instead. Consequently, we introduce two such discrete approximation schemes for BSDEs, the Euler- and the theta-scheme which the forthcoming algorithms are built upon. Subsequently, we briefly explain the idea of Least-Squares Monte Carlo (LSMC) regression, which is the key numerical method in this work to tackle conditional expectations. In the last section we turn to the class of recently proposed deep learning based algorithms to solve the FBSDE system. Since this has been an area of intensive research in the last couple of years resulting in numerous different approaches, we only describe two base algorithms which we call Forward and Backward Deep BSDE solvers respectively. For a broader overview of such deep learning based methods we refer the interested reader to the survey papers [39] and [40]. Finally, at the end of the section we touch upon the drawbacks of these methodologies in order to motivate our proposed algorithms explained in the next chapter.

4.1 Preliminaries

Throughout the whole chapter we are dealing with the decoupled system of FBSDEs given in Equation 1.56 which, for the convenience of the reader, we repeat here once again

$$X_t = x_0 + \int_0^t \mu(s, X_s)\mathrm{d}W_s + \int_0^t \sigma(s, X_s)\mathrm{d}W_s, \qquad 0 \le t \le T, \tag{4.1a}$$

$$Y_t = g(X_T) + \int_t^T f(s, X_s, Y_s, Z_s)\mathrm{d}s - \int_t^T Z_s\mathrm{d}W_s, \qquad 0 \le t \le T. \tag{4.1b}$$

We assume that the above coefficients μ, σ, f and g all satisfy the conditions of Assumption 1.5.1, and therefore by Theorem 1.5.1 we have a unique triple of random processes (X, Y, Z) satisfying the equations above. Moreover, in order to ensure convergence of the upcoming numerical approximations, we have to make further assumptions with respect to the continuity of the coefficients in time.

Assumption 4.1.1 (Unique Solution of FBSDEs with Hölder Continuity)
Let the μ, σ, f, g in Equation 4.1 be such that the conditions of Assumption 1.5.1 are satisfied. Additionally, assume that

1. *μ, σ – on top of being Lipschitz continuous – are also $\frac{1}{2}$-Hölder continuous with Hölder constant L, i.e. $\forall (t_1, x_1), (t_2, x_2) \in [0, T] \times \mathbb{R}^d$*

$$|\mu(t_1, x_1) - \mu(t_2, x_2)| + |\sigma(t_1, x_1) - \sigma(t_2, x_2)| \le L\left(|t_1 - t_2|^{1/2} + |x_1 - x_2|\right); \tag{4.2}$$

2. *f – on top of being Lipschitz continuous – is also $\frac{1}{2}$-Hölder continuous with Hölder constant L, i.e. $\forall (t_1, x_1, y_1, z_1), (t_2, x_2, y_2, z_2) \in [0, T] \times \mathbb{R}^d \times \mathbb{R} \times \mathbb{R}^{1\times d}$*
$$|f(t_1, x_1, y_1, z_1) - f(t_2, x_2, y_2, z_2)| \leq L \left(|t_1 - t_2|^{1/2} + |x_1 - x_2| + |y_1 - y_2| + |z_1 - z_2|\right). \tag{4.3}$$

We emphasize that the additional assumptions in Assumption 4.1.1 are not to assure the existence of unique solutions but rather to provide regularity for discrete numerical approximations.

In what follows, we always denote a discretization of the underlying time horizon by
$$\pi^N := \{0 = t_0 < t_1 < \cdots < t_{N-1} < t_N = T\}, \tag{4.4}$$
and its mesh-size by $|\pi| := \sup_{i=1,\dots,N} t_i - t_{i-1}$. In order to ease the notation we put $X_n^\pi, Y_n^\pi, Z_n^\pi$ for the discrete approximations of their continuous counterparts $X_{t_n}, Y_{t_n}, Z_{t_n}$ for each $0 \leq n \leq N$. As we shall see, the general error bounds of discrete BSDE schemes rely on the L^2*-regularity* of the Z-process which is defined as follows
$$\varepsilon^Z(|\pi|) := \mathbb{E}\left[\sum_{n=0}^{N-1} \int_{t_i}^{t_{i+1}} \left|Z_t - \overline{Z}_{t_i}\right|^2 \mathrm{d}t\right], \quad \text{where} \quad \overline{Z}_{t_i} := \frac{1}{\Delta t_i} \mathbb{E}_i\left[\int_{t_i}^{t_{i+1}} Z_t \mathrm{d}t\right]. \tag{4.5}$$

4.2 Discretization of the FBSDEs

In the following section, we introduce discretization schemes for both SDEs and BSDEs. We start off by briefly recalling the standard Euler–Maruyama scheme and its main properties for forward equations. Thereafter, we explain how the ideas of numerical methods for forward SDEs can be applied to discretize BSDEs. We highlight that the need for the arising conditional expectations stems from the fact that performing backward Euler stepping over the reversed time horizon is not possible, since it would violate the adaptivity condition required from the unique solution of BSDEs – see Definition 1.3.1.

4.2.1 Discretization of SDEs

The solution of forward stochastic differential equations can be approximated numerically by different discretization procedures. For the purpose of this work, we only consider the well-known Euler–Maruyama scheme and refer the more interested reader to, e.g., [41] for a more detailed discussion on the topic. The idea behind Euler–Maruyama is to discretize both the time and stochastic integrals in Equation 4.1a by the left-rectangle rule, which subsequently leads to the following iterative scheme for each $n = 0, \dots, N-1$
$$\begin{aligned} X_0^\pi &= x_0, \\ X_{n+1}^\pi &= X_n^\pi + \mu(t_n, X_n^\pi)\Delta t_n + \sigma(t_n, X_n^\pi)\Delta W_n, \end{aligned} \tag{4.6}$$
where $\Delta t_n := (t_{n+1} - t_n)$ and $\Delta W_n := W_{t_{n+1}} - W_{t_n}$. It is of common knowledge that the scheme above has strong convergence properties established by the following theorem.

Theorem 4.2.1 (Strong Convergence of the Euler–Maruyama Scheme for SDEs)
Let the conditions of Assumption 4.1.1 hold. Then
$$\max_{0\leq i\leq N-1} \mathbb{E}\left[\sup_{t\in[t_i,t_{i+1}]} |X_t - X_i^\pi|^2\right] \leq C\left(1 + |x_0|^2\right)|\pi|, \tag{4.7}$$
with some constant C independent from the time grid.

Proof. See, e.g., [12, Theorem 5.3.1]. □

As in our case the solution of the forward diffusion in the decoupled FBSDE system can be solved independently from the solution of the BSDE part, we can apply the Euler–Maruyama scheme without having any prior knowledge about the trajectories of (Y, Z), and gather discrete approximations $\{X_n^\pi\}_{0\leq n\leq N}$ for the continuous solution $\{X\}_{0\leq t\leq T}$. It is worth to notice that due to the uncoupledness of the system in Equation 4.1, one can solve the forward diffusion on a much finer time grid, assure strong convergence by Theorem 4.2.1 and plug the resulting randomness in the backward equation on a sparser time partition. Consequently, in what follows the simulation error can be made arbitrarily small in expectations for any SDE satisfying the conditions of Theorem 4.2.1.

Nonetheless, partly motivated by the observation above, the simulation error of the forward diffusion shall not be of key interest neither for the upcoming error analysis in chapter 6, nor for the numerical examples presented in chapter 7. The reason for this is that in the upcoming applications we restrict our analysis to the special cases of Arithmetic (ABM) and Geometric Brownian Motions (GBM) where the forward process can be simulated analytically for any realization of the underlying Brownian motion – according to the closed-form expressions Equation 1.12 and Equation 1.13 respectively.

Discretization of the Malliavin Derivative's SDE

Before presenting discretization schemes for BSDEs, in the light of forthcoming applications in chapter 5, let us finally have a few words on the numerical solution of the Malliavin derivative's linear SDE given in Equation 2.15. One, similarly to Equation 4.6, can also apply the Euler–Maruyama scheme to gather discrete time approximations for $D_{t_m}X$ solving Equation 2.15, leading to the following discrete scheme for each $n = 0, \dots, N-1$

$$\begin{aligned} D_m X_n^\pi &= \sigma(t_m, X_m^\pi)\mathbb{1}_{m\le n}(n), && 0 \le n \le m,\\ D_m X_{n+1}^\pi &= D_m X_n^\pi + \nabla_x \mu(t_n, X_n^\pi) D_m X_n^\pi \Delta t_n + \nabla_x \sigma(t_n, X_n^\pi) D_m X_n^\pi \Delta W_n, && m < n \le N, \end{aligned} \tag{4.8}$$

for each $0 \le m \le N$ in the discrete time grid. It is important to notice that, unlike in the case of standard forward diffusions, there is an additional difficulty in solving the Malliavin derivative's linear SDE. This stems from the fact that the initial condition $D_m X_m^\pi = \sigma(t_m, X_m^\pi)$ is not fixed, and is already prone to the errors induced by the Euler–Maruyama approximations deployed for X. Therefore, it can be difficult to assure rigorous convergence bounds for the numerical approximation $D_m X^\pi$. However, this source of simulation error shall not impact the results presented later in chapter 7, as for the case of ABM and GBM one can simulate the Malliavin derivatives analytically through their closed-form expressions in Equation 2.9 and Equation 2.10 respectively.

4.2.2 Discretization of BSDEs

Let us now turn to backward equation in Equation 4.1b and motivate a discrete approximation scheme for its numerical solution similar to the Euler–Maruyama scheme above. Notwithstanding, we are faced with an additional difficulty, since, although the nature of the BSDE system would suggest a backward Euler induction for the solution starting off from $t = T$ but this is not allowed due to the adaptivity requirements posed on the solution pair.[1] As we shall see, one thus has to approximate conditional expectations instead, in order to gather discrete time approximations for (Y, Z). To show this, let us consider the BSDE in Equation 4.1b

$$Y_t = g(X_T) + \int_t^T f(s, X_s, Y_s, Z_s)\mathrm{d}s - \int_t^T Z_s \mathrm{d}W_s, \qquad 0 \le t \le T. \tag{4.9}$$

Combining the dynamics for two adjacent points $t_n, t_{n+1} \in \pi^N$, we gather

$$Y_{t_n} = Y_{t_{n+1}} + \int_{t_n}^{t_{n+1}} f(s, X_s, Y_s, Z_s)\mathrm{d}s - \int_{t_n}^{t_{n+1}} Z_s \mathrm{d}W_s, \qquad t_n, t_{n+1} \in \pi^N. \tag{4.10}$$

Additionally, multiplying both sides above by the Brownian increment $\Delta W_{t_n} := W_{t_{n+1}} - W_{t_n} = \int_{t_n}^{t_{n+1}} \mathrm{d}W_s$ and taking conditional expectations projecting on the σ-algebra $\mathcal{F}_{t_n}$ yields

$$\begin{aligned} Y_{t_n}\mathbb{E}\left[\Delta W_{t_n} | \mathcal{F}_{t_n}\right] &= \mathbb{E}\left[Y_{t_{n+1}} \Delta W_{t_n} | \mathcal{F}_{t_n}\right] \\ &\quad + \mathbb{E}\left[\Delta W_{t_n} \int_{t_n}^{t_{n+1}} f(s, X_s, Y_s, Z_s)\mathrm{d}s \middle| \mathcal{F}_{t_n}\right] - \mathbb{E}\left[\int_{t_n}^{t_{n+1}} Z_s \mathrm{d}s \middle| \mathcal{F}_{t_n}\right], \end{aligned} \tag{4.11}$$

by Itô's isometry. Consequently Equation 4.10 and Equation 4.11 lead to the following representation formulas

$$Y_{t_n} = \mathbb{E}\left[Y_{t_{n+1}} + \int_{t_n}^{t_{n+1}} f(s, X_s, Y_s, Z_s)\mathrm{d}s \middle| \mathcal{F}_{t_n}\right], \tag{4.12a}$$

$$0 = \mathbb{E}\left[Y_{t_{n+1}} \Delta W_{t_n} + \int_{t_n}^{t_{n+1}} \left(\Delta W_{t_n} f(s, X_s, Y_s, Z_s) - Z_s\right) \mathrm{d}s \middle| \mathcal{F}_{t_n}\right]. \tag{4.12b}$$

In order to arrive at discrete approximations (Y_n^π, Z_n^π) for (Y_{t_n}, Z_{t_n}), we need to approximate the continuous time integrals in the arguments of the conditional expectations above. In what follows we present two standard approaches for this.

[1]Meaning that the solution at time step n cannot be a function of the solution at time step $n+1$.

Euler Scheme for BSDEs

The Euler scheme for BSDEs – analogously to the Euler scheme for SDEs – is the most extensively studied discrete scheme for backward equations (see, e.g., [6], [5]), which is explicit in both Y and Z. In this approach the integrals in Equation 4.12 are approximated by the left-rectangle rule. Additionally, in order to avoid implicitness in the scheme of Y_n^π we can also use the approximation $f(t_n, X_n^\pi, Y_n^\pi, Z_n^\pi) \approx f(t_n, X_n^\pi, Y_{n+1}^\pi, Z_n^\pi)$ motivated by the Lipschitz continuity of the driver, and subsequently get the following recursive approximation scheme for each $n = N-1, \ldots, 0$

$$Y_N^\pi = g\left(X_N^\pi\right), Z_N^\pi = \sigma\left(t_N, X_N^\pi\right) \nabla g\left(X_N^\pi\right), \tag{4.13a}$$

$$Z_n^\pi = \frac{1}{\Delta t_n} \mathbb{E}\left[\Delta W_n Y_{n+1}^\pi \middle| \mathcal{F}_n\right], \tag{4.13b}$$

$$Y_n^\pi = \mathbb{E}\left[Y_{n+1}^\pi + \Delta t_n f(t_n, X_n^\pi, Y_{n+1}^\pi, Z_n^\pi) \middle| \mathcal{F}_n\right]. \tag{4.13c}$$

As it was shown by Zhang in [5], the approximation scheme above admits to convergence bounds established by the following theorem.

Theorem 4.2.2 (Convergence of Euler Scheme for BSDEs)
Let the conditions of Assumption 4.1.1 hold. Consider approximations (Y^π, Z^π) given by the discrete scheme in Equation 4.13. Then

$$\sup_{0\le i\le N} \mathbb{E}\left[\left|Y_{t_i} - Y_i^\pi\right|^2\right] + \mathbb{E}\left[\sum_{i=0}^{N-1} \int_{t_i}^{t_{i+1}} \left|Z_s - Z_i^\pi\right|^2 \mathrm{d}s\right]$$
$$\le C\left(\mathbb{E}\left[\left|g\left(X_{t_N}\right) - g\left(X_N^\pi\right)\right|^2\right] + |\pi| + \varepsilon^Z(|\pi|)\right), \tag{4.14}$$

where $\varepsilon^Z(|\pi|)$ is the L^2-regularity of the control process defined in Equation 4.5 and C is a constant independent of the time partition.

Proof. See, e.g., [5, Theorem 5.6]. □

Finally, let us make two important remarks. Firstly, in case the terminal condition g is also Lipschitz continuous then it can be shown that $\varepsilon^Z(|\pi|)$ is an $\mathcal{O}(|\pi|)$ function – see, e.g., [5]. Subsequently the right-hand side of Equation 4.14 simply becomes $C|\pi|$, giving first-order convergence. Secondly, allowing for implicitness in the approximation of Y does not change the convergence properties – see, e.g., [6].

Theta-Scheme for BSDEs

An other technique to discretize the continuous integrals in Equation 4.12 is to use the well-known theta-scheme, an approach which, in the context of BSDEs, was first proposed by the authors in [42] and later generalized in [43]. This way, the integrals in Equation 4.12a and Equation 4.12b, with some $\vartheta_y, \vartheta_z \in [0,1]$ respectively, are approximated by

$$Y_{t_n} \approx \mathbb{E}\left[Y_{t_{n+1}} + \Delta t_n \left(\vartheta_y f_{t_n} + (1-\vartheta_y) f_{t_{n+1}}\right) \middle| \mathcal{F}_{t_n}\right], \tag{4.15a}$$

$$0 \approx \mathbb{E}\left[Y_{t_{n+1}} \Delta W_{t_n} + \Delta t_n \left[-\vartheta_z Z_{t_n} + (1-\vartheta_z)\left(f_{t_{n+1}} \Delta W_{t_n} - Z_{t_{n+1}}\right)\right] \middle| \mathcal{F}_{t_n}\right], \tag{4.15b}$$

where we introduced the notation $f_s := f(s, X_s, Y_s, Z_s)$. Rearranging the equations, using standard properties of conditional expectations results in the following recursive conditional expectation scheme for each $n = N-1, \ldots, 0$

$$Y_N^\pi = g\left(X_N^\pi\right), Z_N^\pi = \sigma\left(t_N, X_N^\pi\right) \nabla g\left(X_T^\pi\right), \tag{4.16a}$$

$$Z_n^\pi = \frac{1}{\vartheta_z \Delta t_n} \mathbb{E}\left[Y_{n+1}^\pi \Delta W_n + (1-\vartheta_z)\Delta t_n \left(f(t_{n+1}, X_{n+1}^\pi, Y_{n+1}^\pi, Z_{n+1}^\pi)\Delta W_n - Z_{n+1}^\pi\right) | \mathcal{F}_n\right], \tag{4.16b}$$

$$Y_n^\pi = \vartheta_y \Delta t_n f(t_n, X_n^\pi, Y_n^\pi, Z_n^\pi) + \mathbb{E}\left[Y_{n+1}^\pi + (1-\vartheta_y)\Delta t_n f(t_{n+1}, X_{n+1}^\pi, Y_{n+1}^\pi, Z_{n+1}^\pi) \middle| \mathcal{F}_n\right]. \tag{4.16c}$$

We remark that the discrete scheme in Equation 4.16 coincides with that of the implicit Euler scheme for BSDEs when $\vartheta_y = \vartheta_z = 1$. Moreover, it is also worth to notice that for any $\vartheta_y > 0, \vartheta_z \in [0,T]$ the scheme is implicit in Y and explicit in Z. We remark that in order to deal with implicit conditional expectations, one most often uses a Picard iteration sequence such as the one established by Corollary 1.4.1 in chapter 1.

To conclude the discussion let us finally have a few words on the errors induced by the discrete scheme in Equation 4.16. Zhao et. al in [42] proved that the scheme above exhibits $\mathcal{O}(|\pi|)$ convergence for any $\vartheta_y, \vartheta_z \in [0,1]$ under the rather strict conditions that: f, g are both smooth and have bounded partial derivatives of all orders; f does not depend on Z; and the underlying forward diffusion is a Brownian motion. Additionally, they also showed that the Crank–Nicolson scheme ($\vartheta_y = \vartheta_z = 1/2$) admits to second-order convergence under the same assumptions. These results have been extended by Ruijter and Oosterlee in [44], where they relaxed the conditions allowing for ABM as the forward diffusion, while keeping second-order convergence in the Y-process. Nevertheless, to the best of our knowledge, second-order convergence for the general case of Assumption 4.1.1 is yet to be proven.

4.3 Approximating Conditional Expectations

As we have just seen, numerical methods for solving the backward part of an FBSDE system are closely related to approximating conditional expectations at each step in time. Moreover, by the schemes Equation 4.13 and Equation 4.16, we have also seen that these conditional expectations are nested recursively backwards in time. Therefore, it is of interest to understand how this can be done efficiently with high precision in a numerical algorithm. The trivial way to calculate conditional expectations for a certain random phenomenon would be to perform inner Monte Carlo simulations on a given Monte Carlo sample and then calculate the mean of the paths starting off from each realization of X_n^π. These schemes, however, are very expensive computationally for large Monte Carlo samples. In order to overcome such computational burdens, we are restricted to other approaches. There have been many methodologies proposed in the literature for estimating conditional expectations. For instance, in [44] the authors propose a Fourier cosine expansion method for smooth functions $u(t, X_n^\pi)$ of forward diffusions X whose characteristic functions are known. In the context of this work, we use another method called Least-Squares Monte Carlo (LSMC) which is explained in the section below.

4.3.1 Least-Squares Monte Carlo

LSMC was first proposed in a paper by Longstaff and Schwartz [45] to approximate conditional expectations in the context of American option pricing. The method has since been extensively studied and used in a wide range of numerical problems. In particular, Gobet et al. in [3] applied it to approximate conditional expectations arising in the discrete scheme of Equation 4.13. We hereby explain the main idea behind LSMC focusing on the context of BSDEs, and refer the more interested reader to the survey paper [46] for more details.

LSMC is built on Markovianity, namely it exploits the fact that in case the driving random phenomenon is Markovian the conditional expectations can be expressed as deterministic functions of the realizations of the underlying randomness. Additionally, it is not too difficult to show that this deterministic function solves a minimization problem as stated in the following proposition.

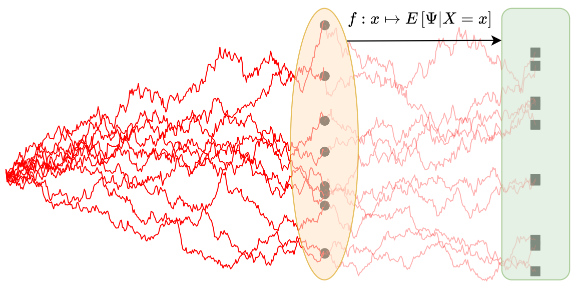


Figure 4.1: Least-Squares Monte Carlo Illustration. Red curves: realizations of a random phenomenon. Black dots: current states. Black squares: future states to be projected through conditional expectation.

Proposition 4.3.1 (Conditional Expectation as Minimizer)
Let $\mathcal{G} \subseteq \mathcal{F}$ be two (sub-)σ-algebras. Let $X \in \mathbb{L}^2_{\mathcal{G}}(\mathbb{R}^d)$ and $\Psi \in \mathbb{L}^2_{\mathcal{F}}(\mathbb{R})$. Assume that the underlying stochastics is a Markov process, i.e. $\sigma(X) \equiv \mathcal{G}$. Denote the set of measurable functions $f : \mathbb{L}^2_{\mathcal{G}}(\mathbb{R}^d) \to \mathbb{L}^2_{\mathcal{F}}(\mathbb{R})$ by $\mathcal{D}^{\mathcal{F}}_{\mathcal{G}}(\mathbb{R}^d; \mathbb{R})$. Then

$$\mathbb{E}\left[\Psi|\mathcal{G}\right] = \underset{f \in \mathcal{D}^{\mathcal{F}}_{\mathcal{G}}(\mathbb{R}^d;\mathbb{R})}{\arg\min}\ \mathbb{E}\left[\left|\Psi - f(X)\right|^2\right]. \tag{4.17}$$

Proof. The proof is an elementary consequence of the law of total probability. We refer the more interested reader to, e.g., [47] for more details. □

In the light of the proposition above – recalling that in the context of FBSDEs we have that for each point in time $Y_t = u(t, X_t), Z_t = v(t, X_t)$ as explained in subsection 1.5.1 – we can break down the approximation of the conditional expectations arising in the discrete schemes above to $d + 1$-many minimization problems of the form of Equation 4.17.

Ordinary Least-Squares Regression

Nevertheless, in order to have a fully implementable discrete scheme, we need to make two further estimations. Firstly, we need to *approximate* the infinite dimensional function space $\mathcal{D}_{\mathcal{G}}^{\mathcal{F}}(\mathbb{R}^d;\mathbb{R})$ by the *span* of a finite set of (independent) basis functions $\{\phi_i\}_{i=1}^K$. This approximation, for any $f \in \mathcal{D}_{\mathcal{G}}^{\mathcal{F}}(\mathbb{R}^d;\mathbb{R})$, can be formulated as follows

$$f(X) \approx \sum_{k=1}^{K} \alpha_k \phi_k(X), \tag{4.18}$$

where α_k expresses the weight of ϕ_k in the decomposition. Secondly, we also need to gather approximations for the true expectations in Equation 4.17. This can be done by taking the empirical sample mean of M independent realizations of the random phenomenon

$$\mathbb{E}\left[\left|\Psi - \sum_{k=1}^{K} \alpha_k \phi_k(X)\right|^2\right] \approx \frac{1}{M}\sum_{j=1}^{M}\left|\Psi_j - \sum_{k=1}^{K} \alpha_k \phi_k(X_j)\right|^2. \tag{4.19}$$

Introducing the notations $\mathbf{\Phi} \in \mathbb{R}^{M\times K} : [\mathbf{\Phi}]_{j,i} := \alpha_i \phi_i(X_j)$, $\mathbf{\Psi} := (\Psi_1, \dots, \Psi_M)$ and $\boldsymbol{\alpha} := (\alpha_1, \dots, \alpha_K)$ it is well-known – see, e.g., [46] – that the best estimator is given by

$$\widehat{\boldsymbol{\alpha}} = (\mathbf{\Phi}^T\mathbf{\Phi})^{-1}\mathbf{\Phi}^T\mathbf{\Psi}. \tag{4.20}$$

Finally, to conclude the discussion of LSMC we make two important remarks. First, the performance of LSMC is inherently linked to the choice of the set of basis functions with which we approximate the space of all measurable functions. Such a choice should be extensive enough to span a wide function class but not too large in order to avoid overfitting. Second, Ordinary Least-Squares LSMC has a closed form expression given by Equation 4.20 unlike neural network LSMC introduced later in section 5.2.

4.4 Deep BSDE Solvers

In the following section we introduce two base approaches developed to tackle the FBSDE problem, using deep learning. These algorithms were formulated with the purpose to solve high-dimensional second-order parabolic PDEs by translating them to the corresponding FBSDE system through the probabilistic representation provided by the general Feynman–Kac relations[2] in Theorem 1.5.2. In this work, we are concerned with the solution of BSDEs and therefore we focus on such purely probabilistic methods which are designed to deal with the system in Equation 4.1. The main ideas of these aforementioned approaches can be classified in two distinct categories, which we briefly explain below.

4.4.1 Forward Deep BSDE Solver

The *Forward Deep BSDE* solver was first proposed in a paper by E et. al in [48] and [7]. The main idea of their paper is built on the following observation. One can exploit the fact that due to Markovianity the initial values of the backward processes are deterministic functions of X_0. In fact, for a fixed initial condition $X_0 = x_0$, we have that the solutions of the backward equations are also fixed $Y_0 = y_0, Z_0 = 0$. Subsequently, one can reformulate the BSDE with an Euler–Maruyama like scheme, forward in time. In order to do this, one needs to parametrize the solution of the BSDE part at $t = 0$ by $\mathcal{Y}_0 = \theta_0^y, \mathcal{Z}_0 = \theta_0^z$ where $\theta_0^y \in \mathbb{R}, \theta_0^z \in \mathbb{R}^d$ are trainable parameters. Moreover, one can also parametrize each time step's control process by a neural network $\mathcal{Z}_n\left(\cdot|\theta_n^z\right) := \mathcal{NN}\left(\cdot|\theta_n^z\right) : \mathbb{R}^d \to \mathbb{R}^d, n = 1, \dots, N-1$ where $\theta_n^z \in \mathbb{R}^{p_n}$ with p_n denoting the number of parameters in the n'th neural network. Thereafter, using these estimations we can forward propagate the initial random guesses through the whole discretized time window π^N in terms of an Euler–Maruyama scheme

$$\begin{aligned}\mathcal{Y}_0 &= \theta_0^y, \mathcal{Z}_0 = \theta_0^z,\\ \mathcal{Y}_{n+1} &= \mathcal{Y}_n - f\left(t_n, X_n^\pi, \mathcal{Y}_n, \mathcal{Z}_n\left(X_n^\pi|\theta_n^z\right)\right)\Delta t_n + \mathcal{Z}_n\left(X_n^\pi|\theta_n^z\right)\Delta W_n.\end{aligned} \tag{4.21}$$

Having finished the forward propagation, one can gather estimations $\mathcal{Y}_N(\theta_0^y, \theta_0^z, \theta_1^z, \dots, \theta_{N-1}^z)$ depending on the initial parameters and all parameters in the sequence of neural networks. By collecting these

[2]We remark that this special "equivalence" between PDEs and FBSDEs establishes a way to approximate the solution of the backward equation by means of an *unsupervised learning problem* over a deterministic spatial domain, formulated on the corresponding PDE in Equation 1.63. Such methods are usually collected under the term *Deep Galerkin Methods* (DGM), a few examples of which can be found in [25], [26] or [27].

Algorithm 4: Forward Deep BSDE (FWDBSDE)

```
**Data:** $\{X_n\}_{t_n\in\pi^N}$ – forward diffusion's Monte Carlo sample of size $M$
**Result:** $(Y^\pi, Z^\pi)$ – BSDE's solution over time grid $\pi^N$
$\mathcal{Y}_0 \leftarrow \theta_0^y, \mathcal{Z}_0 \leftarrow \theta_0^z$ – take initial random guesses
$\mathcal{Z}_n(\cdot|\theta_n^z) \leftarrow \mathcal{NN}(\cdot|\theta_n^z) : \mathbb{R}^d \to \mathbb{R}^d$ – neural network parametrizations
**for** $n = 0, \dots, N-1$ **do**
    $\mathcal{Y}_N$ – forward propagate until terminal time according to Equation 4.21
        $$\mathcal{Y}_{n+1} \leftarrow \mathcal{Y}_n - f(t_n, X_n^\pi, \mathcal{Y}_n, \mathcal{Z}_n(X_n^\pi|\theta_n^z))\Delta t_n + \mathcal{Z}_n(X_n^\pi|\theta_n^z)\Delta W_n.$$
**end**
$\Theta \leftarrow \left(\theta_0^Y, \theta_0^Z, \theta_1^Z, \dots, \theta_{N-1}^Z\right)$ – collect model parameters
$\mathcal{L}(\Theta)$ – define loss function according to Equation 4.22
$\widehat{\Theta} \leftarrow \arg\min_\Theta [\mathcal{L}(\Theta)]$ ; – optimize empirical loss
$Y_0^\pi \leftarrow \theta_0^y$, $Z_0^\pi \leftarrow \theta_0^z$, $Y_n^\pi \leftarrow \left(Y_0^\pi - \sum_{i=1}^{n-1} f(t_i, X_i^\pi, Y_i^\pi, Z_i^\pi)\Delta t_i + \sum_{i=1}^{n-1} Z_i^\pi \Delta W_i\right)$,
 $Z_n^\pi \leftarrow \mathcal{Z}_n\left(X_n^\pi|\widehat{\theta}_n^z\right)$ – gather approximations
```

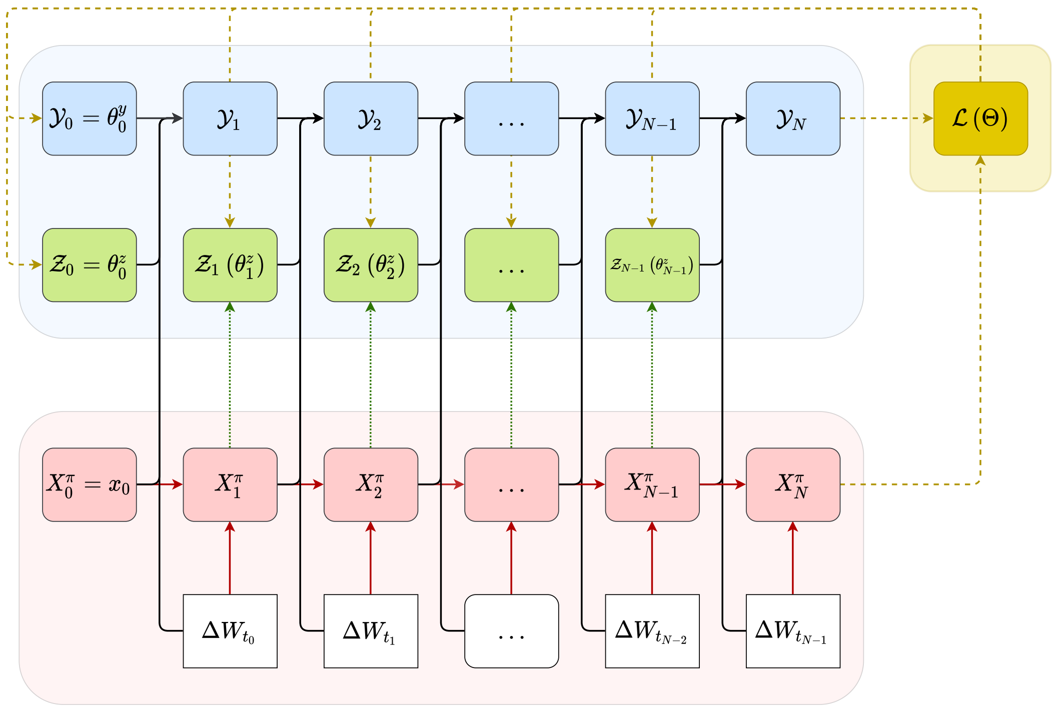


Figure 4.2: Forward Deep BSDE Architecture. Red area: forward part; blue area: backward part; yellow area: optimization part. The coding of the arrows is as follows. Solid-red: forward Euler SDE approximation; solid-black: forward Euler BSDE approximation; dotted-green: neural network approximation; dashed-yellow: optimization input and output.

estimations, one can compare them against the known terminal condition of the FBSDE system and formulate a loss function according to

$$\mathcal{L}(\Theta) := \mathbb{E}\left[\left|\mathcal{Y}_N\left(\theta_0^y, \theta_0^z, \theta_1^z, \ldots, \theta_{N-1}^z\right) - g(X_N^\pi)\right|^2\right]. \tag{4.22}$$

Minimizing this loss and gathering the optimal parameter set $\widehat{\Theta} := \left(\widehat{\theta}_0^y, \widehat{\theta}_0^z, \ldots, \widehat{\theta}_{N-1}^z\right)$ thus then leads to approximations $Z_n^\pi := \mathcal{Z}_n\left(X_n^\pi|\widehat{\theta}_n^z\right)$ and $Y_n^\pi := \mathcal{Y}_n(\widehat{\Theta})$ which satisfy the forward Euler discretization of the BSDE and also adhere to the known terminal condition. The full algorithm is collected in algorithm 4 and the architecture of the Forward Deep BSDE solver is also depicted in Figure 4.2 for illustrative purposes.

Error Analysis

An interesting aspect of the proposed scheme is the error figures it obeys to. Since the problem formulation is very involved, and it is difficult to guarantee rigorous regression error bounds for neural networks due to the nature of stochastic optimization methods, it is non-trivial to prove that the errors induced by Equation 4.21 tend to zero for infinitely small mesh-sizes. Nevertheless, the authors in [49] prove a posteriori error estimate for the Forward Deep BSDE Method, in a more general case of *weakly* coupled FBSDEs, which also stands for the uncoupled system of our interest. This is stated in the following theorem.

Theorem 4.4.1 (Posteriori Error Estimate Forward Deep BSDE, Han–Long [49])
Under some regularity assumptions, there exists a constant C, independent of the time grid such that for sufficiently small time steps we have

$$\sup_{t\in[0,T]}\left[\mathbb{E}\left[\left|X_t - \widehat{X}_t^\pi\right|^2\right] + \mathbb{E}\left[\left|Y_t - \widehat{Y}_t^\pi\right|^2\right]\right] + \int_0^T \mathbb{E}\left[\left|Z_t - \widehat{Z}_t^\pi\right|^2\right] \mathrm{d}t$$
$$\leq C\left(|\pi| + \mathbb{E}\left[|g(X_N^\pi) - Y_T^\pi|^2\right]\right), \tag{4.23}$$

where $\widehat{X}_t^\pi := X_i^\pi, \widehat{Y}_t^\pi := Y_i^\pi, \widehat{Z}_t^\pi := Z_i^\pi$ for $t \in [t_i, t_{i+1})$.

Proof. See [49, Pg. 11, Theorem 1'] □

Let us briefly interpret this result. We see that the overall error of the discrete scheme is bounded by two terms. The first term corresponds to the Euler–Maruyama discretizations of Equation 4.1a and Equation 4.1b, and it says that the error stemming from approximating continuous random processes by discrete counterparts is an $\mathcal{O}(|\pi|)$ function, i.e. it vanishes linearly while decreasing the mesh-size of the time grid. On the other hand, the second term is an approximation error induced by the model in Equation 4.21. This expression corresponds to the loss function of the Forward Deep BSDE method and in fact incorporates all the *regression* errors of each neural network in the sequence. It is important to highlight that the error bound is posteriori, i.e. it does not guarantee convergence of the scheme as $|\pi| \to 0$ only provides a way to evaluate its accuracy.

4.4.2 Backward Deep BSDE Solver

Motivated by the forward Deep BSDE there has been another methodology suggested in the literature to tackle the FBSDE problem using deep learning. This approach was first proposed by Wang et. al [8] for the special case of zero driver BSDEs – recall Equation 1.16. It was later extended by Huré et. al in [9] for general BSDEs and by Chen and Wan in [10] for reflected BSDEs in the context of American option pricing. In what follows we explain this approach briefly, mostly focused on how it differs from the forward scheme seen before.

This method is closer to the original conditional expectation scheme, as it solves the BSDE by a sequence of recursive optimizations, backwards in time. Nevertheless, the main idea is similar to that of the Forward Deep BSDE method: one can parametrize the initial values of the backward equation's solutions by trainable parameters $Y_0^\pi = \theta_0^y, Z_0^\pi = \theta_0^z$. However, on top of parametrizing each further time step's control process, one also parametrizes the Y-process by neural networks. This yields approximations $\mathcal{Y}_n(\cdot|\theta_n^y) = \mathcal{NN}(\cdot|\theta_n^y) : \mathbb{R}^d \to \mathbb{R}$ and $\mathcal{Z}_n(\cdot|\theta_n^z) = \mathcal{NN}(\cdot|\theta_n^z) : \mathbb{R}^d \to \mathbb{R}^d$, with $\theta_n^y \in \mathbb{R}^{p_n^y}$ and $\theta_n^z \in \mathbb{R}^{p_n^z}$ where p_n^y, p_n^z denote the number of parameters in the corresponding networks. The training of the model is then done as follows. One first collects the terminal conditions according to $Y_N^\pi = g(X_N^\pi)$. Subsequently

Algorithm 5: Backward Deep BSDE (BWDBSDE)

```
**Data:** $\{X_n\}_{t_n\in\pi^N}$ – forward diffusion's Monte Carlo sample of size $M$
**Result:** $(Y^\pi, Z^\pi)$ – BSDE's solution over time grid $\pi^N$
$Y_N^\pi \leftarrow g(X_N^\pi)$, $Z_N^\pi \leftarrow \sigma(t_N, X_N^\pi)\nabla_x g(X_N^\pi)$ – collect terminal condition
**for** $n = N-1, \dots, 0$ **do**
    $\mathcal{Y}_n(\cdot|\theta_n^y) \leftarrow \mathcal{NN}(\cdot|\theta_n^y) : \mathbb{R}^d \to \mathbb{R}$, $\mathcal{Z}_n(\cdot|\theta_n^z) \leftarrow \mathcal{NN}(\cdot|\theta_n^z) : \mathbb{R}^d \to \mathbb{R}^d$ – neural network
     parametrizations
    $\Theta_n \leftarrow (\theta_n^y, \theta_n^z)$ – collect time step parameters
    $\mathcal{L}(\Theta_n)$ – define loss function according to Equation 4.24
    $\widehat{\Theta}_n \leftarrow \arg\min_{\Theta_n} [\mathcal{L}(\Theta_n)]$ – optimize empirical loss
    $Y_n^\pi \leftarrow \mathcal{Y}_n\left(X_n^\pi|\widehat{\theta}_n^y\right)$, $Z_n^\pi \leftarrow \mathcal{Z}_n\left(X_n^\pi|\widehat{\theta}_n^z\right)$ – gather approximations
**end**
```

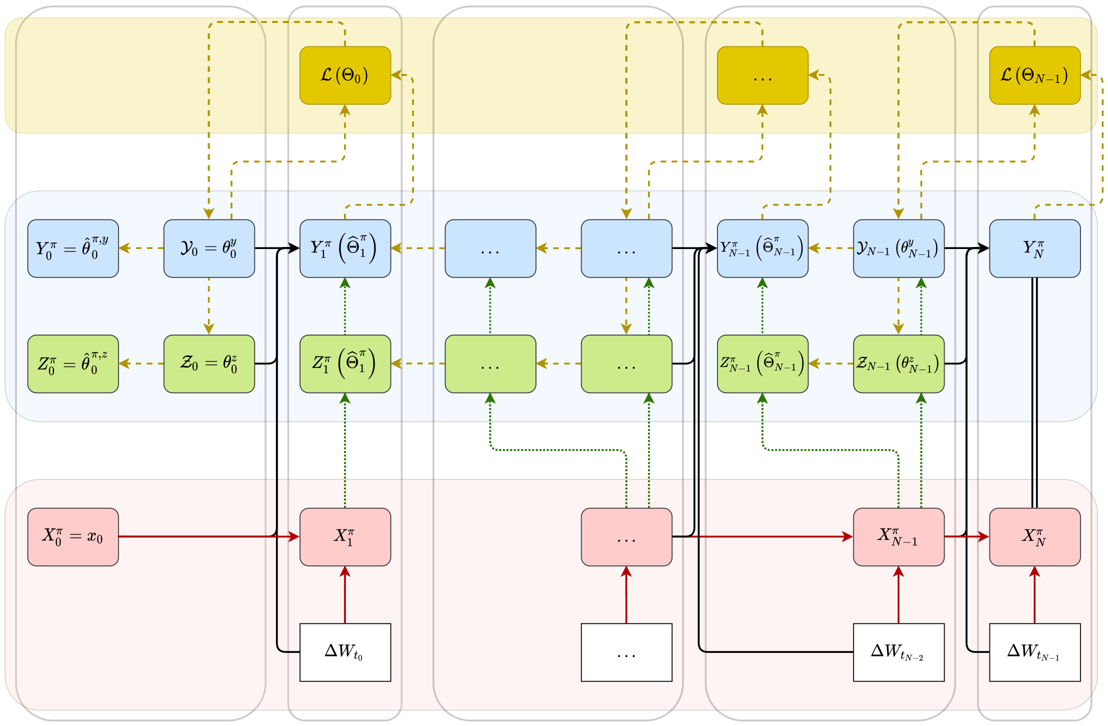

$\mathcal{L}(\Theta_0)$
$\dots$
$\mathcal{L}(\Theta_{N-1})$
$Y_0^\pi = \hat{\theta}_0^{\pi,y}$
$\mathcal{Y}_0 = \theta_0^y$
$Y_1^\pi\left(\widehat{\Theta}_1^\pi\right)$
$\dots$
$\dots$
$Y_{N-1}^\pi\left(\widehat{\Theta}_{N-1}^\pi\right)$
$\mathcal{Y}_{N-1}\left(\theta_{N-1}^y\right)$
Y_N^π
$Z_0^\pi = \hat{\theta}_0^{\pi,z}$
$\mathcal{Z}_0 = \theta_0^z$
$Z_1^\pi\left(\widehat{\Theta}_1^\pi\right)$
$\dots$
$\dots$
$Z_{N-1}^\pi\left(\widehat{\Theta}_{N-1}^\pi\right)$
$\mathcal{Z}_{N-1}\left(\theta_{N-1}^z\right)$
$X_0^\pi = x_0$
X_1^π
$\dots$
X_{N-1}^π
X_N^π
ΔW_{t_0}
$\dots$
$\Delta W_{t_{N-2}}$
$\Delta W_{t_{N-1}}$

Figure 4.3: Backward Deep BSDE Architecture. Red area: forward part; blue area: backward part; yellow area: optimization part. The coding of the arrows is as follows. Solid-red: forward Euler SDE approximation; solid-black: forward Euler BSDE approximation; dotted-green: neural network approximation; dashed-yellow: optimization input and output.

for each previous step in time one solves a regression problem given by the forward Euler–Maruyama discretization of the dynamics of the BSDE in Equation 4.1b, corresponding to the following loss function for each $n = N-1, \dots, 0$

$$\mathcal{L}(\Theta_n) = \mathbb{E}\left[\left|Y^\pi_{n+1} - \mathcal{Y}\left(X^\pi_n|\theta^y_n\right) + f\left(t_n, X^\pi_n, \mathcal{Y}_n\left(X^\pi_n|\theta^y_n\right), \mathcal{Z}_n\left(X^\pi_n|\theta^z_n\right)\right)\Delta t_n - \mathcal{Z}_n\left(X^\pi_n|\theta^z_n\right)\Delta W_n\right|^2\right], \tag{4.24}$$

where we introduced the notation $\Theta_n := (\theta^y_n, \theta^z_n)$. The minimizer of this loss function thus yields an optimal parameter set $\widehat{\Theta} := \left(\widehat{\theta}^y_n, \widehat{\theta}^z_n\right) \in \mathbb{R}^{p^y_n + p^z_n}$ such that the approximations $Y^\pi_n := \mathcal{Y}_n\left(X^\pi_n|\widehat{\theta}^y_n\right)$ and $Z^\pi_n := \mathcal{Z}_n\left(X^\pi_n|\widehat{\theta}^z_n\right)$ closely adhere to an Euler–Maruyama discretization of Equation 4.1b and can therefore be considered good approximations of (Y, Z). The full algorithm is collected in algorithm 5. The architecture of the Backward Deep BSDE scheme is depicted in Figure 4.3 for illustrative purposes.

Finally, we remark that the scheme above has a modification in which only the Y-process is parametrized by neural networks and – motivated by the Feynman–Kac lemma in Theorem 1.2.2 and the connection in Equation 2.28 – the Z-process is approximated by the derivative of the network of $\mathcal{Y}_n$

$$\mathcal{Z}_n\left(X^\pi_n\right) = \sigma(t_n, X^\pi_n)\nabla_x \mathcal{Y}_n\left(X^\pi_n|\theta^y_n\right), \tag{4.25}$$

taken by automatic differentiation – see section 3.5. However, as this modification usually yields numerically less stable results in high-dimensions because of the training of the derivative of a network – see [9] for more details –, we restrict our analysis to the original formulation explained above.

Multi-Step Discretization

As we shall see in the next chapter, multi-step discretization schemes play an important role in one of the proposed algorithms – see algorithm 7 in particular. Therefore, we remark that the Backward Deep BSDE scheme described above has an extension proposed by Germain et. al in [50] to multi-step Euler–Maruyama discretizations. In fact, instead of discretizing the BSDE in between two adjacent time steps as in Equation 4.10, they discretize Equation 4.9 by splitting it up to a sum of integrals between adjacent time steps, each discretized by the left-rectangle rule of the Euler–Maruyama scheme. Through this approach, their loss consequently becomes for each $n = N-1, \dots, 0$

$$\mathcal{L}\left(\Theta_n\right) = \mathbb{E}\Bigg[\Bigg|Y^\pi_N - \mathcal{Y}_n(X^\pi_n|\theta^y_n) + f\left(t_n, X^\pi_n, \mathcal{Y}_n\left(X^\pi_n|\theta^y_n\right), \mathcal{Z}_n\left(X^\pi_n|\theta^z_n\right)\right)\Delta t_n - \mathcal{Z}_n\left(X^\pi_n|\theta^z_n\right)\Delta W_n \\ .+\sum_{i=n+1}^{N-1} f\left(t_i, X^\pi_i, Y^\pi_i, Z^\pi_i\right)\Delta t_i - Z^\pi_i \Delta W_i\Bigg|^2\Bigg]. \tag{4.26}$$

This approach has shown similar performance as its one-step counterpart with mitigated regression errors.

Error Analysis

The authors in [9, Theorem 4.1] show that the above proposed recursive scheme admits to similar error figures as in the case of the forward Deep BSDE method. However, as their error bound depends on quantities only defined in chapter 6, for now we refrain from its concrete formulation and rather describe them qualitatively. Analogously to the first term on the right-hand side of Equation 4.23 in Theorem 4.4.1, the error bound of the Backward Deep BSDE scheme also has an $\mathcal{O}(|\pi|)$ term on top of which we have quantities corresponding to the quality of neural network regressions. The authors in [9] then motivate the consistency of their scheme by referring to the universal approximation capability of neural networks provided by Theorem 3.2.2. The precise formulation of the theorem is given in Theorem 6.4.1 and Theorem 6.4.2 for the one- and multi-step schemes respectively.

4.4.3 Comparison of Deep BSDE Methods

To conclude our discussion on Deep BSDE solvers, let us finally compare the two approaches described above. For the sake of readability, we split up the comparison to the following list of aspects which are also collected in Table 4.1. It is important to highlight that the performances below are only characterized qualitatively. We refer to the numerical experiments presented in chapter 7 for their precise quantifications on specific problems.

	Forward Deep BSDE	Backward Deep BSDE
Consistency	posteriori guarantee	apriori guarantee (incl. regression err.)
Speed	faster	slower
Model Complexity	higher	lower
Memory Consumption	higher	lower
Robustness	prone to local optima	prone to rolling errors
Accuracy at $t = 0$	Y_0: good Z_0: good	Y_0: good Z_0: good
Accuracy at $0 < t < T$	Y_t: poor Z_t: poor	Y_t: good Z_t : satisfactory

Table 4.1: Deep BSDE Methods Comparison

- Consistency: the theoretical bounds of each algorithm depend on the approximation capabilities of neural networks which are difficult to give rigorous bounds for. Nevertheless, in the case of the Forward Deep BSDE solver the final approximation accuracy can only be measured posteriori, whereas its backward counterpart provides an apriori error estimate which, however, still depends on the regression errors of neural networks.

- Speed: due to the collective training of all the parameters in the model within one optimization cycle, the Forward Deep BSDE method excels in terms of speed. In fact, as demonstrated by [51, Test 1], the Forward Deep BSDE method reaches the same error figure at $t = 0$ as its backward alternative within two orders of magnitude less time for certain problems. This effect is mostly due to the computationally intensive recursive, multiple optimizations performed at each time step in the backward algorithm.

- Model Complexity and Size: as seen above, the Forward Deep BSDE method parametrizes the BSDE with a sequence of neural networks which are trained within the same optimization cycle. Contrarily, the Backward Deep BSDE method splits up the optimization to multiple regressions which are solved recursively backwards in time. Consequently, the backward approach yields smaller learning problems. In fact, as it is demonstrated by [9], for fine time grids – i.e. where N is large – the Forward Deep BSDE method often does not even fit into memory.

- Robustness: strongly related to the point above, because of the smaller learning problems, the Backward Deep BSDE method is less prone to local optima. The authors in [9] present multiple examples – even in low-dimensional problems – where the Forward Deep BSDE method diverges for fine time grids and long time horizons. Moreover, the Forward Deep BSDE method is also known to be sensitive for the initial choice of the parameters θ_0^y, θ_0^z. If one starts the optimization far off from the true values Y_0, Z_0, then the algorithm is susceptible to fall into local optima and often does not converge. On the other hand, the backward approach is more susceptible to the effect of *rolling errors*, i.e. the regression targets at time step n include all the fixed approximation errors of time steps $m = n+1, \dots, N-1$. Because of this phenomenon, one must obtain very high accuracy for time steps close to the terminal condition in order to ensure convergence at $t = 0$.

- Accuracy: both algorithms are designed to yield accurate approximations for (Y, Z) at $t = 0$ and duly do so. Notwithstanding, their accuracies significantly worsen for other points in time. In case of the Forward Deep BSDE scheme the quality of approximations quickly deteriorates for both the Y- and Z-processes, in fact the method only manages to yield reliable approximations at the initial point in time. The Backward Deep BSDE solver – with carefully chosen hyperparameters alleviating the impact of rolling errors – performs better at the approximation of Y_t for $t > 0$, however, it also fails to give control estimates of similar precision. In conclusion, both methods struggle with the estimation of Z which indeed is the biggest challenge in the numerical solution of BSDEs.

Chapter 5

Algorithmic Proposals

In the following chapter we describe the two algorithms proposed in this thesis, which we call the One-Step Malliavin (OSM) and Multi-Step Malliavin (MSM) separated Deep BSDE solvers. We highlight that the main goal of the algorithms is to overcome the drawbacks of the Backward Deep BSDE solver by giving accurate approximations for the Z-process throughout the whole time window for high-dimensional FBSDEs. Consequently, OSM and MSM are designed in a way that they separate the training phase of the Z- and Y-processes, yielding smaller learning problems and more regular losses. The control process is trained directly by the dynamics of the Malliavin derivative, which is enabled by the Malliavin chain rule formula and the differentiable function approximators given by neural networks. The proposed schemes only differ in their discretizations which we shall derive in section 5.4 and section 5.5 respectively. In what follows we focus on the derivation of the algorithms. Their error analysis shall be carried out in chapter 6, whereas we demonstrate their numerical performance in chapter 7.

This chapter is built up as follows. First, we extend the concept of LSMC and provide an alternative where, instead of ordinary least-squares, the regression is done with neural networks. Thereafter, we motivate the main idea behind the proposed algorithms and derive the continuous linear BSDE dynamics driving the evolution of the Z-process. So that we can avoid the explicit dependence on the Malliavin derivatives $D_{t_n}Y$ and $D_{t_n}Z$ we exploit the fact that – provided by the assumption of Markovianity – these processes can be estimated by the Malliavin chain rule. Subsequently, we discretize the resulting continuous equations given in Equation 5.12 and Equation 5.11 by separate theta-schemes and gather fully implementable recursive discrete approximations in Equation 5.15 and Equation 5.22 for both OSM and MSM. Afterwards, we discuss the details of the neural network regressions, and give out the precise formulations of both algorithms. Finally, to conclude the chapter, we compare the resulting algorithms to other discrete schemes founded on Malliavin calculus and qualitatively analyze the differences between OSM, MSM and the Backward Deep BSDE method.

5.1 Preliminaries

Throughout the whole chapter we are considering the dynamics of an FBSDE system and its Malliavin derivatives given by the following set of equations for any $0 \leq s \leq t \leq T$

$$X_t = x_0 + \int_0^t \mu(s, X_s)\mathrm{d}W_s + \int_0^t \sigma(s, X_s)\mathrm{d}W_s, \tag{5.1a}$$

$$Y_t = g(X_T) + \int_t^T f(s, X_s, Y_s, Z_s)\mathrm{d}s - \int_t^T Z_s\mathrm{d}W_s, \tag{5.1b}$$

$$D_sX_t = \sigma(s, X_s) + \int_s^t \nabla_x\mu\left(r, X_r\right) D_sX_r\mathrm{d}r + \int_s^t \nabla_x\sigma\left(r, X_r\right) D_sX_r\mathrm{d}W_r, \tag{5.1c}$$

$$\begin{aligned} D_sY_t = \nabla_x g(X_T)D_sX_T + \int_t^T & [\nabla_x f(r, X_r, Y_r, Z_r)D_sX_r + \nabla_y f(r, X_r, Y_r, Z_r)D_sY_r \\ & +\nabla_z f(r, X_r, Y_r, Z_r)D_sZ_r]\,\mathrm{d}r - \int_t^T D_sZ_r\mathrm{d}W_r, \end{aligned} \tag{5.1d}$$

with $D_sX_t = 0, D_sY_t = 0, D_sZ_t = 0$ when $0 \leq t < s \leq T$. Accordingly, we assume that the conditions of Assumption 2.5.1 are satisfied from which – by Theorem 2.4.1 and Theorem 2.5.1 – it follows that there are two unique triples of random processes $(X, Y, Z), (D_sX, D_sY, D_sZ)$ satisfying the equations above for each $0 \leq s \leq T$. We recall that, provided by Theorem 2.5.1, we define the control process as the version of the Malliavin derivative satisfying Equation 5.1d, therefore $Z_t = D_t Y_t$ for any $0 \leq t \leq T$.

Similarly as in the previous chapter, in what follows we denote a discretization of the underlying time horizon by

$$\pi^N := \{0 = t_0 < t_1 < \cdots < t_{N-1} < t_N = T\}, \tag{5.2}$$

and its mesh size by $|\pi| := \sup_{i=1,\ldots,N} t_i - t_{i-1}$.

In order to ease the notation, we put $X_n^\pi, Y_n^\pi, Z_n^\pi$, $D_m X_n^\pi, D_m Y_n^\pi, D_m Z_n^\pi$ for the discrete approximations of their continuous counterparts $X_{t_n}, Y_{t_n}, Z_{t_n}$, $D_{t_m} X_{t_n}, D_{t_m} Y_{t_n}, D_{t_m} Z_{t_n}$ for each $0 \leq m, n \leq N$.

5.2 Neural Network Regression

Before we start to explain the algorithms proposed in this thesis, let us first take a short – seemingly unrelated – detour and introduce the concept of neural network Least-Squares Monte Carlo method, where the regression in LSMC is performed by training neural networks. In the context of this work, this is motivated by the following three arguments. First, neural networks – provided by the Universal Approximation Theorem in Theorem 3.2.2 – span a wide function class and therefore provide a good basis for regression. Additionally, when choosing sufficiently differentiable activation functions, a neural network is a continuously differentiable function approximator whose derivatives can efficiently be calculated by automatic differentiation. As we shall see – we refer to Equation 5.6 in particular –, this observation is essential in terms of the estimations of the Malliavin derivatives $D_s Y, D_s Z$ through the Malliavin chain rule. Finally, the optimization of neural networks scale linearly in the number of input dimensions and there is encouraging empirical evidence that they could be capable of overcoming the curse of dimensionality. In the literature there exist a wide – and rapidly growing – range of applications of neural network regressions in the context of LSMC, out of which we mention a few [47], [52], [53] or [54].

Recalling the notation from subsection 4.3.1, let us now formulate this idea explicitly. Instead of using a finite set of basis functions $\{\phi_i\}_{i=1}^K$ as in Equation 4.18, we parametrize a Markovian conditional expectation by a deep neural network $\mathcal{NN}(\cdot|\Theta)$ of the form Equation 3.14. Recall that by the hierarchical composition, neural networks themselves can be interpreted as *non-linear decompositions of affine basis functions*. Using this parametrization, we then approximate the expected quadratic loss estimated by a sample mean of independent realizations

$$\mathbb{E}\left[|\Psi - \mathcal{NN}(X|\Theta)|^2\right] \approx \frac{1}{M}\sum_{j=1}^{M} |\Psi_j - \mathcal{NN}(X_j|\Theta)|^2. \tag{5.3}$$

The empirical loss function can be minimized by a stochastic optimization method such as Stochastic Gradient Descent in algorithm 1 or Adam in algorithm 2, which subsequently give an estimation for the optimal *weights* $\widehat{\Theta}$.

Although neural networks have proven to be a good, *universal* basis for the regression phase, they also have an important downside compared to Ordinary Least-Squares regression introduced in subsection 4.3.1 . In fact, in case of neural network regression we do not have a closed-form expression for the true minimizer of the expression in Equation 5.3. Consequently, due to the nature of optimization algorithms, in practice the estimator $\widehat{\Theta}$ is never the true minimizer but rather a close approximation of it.

5.3 Description of the Main Idea

Let us now turn back to the numerical solution of the FBSDE system in Equation 5.1 and describe the main idea behind the One-Step Malliavin and Multi-Step Malliavin algorithms proposed in this work. We divide the discussion in two parts. First, referring to the Deep BSDE methods explained in the previous chapter, we identify the area in which they could be improved, and subsequently motivate a new separated scheme which is focused on what Deep BSDE solvers struggle the most with: the approximation of the Z-process. Thereafter, building on this observation, we exploit that the natural dynamics of the control process is driven by the linear BSDE of the Malliavin derivative in Equation 5.1d and propose control approximations which take this phenomenon into account. We derive the equations for the continuous dynamics of both the Y- and Z-processes, from which the forthcoming discrete schemes shall straightforwardly follow in the next section.

5.3.1 Motivation

As we have seen in the previous chapter, although Deep BSDE solvers have shown remarkable successes in solving high-dimensional FBSDEs, they are also not without drawbacks – see subsection 4.4.3 in particular. In fact, we have mentioned that the Forward Deep BSDE method fails to give accurate approximations for the trajectories of Y and Z, and it only manages to solve the BSDE at $t = 0$. Moreover, we have also covered that its backward alternative exhibits a similar behaviour, and although it yields approximations of higher accuracy for the trajectories of the Y-process, its precision quickly deteriorates for Z_t as $t > 0$. One reason behind this latter phenomenon is that – unlike in recursive conditional expectation schemes such as the Euler scheme for BSDEs in Equation 4.13 – the Y- and Z-processes are fit *simultaneously* within the same optimization cycle, and the Z-process is only trained implicitly through the loss of the Backward Deep BSDE method given in Equation 4.24. The algorithms proposed in this work address these latter two observations.

In fact, OSM and MSM are motivated by the Backward Deep BSDE method and are designed in such a way that they separate the approximations of the Y- and Z-processes, and estimate their trajectories by solutions of distinct LSMC regressions. This way, we obtain a formulation closer to the classical recursive conditional expectation schemes for BSDEs, as the Euler in Equation 4.13 or theta-schemes in Equation 4.16. Consequently, it is expected that through the aforementioned approach, the resulting discrete schemes yield smaller learning problems and more regular loss functions, which are less prone to the effect of rolling errors and can provide more robust approximations for both Y and Z throughout the whole time window.

Moreover, OSM and MSM are also intended to mitigate the problem of Deep BSDE solvers with respect to the estimations of the control process. This is done by formulating the evolution of the Z-process through its natural dynamics provided by the linear BSDE of the Malliavin derivatives in Equation 5.1d. Thereafter, both algorithms impose direct training on the control process by formulating a loss function on a discretization of Equation 5.1d. It is expected that in such a way more accurate control trajectories can be obtained, as Z is not merely trained implicitly through the dynamics of Y but rather through the equation driving its own trajectories.

5.3.2 Formulation

Motivated by the qualitative arguments listed above, let us now formulate the main idea in mathematical terms. Under the assumptions of Theorem 2.5.1, we have that the linear BSDE driving the Malliavin derivatives of the solutions

$$\begin{aligned} D_sY_t = \nabla_x g(X_T)D_sX_T + \int_t^T &[\nabla_x f(r, X_r, Y_r, Z_r)D_sX_r + \nabla_y f(r, X_r, Y_r, Z_r)D_sY_r \\ &+\nabla_z f(r, X_r, Y_r, Z_r)D_sZ_r]\,\mathrm{d}r - \int_t^T D_sZ_r\mathrm{d}W_r, \quad 0 \le s \le t \le T, \end{aligned} \tag{5.4}$$

with $D_sY_t = 0, D_sZ_t = 0$ for $s < t$, has a unique solution pair (D_sY, D_sZ). In particular, exploiting the fact that by Theorem 2.5.1 the linear BSDE above gives a version for the Malliavin derivative for which $Z_s = D_sY_s$, we get

$$\begin{aligned} Z_s = \nabla_x g(X_T)D_sX_T + \int_s^T &[\nabla_x f(r, X_r, Y_r, Z_r)D_sX_r + \nabla_y f(r, X_r, Y_r, Z_r)D_sY_r \\ &+\nabla_z f(r, X_r, Y_r, Z_r)D_sZ_r]\,\mathrm{d}r - \int_s^T D_sZ_r\mathrm{d}W_r. \end{aligned} \tag{5.5}$$

Now, due to Markovianity – see subsection 1.5.1 in particular – we know that the solutions at time t are deterministic functions of the realization of the forward diffusion $Y_t = u(t, X_t), Z_t = v(t, X_t)$, for some deterministic functions $u : [0, T] \times \mathbb{R}^d \to \mathbb{R}$ and $v : [0, T] \times \mathbb{R}^d \to \mathbb{R}^d$. In fact, taking advantage of the connection between PDEs and BSDEs established by the general Feynman–Kac relations (Theorem 1.5.2) we also know that the control process is related to the spatial derivative of u, namely it is given by $Z_t = v(t, X_t) = \sigma(t, X_t)\nabla_x u(t, X_t)$. Therefore, using the Malliavin chain rule provided by Lemma 2.3.1, we conclude that the Malliavin derivatives (D_sY, D_sZ) of the solutions of the BSDE can be approximated at each point in time by the expressions $D_sY_t = \nabla_x u(t, X_t)D_sX_t$ and $D_sZ_t = \nabla_x v(t, X_t)D_sX_t$. Alternatively, by the results on variational processes provided by Lemma 2.5.1, we can write

$$D_sY_t = \nabla_x Y_t^{(t,x)} D_sX_t, \quad D_sZ_t = \nabla_x Z_t^{(t,x)} D_sX_t. \tag{5.6}$$

In what follows motivated by the Markovianity of the considered processes, when it is clear by the context we suppress the superscripts above and denote $\nabla_x Y_r = \nabla_x Y_r^{(r,x)}$, $\nabla_x Z_r = \nabla_x Z_r^{(r,x)}$ respectively. Substituting these observations back into Equation 5.5, for any $0 \leq s \leq T$ we get

$$Z_s = \nabla g(X_T) D_s X_T + \int_s^T \left[\nabla_x f(r, X_r, Y_r, Z_r) + \nabla_y f(r, X_r, Y_r, Z_r)\nabla_x Y_r \right. \\ \left. + \nabla_z f(r, X_r, Y_r, Z_r)\nabla_x Z_r\right] D_s X_r \mathrm{d}r - \int_s^T \nabla_x Z_r D_s X_r \mathrm{d}W_r, \quad (5.7)$$

on top of the dynamics of the original BSDE

$$Y_s = g(X_T) + \int_s^T f(r, X_r, Y_r, Z_r)\mathrm{d}r - \int_s^T Z_r \mathrm{d}W_r. \quad (5.8)$$

In particular, considering a discrete time partition, we have that for each $t_n \in \pi^N$ the following set of equations holds

$$Z_{t_n} = \nabla g(X_T) D_{t_n} X_T + \int_{t_n}^T \left[\nabla_x f(r, X_r, Y_r, Z_r) + \nabla_y f(r, X_r, Y_r, Z_r)\nabla_x Y_r \right. \quad (5.9a) \\ \left. + \nabla_z f(r, X_r, Y_r, Z_r)\nabla_x Z_r\right] D_{t_n} X_r \mathrm{d}r - \int_{t_n}^T \nabla_x Z_r D_{t_n} X_r \mathrm{d}W_r,$$

$$Y_{t_n} = g(X_T) + \int_{t_n}^T f(r, X_r, Y_r, Z_r)\mathrm{d}r - \int_{t_n}^T Z_r \mathrm{d}W_r. \quad (5.9b)$$

Equivalently, combining the dynamics of two adjacent time steps $t_n, t_{n+1} \in \pi^N$ and using the relations in Equation 5.6 yield

$$Z_{t_n} = \nabla_x Y_{t_{n+1}} D_{t_n} X_{t_{n+1}} + \int_{t_n}^{t_{n+1}} \left[\nabla_x f(r, X_r, Y_r, Z_r) + \nabla_y f(r, X_r, Y_r, Z_r)\nabla_x Y_r \right. \quad (5.10a) \\ \left. + \nabla_z f(r, X_r, Y_r, Z_r)\nabla_x Z_r\right] D_{t_n} X_r \mathrm{d}r - \int_{t_n}^{t_{n+1}} \nabla_x Z_r D_{t_n} X_r \mathrm{d}W_r,$$

$$Y_{t_n} = Y_{t_{n+1}} + \int_{t_n}^{t_{n+1}} f(r, X_r, Y_r, Z_r)\mathrm{d}r - \int_{t_n}^{t_{n+1}} Z_r \mathrm{d}W_r, \quad (5.10b)$$

subject to the terminal conditions $Z_{t_N} = \sigma(t_N, X_{t_N}) \nabla_x g(X_{t_N}), Y_{t_N} = g(X_{t_N})$. Consequently, taking conditional expectations with respect to $\mathcal{F}_{t_n}$ we arrive at the representations

$$Z_{t_n} = \mathbb{E}\left[\nabla g(X_T) D_{t_n} X_T + \int_{t_n}^T \left[\nabla_x f(r, X_r, Y_r, Z_r) + \nabla_y f(r, X_r, Y_r, Z_r)\nabla_x Y_r \right.\right. \quad (5.11a) \\ \left.\left. + \nabla_z f(r, X_r, Y_r, Z_r)\nabla_x Z_r\right] D_{t_n} X_r \mathrm{d}r | \mathcal{F}_{t_n}\right],$$

$$Y_{t_n} = \mathbb{E}\left[g(X_T) + \int_{t_n}^T f(r, X_r, Y_r, Z_r)\mathrm{d}r \middle| \mathcal{F}_{t_n}\right], \quad (5.11b)$$

and

$$Z_{t_n} = \mathbb{E}\left[\nabla_x Y_{t_{n+1}} D_{t_n} X_{t_{n+1}} + \int_{t_n}^{t_{n+1}} \left[\nabla_x f(r, X_r, Y_r, Z_r) + \nabla_y f(r, X_r, Y_r, Z_r)\nabla_x Y_r \right.\right. \quad (5.12a) \\ \left.\left. + \nabla_z f(r, X_r, Y_r, Z_r)\nabla_x Z_r\right] D_{t_n} X_r \mathrm{d}r | \mathcal{F}_{t_n}\right],$$

$$Y_{t_n} = \mathbb{E}\left[Y_{t_{n+1}} + \int_{t_n}^{t_{n+1}} f(r, X_r, Y_r, Z_r)\mathrm{d}r \middle| \mathcal{F}_{t_n}\right] \quad (5.12b)$$

respectively, subject to the terminal conditions $Z_{t_N} = \sigma(t_N, X_{t_N}) \nabla_x g(X_{t_N})$ and $Y_{t_N} = g(X_{t_N})$. These continuous equations form the basis of the discrete schemes proposed in MSM and OSM. For the upcoming sections, in order to make the presentation more tractable, let us introduce the following notation

$$f^D(r, X_r, Y_r, Z_r) := \nabla_x f(r, X_r, Y_r, Z_r) + \nabla_y f(r, X_r, Y_r, Z_r)\nabla_x Y_r + \nabla_z f(r, X_r, Y_r, Z_r)\nabla_x Z_r. \quad (5.13)$$

Finally, if (Y_{t_n}, Z_{t_n}) are parametrized by a pair of neural networks $(\mathcal{Y}_n(\cdot|\theta_n^y), \mathcal{Z}_n(\cdot|\theta_n^z))$ for each $t_n \in \pi^N$ then we do not merely gather estimations for the processes themselves but also differentiable function approximators whose derivatives can be made arbitrarily accurate as well – provided by the universal approximation capability in Theorem 3.2.2. Furthermore, we can efficiently calculate these derivatives through automatic differentiation and collect approximations for the spatial gradients of the networks $(\nabla_x \mathcal{Y}_n(\cdot|\theta_n^y), \nabla_x \mathcal{Z}_n(\cdot|\theta_n^z))$ accordingly. Thereafter, we can substitute these approximations back into discretized versions of the continuous dynamics in Equation 5.10 or Equation 5.9 and approximate each time step's Y- and Z-processes by separate neural network LSMC problems.

In conclusion, the idea of the proposed algorithms OSM and MSM can be summarized as follows. We develop a recursive conditional expectation scheme, where the control process Z is approximated by the evolution of the Malliavin derivative. In order to be able to approximate the Malliavin derivatives of the solution of the BSDE $(D_{t_n}Y_{t_j}, D_{t_n}Z_{t_j})$ for each time step $0 \le t_n < t_j \le T$, we take advantage of the Malliavin chain rule formula provided by Lemma 2.3.1 and exploit the universal approximation capability of neural networks. In the following sections, we explain the concrete discretization procedures which make the continuous dynamics in Equation 5.12 and Equation 5.11 fully implementable.

5.4 One-Step Malliavin (OSM) Scheme

We start by presenting the exact formulation of OSM. This algorithm, as its name already suggests, is built on dynamics in Equation 5.12 which describe the evolution of the Z- and Y-processes in between two adjacent time steps in a finite time partition π^N. In order to arrive at a completely discrete scheme, we need to approximate the continuous time integrals arising in the arguments of the conditional expectations in Equation 5.12a and Equation 5.12b. For the One-Step Malliavin algorithm this is done by a theta-discretization scheme similar to Equation 4.16. However, there is one significant difference compared to the theta-scheme presented in the previous chapter which is related to the separated conditional expectations formulated on different BSDEs[1] in Equation 5.12. In fact, for a separated conditional expectation scheme – which approximates each time step's Z_{t_n} first and then plugs that estimation into the conditional expectation of Y_{t_n} – we cannot allow for implicitness in the approximation of the Z-process, since Y_{t_n} is not available at the time of the regression. Hence, in order to avoid dependency of the approximation of Z_{t_n} on Y_{t_n}, we approximate the time integral in Equation 5.12 between t_n and t_{n+1} by the right-rectangle rule. Consequently, we gather the following discrete time approximations for the continuous dynamics in Equation 5.12 with some $\vartheta_y \in [0,1]$

$$Z_{t_n} \approx \mathbb{E}\left[\nabla_x Y_{t_{n+1}} D_{t_n} X_{t_{n+1}} + \Delta t_n f^D(t_{n+1}, X_{t_{n+1}}, Y_{t_{n+1}}, Z_{t_{n+1}}) D_{t_n} X_{t_{n+1}} \middle| \mathcal{F}_{t_n}\right], \tag{5.14a}$$
$$Y_{t_n} \approx \Delta t_n \vartheta_y f(t_n, X_{t_n}, Y_{t_n}, Z_{t_n}) + \mathbb{E}\left[Y_{t_{n+1}} + \Delta t_n (1-\vartheta_y) f(t_{n+1}, X_{t_{n+1}}, Y_{t_{n+1}}, Z_{t_{n+1}}) \middle| \mathcal{F}_{t_n}\right]. \tag{5.14b}$$

We remark that the resulting estimation is actually only "theta" in the discretization of the Y-process and is rather "explicit Euler" in the Z-process.

By now, we have gathered all the prerequisites to formulate the One-Step Malliavin approach proposed in this work. The algorithm starts off by simulating trajectories of the underlying forward diffusion $\{X_n^\pi\}_{0\le n\le N}$, and all of its Malliavin derivatives $\{D_m X_n^\pi\}_{0\le m\le n\le N}$ through the Euler–Maruyama scheme presented in Equation 4.6 and Equation 4.8. Then – just like in the Euler or theta-schemes for BSDEs – we can collect the terminal states X_N^π of the simulations, from which we gather the approximations for the terminal states of the solutions $Y_N^\pi = g(X_N^\pi)$ and $Z_N^\pi = \sigma(t_N, X_N^\pi)\nabla_x g(X_N^\pi)$. Here, the latter equation is given by the generalized Feynman–Kac relations in Theorem 1.5.2. Since the terminal condition is a given deterministic (and by Assumption 2.5.1 also differentiable) function of the state of the forward process, we also have analytical formulas for $\nabla_x Y_N^\pi \coloneqq \nabla_x g(X_N^\pi)$ and $\nabla_x Z_N^\pi \coloneqq \nabla_x \circ (\sigma \nabla_x g)(t_N, X_N^\pi)$, which can be plugged in the discrete dynamics in Equation 5.14. Thereafter, one can repeat the same procedure for all previous steps in time and arrive at the following recursive discrete scheme for each $n = N-1, \dots, 0$

$$Y_N^\pi = g(X_N^\pi), \quad Z_N^\pi = \sigma(t_N, X_N^\pi)\nabla_x g(X_N^\pi), \tag{5.15a}$$
$$Z_n^\pi = \mathbb{E}\left[\nabla_x Y_{n+1}^\pi D_n X_{n+1}^\pi + \Delta t_n f^D(t_{n+1}, X_{n+1}^\pi, Y_{n+1}^\pi, Z_{n+1}^\pi) D_n X_{n+1}^\pi | \mathcal{F}_{t_n}\right], \tag{5.15b}$$
$$Y_n^\pi = \Delta t_n \vartheta_y f(t_n, X_n^\pi, Y_n^\pi, Z_n^\pi) + \mathbb{E}\left[Y_{n+1}^\pi + \Delta t_n (1-\vartheta_y) f(t_{n+1}, X_{n+1}^\pi, Y_{n+1}^\pi, Z_{n+1}^\pi) | \mathcal{F}_{t_n}\right]. \tag{5.15c}$$

Notice that the scheme is explicit in Z and implicit in Y for any $\vartheta_y \in (0,1]$.

[1] i.e. the original one in Equation 5.1b and the linear one of its Malliavin derivatives in Equation 5.1d.

5.4.1 Neural Network Regression

Ultimately, in order to derive a fully implementable discrete scheme, let us explain the regression phase of the algorithm, which is done through neural network LSMC introduced in section 5.2. In order to simplify the presentation, let us introduce the following auxiliary discrete processes which correspond to the arguments of the conditional expectations in Equation 5.15.

$$\Psi^\pi_{n,n+1} \coloneqq \nabla_x Y^\pi_{n+1} D_n X^\pi_{n+1} + \Delta t_n f^D(t_{n+1}, X^\pi_{n+1}, Y^\pi_{n+1}, Z^\pi_{n+1}) D_n X^\pi_{n+1}, \tag{5.16a}$$

$$\Upsilon^\pi_{n,n+1} \coloneqq Y^\pi_{n+1} + \Delta t_n (1-\vartheta_y) f(t_{n+1}, X^\pi_{n+1}, Y^\pi_{n+1}, Z^\pi_{n+1}). \tag{5.16b}$$

We subsequently get $Z^\pi_n \equiv \mathbb{E}\left[\Psi^\pi_{n,n+1} \middle| X^\pi_n\right]$ and $Y^\pi_n \equiv \Delta t_n \vartheta_y f(t_n, X^\pi_n, Y^\pi_n, Z^\pi_n) + \mathbb{E}\left[\Upsilon^\pi_{n,n+1} \middle| X^\pi_n\right]$. Hence, motivated by section 5.2, we can thus parametrize the conditional expectations by fully-connected, feed-forward neural networks $\mathcal{Y}(\cdot|\theta^y_n) \coloneqq \mathcal{NN}(\cdot|\theta^y_n) : \mathbb{R}^d \to \mathbb{R}$ and $\mathcal{Z}(\cdot|\theta^z_n) \coloneqq \mathcal{NN}(\cdot|\theta^z_n) : \mathbb{R}^d \to \mathbb{R}^d$ of the form Equation 3.14. Moreover, we can define the quadratic loss functions

$$\mathcal{L}^z(\theta^z_n) \coloneqq \mathbb{E}\left[\left|\Psi^\pi_{n,n+1} - \mathcal{Z}(X^\pi_n|\theta^z_n)\right|^2\right], \tag{5.17a}$$

$$\mathcal{L}^y(\theta^y_n) \coloneqq \mathbb{E}\left[\left|\Upsilon^\pi_{n,n+1} - [\mathcal{Y}(X^\pi_n|\theta^y_n) - \Delta t_n \vartheta_y f(t_n, X^\pi_n, \mathcal{Y}(X^\pi_n|\theta^y_n), Z^\pi_n)]\right|^2\right], \tag{5.17b}$$

which measure the mean squared errors of the approximations. Consequently, argued by Proposition 4.3.1, we know that the true minimizer of the losses above over the space of all measurable functions is the conditional expectation itself. Therefore, if the function basis provided by the corresponding feedforward neural networks is wide enough, minimizing the expressions in Equation 5.17a and Equation 5.17b results in good approximations of the corresponding conditional expectations. The losses can be minimized separately by a stochastic optimization method such as SGD in algorithm 1 or Adam in algorithm 2. The resulting parameter sets are defined as

$$\widehat{\theta}^z_n \in \arg\min_\theta \mathcal{L}^z(\theta), \quad \widehat{\theta}^y_n \in \arg\min_\theta \mathcal{L}^y(\theta), \tag{5.18}$$

giving approximations $Y^\pi_n \coloneqq \mathcal{Y}\left(X^\pi_n|\widehat{\theta}^y_n\right)$, $\nabla_x Y^\pi_n \coloneqq \nabla_x \mathcal{Y}\left(X^\pi_n|\widehat{\theta}^y_n\right)$ and $Z^\pi_n \coloneqq \mathcal{Z}\left(X^\pi_n|\widehat{\theta}^z_n\right)$, $\nabla_x Z^\pi_n \coloneqq \nabla_x \mathcal{Z}\left(X^\pi_n|\widehat{\theta}^z_n\right)$ where the derivatives of the networks are calculated by automatic differentiation. Finally, exploiting the continuity of the estimated processes over time, in order to start the optimization *close* to the true solution, we use a transfer learning trick and initialize the parameters of the $n-1$'th time step according to

$$\theta^z_{n-1} \leftarrow \widehat{\theta}^z_n, \quad \theta^y_{n-1} \leftarrow \widehat{\theta}^y_n, \tag{5.19}$$

before proceeding with the optimization.

The complete algorithm with detailed steps is collected in algorithm 6. Furthermore, an illustration of its architecture at time step n is depicted in Figure 5.1.

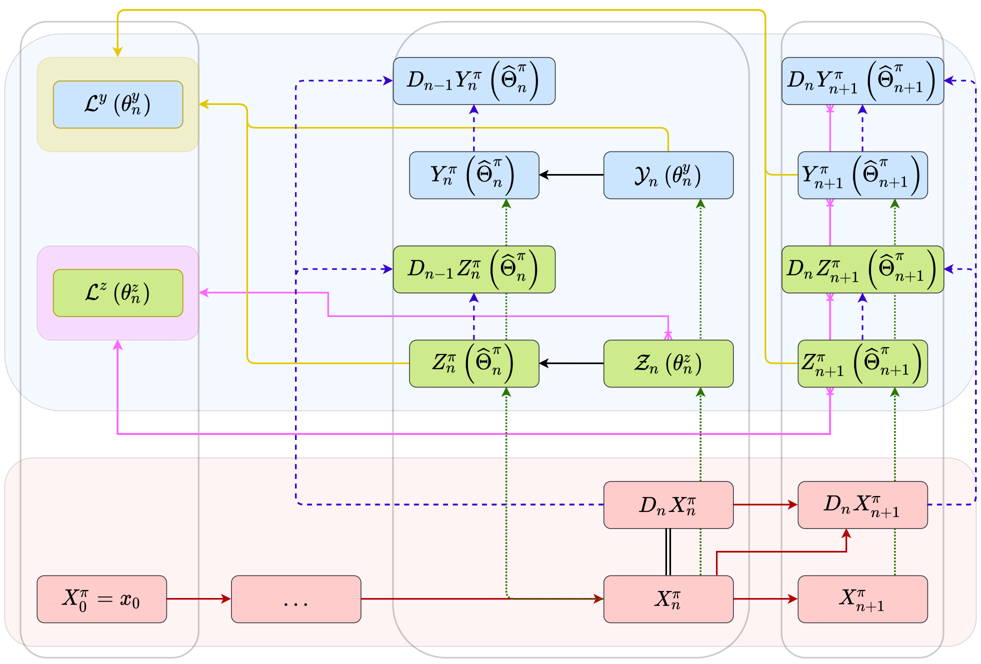
5.5 Multi-Step Malliavin (MSM) Scheme

Besides OSM, we propose an other discrete numerical algorithm which we call the Multi-Step Malliavin (MSM) scheme. The naming of this approach is self-explanatory as the only difference between OSM and MSM is captured by the fact that instead of considering the evolution of the Z- and Y-processes between adjacent time steps, we consider their dynamics until terminal time given in Equation 5.11. The reason for this is inspired by the theoretical results which imply that using multi-step discretization for the dynamics of BSDEs can help mitigating the interdependency of regression errors – see [4], [21] and [55] for more details. As we shall later see in chapter 6 – see the right-hand sides of Equation 6.76 and Equation 6.154 in particular –, this in fact can be proven for MSM as well for a special class of FBSDEs. The derivation of the algorithm is analogous to that of OSM, thus in order to avoid the repetition of arguments, we only highlight those aspects in which they differ.

By splitting up the integrals in between t_n and T to a sum of integrals in between adjacent time steps

Algorithm 6: One-Step Malliavin Algorithm (OSM)

```
Data: $\{X_n\}_{t_n\in\pi^N}$ – forward diffusion's Monte Carlo sample of size $M$
Data: $\{D_mX_n\}_{t_m,t_n\in\pi^N}$ – forward diffusion's Mallivain derivative's Monte Carlo
        sample of size $N\times M$
Input: $\vartheta_y\in[0,1]$ – choice of discretization parameter
Result: $(Y^\pi, Z^\pi)$ – BSDE's solution over time grid $\pi^N$
$Y^\pi_N\leftarrow g(X^\pi_N)$, $Z^\pi_N\leftarrow\sigma(t_N,X^\pi_N)\nabla_x g(X^\pi_N)$ – collect terminal condition
$\nabla_x Y^\pi_N\leftarrow\nabla_x g(X^\pi_N)$ ,$\nabla_x Z^\pi_N\leftarrow\nabla_x\circ(\sigma\nabla_x g)(t_N,X^\pi_N)$ – collect derivatives of terminal
 states
for $n=N-1,\dots,0$ do
    $\Psi^\pi_{n,n+1}$ – collect regression targets of $Z$ by Equation 5.16a
    $\mathcal{Y}(\cdot|\theta^y_n):=\mathcal{NN}(\cdot|\theta^y_n):\mathbb{R}^d\to\mathbb{R}$, $\mathcal{Z}(\cdot|\theta^z_n):=\mathcal{NN}(\cdot|\theta^z_n):\mathbb{R}^d\to\mathbb{R}^d$ – neural network
     parametrizations
    $\mathcal{L}^z(\theta^z_n)$ – define loss function according to Equation 5.17a
    $\widehat{\theta}^z_n\leftarrow\arg\min_\theta\mathcal{L}^z(\theta)$ – optimize empirical loss
    $Z^\pi_n\leftarrow\mathcal{Z}\left(X^\pi_n|\widehat{\theta}^z_n\right)$, $\nabla_x Z^\pi_n\leftarrow\nabla_x\mathcal{Z}\left(X^\pi_n|\widehat{\theta}^z_n\right)$ – gather approximations
    $\Upsilon^\pi_{n,n+1}$ – collect regression targets of $Y$ by Equation 5.16b
    $\mathcal{L}^y(\theta^y_n)$ – define loss function according to Equation 5.17b
    $\widehat{\theta}^y_n\leftarrow\arg\min_\theta\mathcal{L}^y(\theta)$ – optimize empirical loss
    $Y^\pi_n\leftarrow\mathcal{Y}\left(X^\pi_n|\widehat{\theta}^y_n\right)$, $\nabla_x Y^\pi_n\leftarrow\nabla_x\mathcal{Y}\left(X^\pi_n|\widehat{\theta}^y_n\right)$ – gather approximations
    $\theta^z_{n-1}\leftarrow\widehat{\theta}^z_n$, $\theta^y_{n-1}\leftarrow\widehat{\theta}^y_n$ – initialize parameters by transfer learning
end
```



$D_{n-1}Y_n^\pi\left(\widehat{\Theta}_n^\pi\right)$
$D_nY_{n+1}^\pi\left(\widehat{\Theta}_{n+1}^\pi\right)$
$\mathcal{L}^y\left(\theta_n^y\right)$
$Y_n^\pi\left(\widehat{\Theta}_n^\pi\right)$
$\mathcal{Y}_n\left(\theta_n^y\right)$
$Y_{n+1}^\pi\left(\widehat{\Theta}_{n+1}^\pi\right)$
$\mathcal{L}^z\left(\theta_n^z\right)$
$D_{n-1}Z_n^\pi\left(\widehat{\Theta}_n^\pi\right)$
$D_nZ_{n+1}^\pi\left(\widehat{\Theta}_{n+1}^\pi\right)$
$Z_n^\pi\left(\widehat{\Theta}_n^\pi\right)$
$\mathcal{Z}_n\left(\theta_n^z\right)$
$Z_{n+1}^\pi\left(\widehat{\Theta}_{n+1}^\pi\right)$
$D_nX_n^\pi$
$D_nX_{n+1}^\pi$
$X_0^\pi = x_0$
$\dots$
X_n^π
X_{n+1}^π

Figure 5.1: One-Step Malliavin (OSM) Architecture at Time Step n. Red area: forward part; blue area: backward part. The coding of the arrows is as follows. Solid-red: forward Euler SDE approximations; dotted-green: neural network approximation; dashed-purple: Malliavin chain rule through automatic differentiation; solid-yellow: loss of ϑ_y-discretization of the BSDE; solid-pink: loss of ϑ_z-discretization of the Malliavin BSDE.

$t_i, t_{i+1} \in \pi^N$, Equation 5.11 subsequently becomes

$$\begin{aligned}
Z_{t_n} &= \mathbb{E}\,[\nabla_x g(X_{t_N}) D_{t_n} X_{t_N} \\
&\quad + \sum_{i=n}^{N-1} \int_{t_i}^{t_{i+1}} [\nabla_x f(r, X_r, Y_r, Z_r) + \nabla_y f(r, X_r, Y_r, Z_r) \nabla_x Y_r \\
&\qquad + \nabla_z f(r, X_r, Y_r, Z_r) \nabla_x Z_r]\, D_{t_n} X_r \mathrm{d}r | \mathcal{F}_{t_n}], \qquad (5.20a) \\
Y_{t_n} &= \mathbb{E}\left[g(X_{t_N}) + \sum_{i=n}^{N-1} \int_{t_i}^{t_{i+1}} f(r, X_r, Y_r, Z_r) \mathrm{d}r \middle| \mathcal{F}_{t_n} \right]. \qquad (5.20b)
\end{aligned}$$

For a completely discrete, fully implementable scheme we need to use discrete approximations for the continuous time integrals in Equation 5.20. For the same reason as in the previous section, in order to avoid implicitness of the conditional expectations of Z, we approximate the first integral in the sum of Equation 5.20a by the right-rectangle rule. However, in order to keep the discussion as general as possible we propose a theta-discretization with $\vartheta_z \in [0, 1]$ for the rest of the integrals in the sum. In a similar fashion, we apply a theta-discretization with $\vartheta_y \in [0, 1]$ for all the integrals in the sum of Equation 5.20b. Consequently, we arrive at the following discrete approximations for the continuous dynamics in Equation 5.20

$$\begin{aligned}
Z_{t_n} &\approx \mathbb{E}\left[\nabla_x g(t_N, X_{t_N}) D_{t_n} X_{t_N} + \Delta t_n f^D(t_{n+1}, X_{t_{n+1}}, Y_{t_{n+1}}, Z_{t_{n+1}}) D_{t_n} X_{t_{n+1}} \right. \qquad (5.21a) \\
&\quad + \sum_{j=n+1}^{N-1} \Delta t_j \big(\vartheta_z f^D(t_j, X_{t_j}, Y_{t_j}, Z_{t_j}) D_{t_n} X_{t_j} \\
&\qquad \left. + (1-\vartheta_z) f^D(t_{j+1}, X_{t_{j+1}}, Y_{t_{j+1}}, Z_{t_{j+1}}) D_{t_n} X_{t_{j+1}} \big) | \mathcal{F}_{t_n} \right], \\
Y_{t_n} &\approx \Delta t_n \vartheta_y f(t_n, X_{t_n}, Y_{t_n}, Z_{t_n}) \qquad (5.21b) \\
&\quad + \mathbb{E}\Bigg[g(X_{t_N}) + \Delta t_n (1-\vartheta_y) f(t_{n+1}, X_{t_{n+1}}, Y_{t_{n+1}}, Z_{t_{n+1}}) \\
&\quad + \sum_{j=n+1}^{N-1} \Delta t_j \big(\vartheta_y f(t_j, X_{t_j}, Y_{t_j}, Z_{t_j}) + (1-\vartheta_y) f(t_{j+1}, X_{t_{j+1}}, Y_{t_{j+1}}, Z_{t_{j+1}}) \big) | \mathcal{F}_{t_n} \Bigg],
\end{aligned}$$

where we used the notation f^D defined in Equation 5.13.

Similarly, we can apply the same recursive approach as for OSM, collect the terminal conditions of a set of realizations for the underlying forward diffusion and its Malliavin derivatives, and plug those terminal states in the discrete approximations in Equation 5.21. Thereupon, we arrive at the following recursive discrete scheme for each $n = N-1, \ldots, 0$

$$Y_N^\pi = g(X_N^\pi), \quad Z_N^\pi = \sigma(t_N, X_N^\pi)\nabla_x g(X_N^\pi), \tag{5.22a}$$

$$\begin{aligned} Z_n^\pi = \mathbb{E}\big[&\nabla_x g(t_N, X_N^\pi) D_n X_N^\pi + \Delta t_n f^D(t_{n+1}, X_{n+1}^\pi, Y_{n+1}^\pi, Z_{n+1}^\pi) D_n X_{n+1}^\pi \\ &+ \sum_{j=n+1}^{N-1} \Delta t_j \big(\vartheta_z f^D(t_j, X_j^\pi, Y_j^\pi, Z_j^\pi) D_n X_j^\pi \\ &\quad + (1-\vartheta_z) f^D(t_{j+1}, X_{j+1}^\pi, Y_{j+1}^\pi, Z_{j+1}^\pi) D_n X_{j+1}^\pi\big) | \mathcal{F}_{t_n}\big], \end{aligned} \tag{5.22b}$$

$$\begin{aligned} Y_n^\pi = {}& \Delta t_n \vartheta_y f(t_n, X_n^\pi, Y_n^\pi, Z_n^\pi) \\ &+ \mathbb{E}\Big[g(X_N^\pi) + \Delta t_n (1-\vartheta_y) f(t_{n+1}, X_{n+1}^\pi, Y_{n+1}^\pi, Z_{n+1}^\pi) \\ &+ \sum_{j=n+1}^{N-1} \Delta t_j \left(\vartheta_y f(t_j, X_j^\pi, Y_j^\pi, Z_j^\pi) + (1-\vartheta_y) f(t_{j+1}, X_{j+1}^\pi, Y_{j+1}^\pi, Z_{j+1}^\pi)\right) | \mathcal{F}_{t_n}\Big]. \end{aligned} \tag{5.22c}$$

We remark that the proposed scheme is explicit in Z for any choice of $\vartheta_z \in [0,1]$ and implicit in Y for any choice of $\vartheta_y \in (0,1]$.

5.5.1 Neural Network Regression

Finally, in order to derive a fully implementable discrete scheme, we use a similar neural network regression as in case of OSM. We introduce the following auxiliary discrete processes which correspond to the arguments of the conditional expectations in Equation 5.22.

$$\begin{aligned} \Psi_{n,N}^\pi := {}& \nabla_x g(t_N, X_N^\pi) D_n X_N^\pi + \Delta t_n f^D(t_{n+1}, X_{n+1}^\pi, Y_{n+1}^\pi, Z_{n+1}^\pi) D_n X_{n+1}^\pi \\ &+ \sum_{j=n+1}^{N-1} \Delta t_j \big(\vartheta_z f^D(t_j, X_j^\pi, Y_j^\pi, Z_j^\pi) D_n X_j^\pi \\ &\quad + (1-\vartheta_z) f^D(t_{j+1}, X_{j+1}^\pi, Y_{j+1}^\pi, Z_{j+1}^\pi) D_n X_{j+1}^\pi\big), \end{aligned} \tag{5.23a}$$

$$\begin{aligned} \Upsilon_{n,N}^\pi := {}& g(X_N^\pi) + \Delta t_n (1-\vartheta_y) f(t_{n+1}, X_{n+1}^\pi, Y_{n+1}^\pi, Z_{n+1}^\pi) \\ &+ \sum_{j=n+1}^{N-1} \Delta t_j \left(\vartheta_y f(t_j, X_j^\pi, Y_j^\pi, Z_j^\pi) + (1-\vartheta_y) f(t_{j+1}, X_{j+1}^\pi, Y_{j+1}^\pi, Z_{j+1}^\pi)\right). \end{aligned} \tag{5.23b}$$

Thereupon, we get $Z_n^\pi = \mathbb{E}\left[\Psi_{n,N}^\pi | X_n^\pi\right], Y_n^\pi = \Delta t_n \vartheta_y f(t_n, X_n^\pi, Y_n^\pi, Z_n^\pi) + \mathbb{E}\left[\Upsilon_{n,N}^\pi | X_n^\pi\right]$. For the same reasons as seen in subsection 5.4.1, we parametrize these conditional expectations by fully-connected, feedforward neural networks $\mathcal{Y}(\cdot|\theta_n^y) := \mathcal{NN}(\cdot|\theta_n^y) : \mathbb{R}^d \to \mathbb{R}$ and $\mathcal{Z}(\cdot|\theta_n^z) := \mathcal{NN}(x|\theta_n^z) : \mathbb{R}^d \to \mathbb{R}^d$ of the form Equation 3.14. Furthermore, we define the following losses

$$\mathcal{L}^z(\theta_n^z) := \mathbb{E}\left[\left|\Psi_{n,N}^\pi - \mathcal{Z}(X_n^\pi|\theta_n^z)\right|^2\right], \tag{5.24a}$$

$$\mathcal{L}^y(\theta_n^y) := \mathbb{E}\left[\left|\Upsilon_{n,N}^\pi - [\mathcal{Y}(X_n^\pi|\theta_n^y) - \Delta t_n \vartheta_y f(t_n, X_n^\pi, \mathcal{Y}(X_n^\pi|\theta_n^y), Z_n^\pi)]\right|^2\right]. \tag{5.24b}$$

which measure the mean squared errors of the approximations. Now, provided that the neural networks span a wide enough function class, with the same arguments established by Proposition 4.3.1, we have that minimizing the expressions in Equation 5.24 results in close approximations of the true conditional expectations. The losses can in fact be minimized by a stochastic optimization method such as SGD in algorithm 1 or Adam in algorithm 2, resulting in estimations of the optimal parameter set defined by

$$\widehat{\theta}_n^z \in \arg\min_\theta \mathcal{L}^z(\theta), \quad \widehat{\theta}_n^y \in \arg\min_\theta \mathcal{L}^y(\theta). \tag{5.25}$$

These parameters subsequently give approximations $Y_n^\pi := \mathcal{Y}\left(X_n^\pi|\widehat{\theta}_n^y\right), \nabla_x Y_n^\pi := \nabla_x \mathcal{Y}\left(X_n^\pi|\widehat{\theta}_n^y\right)$ and $Z_n^\pi := \mathcal{Z}\left(X_n^\pi|\widehat{\theta}_n^z\right), \nabla_x Z_n^\pi := \nabla_x \mathcal{Z}\left(X_n^\pi|\widehat{\theta}_n^z\right)$, where the derivatives of the networks are calculated with automatic differentiation. Using the same transfer learning trick as before, we initialize the next time step's parameters according to the newly found optimal ones

$$\theta_{n-1}^z \leftarrow \widehat{\theta}_n^z, \quad \theta_{n-1}^y \leftarrow \widehat{\theta}_n^y, \tag{5.26}$$

and proceed with the optimization.

The complete algorithm with detailed steps is collected in algorithm 7. Furthermore, an illustration of its architecture at time step n is depicted in Figure 5.2.

Algorithm 7: Multi-Step Malliavin Algorithm (MSM)

```
Data: $\{X_n\}_{t_n\in\pi^N}$ – forward diffusion's Monte Carlo sample of size $M$
Data: $\{D_m X_n\}_{t_m,t_n\in\pi^N}$ – forward diffusion's Mallivain derivative's Monte Carlo
      sample of size $N \times M$
Input: $\vartheta_z, \vartheta_y \in [0,1]$ – choice of discretization parameters
Result: $(Y^\pi, Z^\pi)$ – BSDE's solution over time grid $\pi^N$
$Y_N^\pi \leftarrow g(X_N^\pi)$, $Z_N^\pi \leftarrow \sigma(t_N, X_N^\pi)\nabla_x g(X_N^\pi)$ – collect terminal condition
$\nabla_x Y_N^\pi \leftarrow \nabla_x g(X_N^\pi)$ ,$\nabla_x Z_N^\pi \leftarrow \nabla_x \circ (\sigma\nabla_x g)(t_N, X_N^\pi)$ – collect derivatives of terminal
  states
for $n = N-1, \dots, 0$ do
    $\Psi_{n,N}^\pi$ – collect regression targets of $Z$ by Equation 5.23a
    $\mathcal{Y}(\cdot|\theta_n^y) := \mathcal{NN}(\cdot|\theta_n^y) : \mathbb{R}^d \to \mathbb{R}$, $\mathcal{Z}(\cdot|\theta_n^z) := \mathcal{NN}(\cdot|\theta_n^z) : \mathbb{R}^d \to \mathbb{R}^d$ – neural network
     parametrizations
    $\mathcal{L}^z(\theta_n^z)$ – define loss function according to Equation 5.24a
    $\widehat{\theta}_n^z \leftarrow \arg\min_\theta \mathcal{L}^z(\theta)$ – optimize empirical loss
    $Z_n^\pi \leftarrow \mathcal{Z}\left(X_n^\pi|\widehat{\theta}_n^z\right)$, $\nabla_x Z_n^\pi \leftarrow \nabla_x \mathcal{Z}\left(X_n^\pi|\widehat{\theta}_n^z\right)$ – gather approximations
    $\Upsilon_{n,N}^\pi$ – collect regression targets of $Y$ by Equation 5.23b
    $\mathcal{L}^y(\theta_n^y)$ – define loss function according to Equation 5.24b
    $\widehat{\theta}_n^y \leftarrow \arg\min_\theta \mathcal{L}^y(\theta)$ – optimize empirical loss
    $Y_n^\pi \leftarrow \mathcal{Y}\left(X_n^\pi|\widehat{\theta}_n^y\right)$, $\nabla_x Y_n^\pi \leftarrow \nabla_x \mathcal{Y}\left(X_n^\pi|\widehat{\theta}_n^y\right)$ – gather approximations
    $\theta_{n-1}^z \leftarrow \widehat{\theta}_n^z$, $\theta_{n-1}^y \leftarrow \widehat{\theta}_n^y$ – initialize parameters by transfer learning
end
```

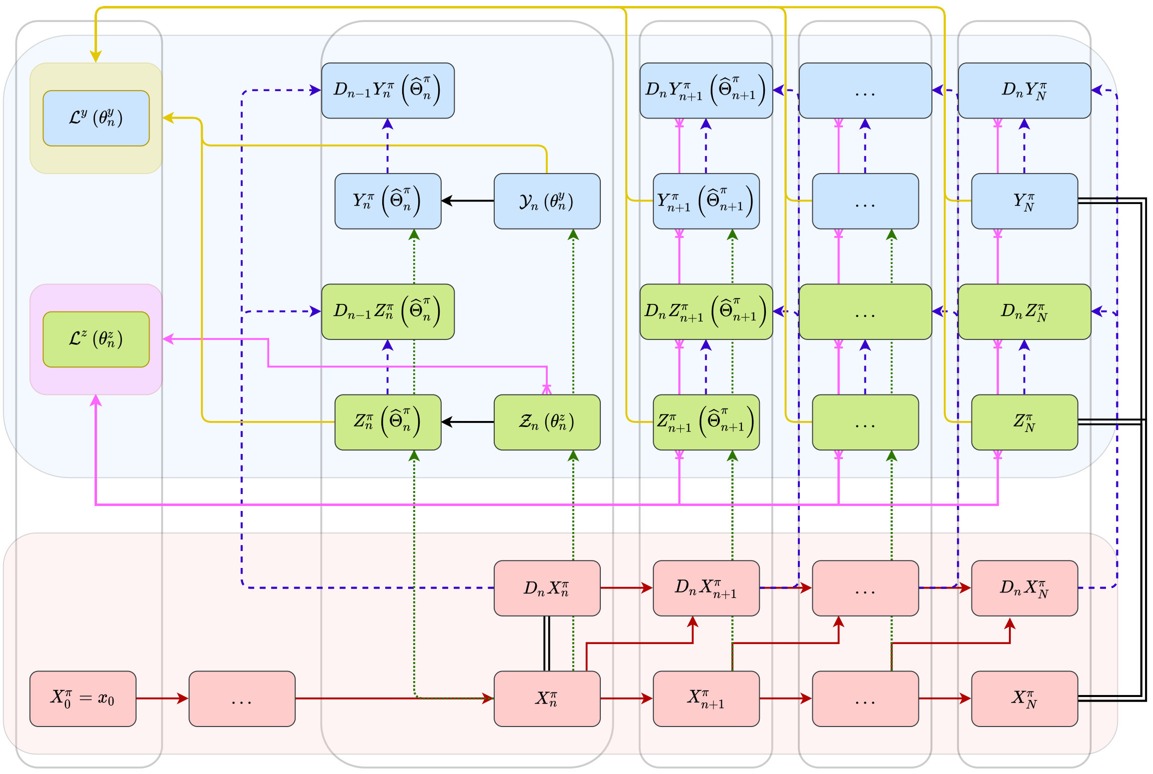
5.6 Qualitative Remarks

Ultimately, to conclude the introduction of the proposed algorithms, let us make three important qualitative remarks. The first and the third corresponding to the placement of OSM and MSM in the literature, whereas the second motivates the decision of explicit expressions in Equation 5.15, Equation 5.22 for Z.

5.6.1 Comparison with Deep BSDE Solvers

First and foremost, we need to have a few words on how the schemes proposed in Equation 5.15 and Equation 5.22 relate to the Deep BSDE solvers introduced in the previous chapter. It is clear that OSM and MSM were inspired by the Backward Deep BSDE method. In fact, they also start off from the terminal condition and perform an optimization at each previous step, recursively backwards in time. Nonetheless, there are three key differences which we shall not leave unmentioned.

Firstly, unlike the Backward Deep BSDE method, OSM and MSM separate the optimizations of the Z- and Y-processes, approximate the conditional expectations corresponding to the control process first and subsequently plug those estimations in the scheme of Y for each time step. Notice that – unlike in Equation 4.24 – the losses imposed on Y in Equation 5.17b and Equation 5.24b do not depend on the parametrization of the Z-process θ_n^z. In this regard, OSM and MSM are one step closer to classical



$\mathcal{L}^y(\theta_n^y)$
$\mathcal{L}^z(\theta_n^z)$
$D_{n-1}Y_n^\pi\left(\widehat{\Theta}_n^\pi\right)$
$Y_n^\pi\left(\widehat{\Theta}_n^\pi\right)$
$\mathcal{Y}_n(\theta_n^y)$
$D_{n-1}Z_n^\pi\left(\widehat{\Theta}_n^\pi\right)$
$Z_n^\pi\left(\widehat{\Theta}_n^\pi\right)$
$\mathcal{Z}_n(\theta_n^z)$
$D_nY_{n+1}^\pi\left(\widehat{\Theta}_{n+1}^\pi\right)$
$Y_{n+1}^\pi\left(\widehat{\Theta}_{n+1}^\pi\right)$
$D_nZ_{n+1}^\pi\left(\widehat{\Theta}_{n+1}^\pi\right)$
$Z_{n+1}^\pi\left(\widehat{\Theta}_{n+1}^\pi\right)$
$D_nY_N^\pi$
Y_N^π
$D_nZ_N^\pi$
Z_N^π
$D_nX_n^\pi$
$D_nX_{n+1}^\pi$
$D_nX_N^\pi$
$X_0^\pi = x_0$
X_n^π
X_{n+1}^π
X_N^π
$\dots$

Figure 5.2: Multi-Step Malliavin (MSM) Architecture at Time Step n. Red area: forward part; blue area: backward part. The coding of the arrows is as follows. Solid-red: forward Euler SDE approximations; dotted-green: neural network approximation; dashed-purple: Malliavin chain rule through automatic differentiation; solid-yellow: loss of ϑ_y-discretization of the BSDE; solid-pink: loss of ϑ_z-discretization of the Malliavin BSDE.

schemes such as the Euler in Equation 4.13 or the theta-schemes in Equation 4.16 which deploy a similar logic.

Secondly, as their names already suggest, OSM and MSM incorporate the dynamics of the Malliavin derivative's BSDE driving the control process, by which they apply direct training on the Z-process. It is expected that by considering the natural dynamics of the control, more accurate approximations could be gathered for Z throughout the whole time window. In the next two chapters we back this claim up with both theoretical error bounds – see Theorem 6.2.1 and Theorem 6.3.1 in particular – and empirical evidence in chapter 7.

Lastly, the hereby proposed algorithms also differ from the Backward Deep BSDE method in their discretization. In this regard, OSM and MSM are both more general than algorithm 5, as they apply a general theta-discretization on the BSDE in Equation 5.1b. Indeed, it can easily be seen that the discretization of Y in Equation 5.15 coincides with that of the Backward Deep BSDE method when $\vartheta_y = 1$.

5.6.2 Implicitness

The real reason why we allow for implicitness in the scheme of Y but not in the scheme of Z is related to the dimensionality and the Malliavin chain rule. We know that Y is a deterministic scalar function of the realization of X, and that Z is a deterministic d-dimensional vector-valued function of the realization of X at each point in time. Therefore, the differentiation of these functions is inherently different: the derivative of Y is a gradient, whereas the derivative of Z is a Jacobian matrix. Hence, if we allowed for implicitness in the scheme of Z, through the term of $D_n Z_n^\pi = \nabla_x Z_n^\pi D_n X_n^\pi$ during the training of the corresponding time step, the loss would depend on the Jacobian matrix of the parametrized function $Z_n^\pi = \mathcal{Z}\left(X_n^\pi | \theta_n^z\right)$. This means that with any stochastic gradient descent method built on the gradient of the objective function, while improving the model parameters in each iteration step, one would have dependency on the derivative of the Jacobian matrix – see Equation 3.13 in particular., However, that corresponds to the Hessian of a vector valued function which is an $\mathbb{R}^{d\times d\times d}$ tensor. Consequently, the optimization procedure would become computationally very intensive as the construction of such Hessians scales cubicly in the number of dimensions d. Since the goal of these algorithms is to tackle high-dimensional BSDE problems, we thus decided not to allow for implicitness in the estimations of the Z-process. On the other hand, when approximating the conditional expectations arising in Equation 5.15 and Equation 5.22 corresponding to Y, we have no dependence of the loss on the derivative of the parametrization of Y. Therefore, while optimizing the expressions in Equation 5.17b and Equation 5.24b, SGD algorithms only depend on the gradient of a scalar function whose construction is of linear cost. This explains why the recursive schemes of Y are allowed to be implicit.

5.6.3 Place in the Literature: Other Malliavin Algorithms

The One-Step Malliavin and Multi-Step Malliavin approaches are not the first algorithms in the literature exploiting the dynamics of the Malliavin derivative in Equation 5.1d. However, to the best of our knowledge, OSM and MSM are the first discrete formulations proposed directly on the linear BSDE in Equation 5.1d, whereas other schemes are built on certain representation formulas. In fact, Hu et. al in [22] developed a discrete scheme which is inspired by the fact that the BSDE of the Malliavin derivative in Equation 5.1d is linear and its solution admits to the representation formula in Equation 2.30 established by Proposition 2.6.1. Similarly, Turkedjiev in [21] proposed a numerical approach exploiting Equation 2.34 given by Proposition 2.6.2 – see also [55] for an error analysis. Finally, Briand and Labart in [56] proposed an algorithm which is founded on the *Wiener chaos decomposition* of random variables. Their discrete scheme uses the fact that $Z_t = D_t Y_t = D_t \mathbb{E}\left[Y_T + \int_t^T f(r, X_r, Y_r, Z_r)\mathrm{d}r \middle| \mathcal{F}_t\right]$ established by Proposition 2.3.1. Thereafter, they approximate the conditional expectations by chaos decomposition formulas and collect the Malliavin derivative according to the previous expression. These approaches are all built on representation formulas for the Malliavin derivative and – unlike OSM or MSM – are not designed to solve the linear BSDE in Equation 5.1d.

We emphasize that the reason why OSM and MSM can directly solve the linear BSDE of the Malliavin derivative is inherently linked to neural network regressions which themselves enable us to use the Malliavin chain rule for the approximations of $D_{t_n} Y$ and $D_{t_n} Z$ – see Equation 5.6 in particular. This observation justifies the choice of using neural networks in the regression phase.

Chapter 6

An Error Analysis

In the following chapter, we provide an error analysis for the algorithms proposed in the previous chapter. Our main goal is to show that the approximation errors induced by OSM in Equation 5.15 and MSM in Equation 5.22 are all of $\mathcal{O}(|\pi|)$ functions, where $|\pi|$ is the mesh-size of an equidistant time grid. We call such schemes which satisfy this requirement *consistent*. The chapter is structured as follows. First, we start by introducing some additional notations and classical preliminary results which shall ease the presentation. Afterwards, we prove the consistency of the One-Step Malliavin scheme proposed in algorithm 6. This proof is split up to different lemmas concerning the estimations of the Y- and Z-processes, which are stated under different conditions. In fact, it is important to highlight that the bounds for the Z-process are only proven under rather strict conditions collected in Assumption 6.1.2. Second, using some bounds of the one-step scheme, we extend the results to the Multi-Step Malliavin scheme proposed in algorithm 7 and show that one – by taking multiple time steps into account – gains an order of magnitude in the cumulative regression error of the Y-process. Thereafter, we compare the consistency results established by Theorem 6.2.1 and Theorem 6.3.1 to the error bounds proven for the Backward Deep BSDE methods. We highlight that due to the Malliavin problem formulation used in this work, our results bound a strictly stronger supremum norm than that of the standard Deep BSDE methods. Finally, concluding the chapter, we elaborate on the restrictions stated in Assumption 6.1.2 under which the theorems are presented and motivate why they had to be made.

6.1 Preliminaries

In order to enhance readability, we hereby collect the most important notations and preliminary results used throughout the chapter.

Definition 6.1.1 (Notations, spaces)**.** Let us introduce the following additional notations:

- we denote the mesh size of a time grid by $|\pi| := \sup_{t_i\in\pi^N} |t_{i+1} - t_i|$.
  It is worth to highlight that for an equidistant time partition this simply comes down to $|\pi| \equiv \Delta t_i =: T/N$;
- to ease the notation, we take advantage of the usual symbol $\mathbb{E}_i\left[\cdot\right] := \mathbb{E}\left[\cdot|\mathcal{F}_{t_i}\right]$.
  Keeping Markovianity in mind, this in fact means $\mathbb{E}_i^x\left[\cdot\right] \equiv \mathbb{E}\left[\cdot|X_{t_i} = x\right]$. However, since we only deal with deterministic initial conditions at $t = 0$, to avoid overly complex formulas we drop the superscript of the spatial part;
- argued by the Malliavin chain rule in Lemma 2.3.1, we denote for each $i \leq j$

  $$D_iY_j^\pi := \nabla_x Y_j^\pi D_iX_j^\pi, \quad D_iZ_j^\pi := \nabla_x Z_j^\pi D_iX_j^\pi \tag{6.1}$$

  for the approximations of the Malliavin derivatives.
- in the comparison against standard Backward Deep BSDE solvers we will need the concept of L^2-regularity of Z, which is defined as follows

  $$\varepsilon^Z(|\pi|) := \mathbb{E}\left[\sum_{n=0}^{N-1}\int_{t_i}^{t_{i+1}} \left|Z_t - \overline{Z}_{t_i}\right|^2 \mathrm{d}t\right], \quad \text{where} \quad \overline{Z}_{t_i} := \frac{1}{\Delta t_i}\mathbb{E}_i\left[\int_{t_i}^{t_{i+1}} Z_t \mathrm{d}t\right]. \tag{6.2}$$

Throughout the whole chapter it is assumed that the conditions of Assumption 2.5.1 for a unique solution both for the original and the Malliavin BSDEs are satisfied, leading to two unique triples of random processes $\{(X, Y, Z)\}_t$ and $\{(D_sX, D_sY, D_sZ)\}_{s,t}$. It is important to keep in mind that just as in chapter 2 we define the Z-process as the version of the Malliavin derivative which satisfies Theorem 2.5.1. In order to assure convergence of numerical schemes we make the following standard assumption on the continuity of the driver in time.

Assumption 6.1.1 (Unique Solution of Malliavin FBSDEs with Hölder Continuity)
Let the μ, σ, f, g in Equation 5.1 be such that the conditions of Assumption 1.5.1 are satisfied. Additionally, assume that

1. *μ, σ – on top of being Lipschitz continuous – are also $\frac{1}{2}$-Hölder continuous with Hölder constant L, i.e. $\forall (t_1, x_1), (t_2, x_2) \in [0, T] \times \mathbb{R}^d$*
$$|\mu(t_1, x_1) - \mu(t_2, x_2)| + |\sigma(t_1, x_1) - \sigma(t_2, x_2)| \leq L \left(|t_1 - t_2|^{1/2} + |x_1 - x_2| \right); \tag{6.3}$$
2. *f – on top of being Lipschitz continuous – is also $\frac{1}{2}$-Hölder continuous with Hölder constant L, i.e. $\forall (t_1, x_1, y_1, z_1), (t_2, x_2, y_2, z_2) \in [0, T] \times \mathbb{R}^d \times \mathbb{R} \times \mathbb{R}^{1\times d}$*
$$|f(t_1, x_1, y_1, z_1) - f(t_2, x_2, y_2, z_2)| \leq L \left(|t_1 - t_2|^{1/2} + |x_1 - x_2| + |y_1 - y_2| + |z_1 - z_2| \right). \tag{6.4}$$

We remark that these additional conditions are only required to assure the convergence of numerical methods and not for the well-posedness of the FBSDE problem. Nevertheless, in what follows, we shall state some results concerning the discretization of the Z-process under the following, stricter assumption.

Assumption 6.1.2 (Assumptions for the Bounds on the Z-process)
Let the conditions of Assumption 6.1.1 hold. Additionally, assume that

1. *the driver of the BSDE is independent of Z, i.e. $\nabla_z f \equiv 0$;*
2. *the underlying forward diffusion is an Arithmetic Brownian Motion given by Equation 1.10;*
3. *the first order derivatives of f are all $\frac{1}{2}$-Hölder continuous with universal Hölder constant L.*

Below, it is often used that for any continuously differentiable Lipschitz continuous function whose partial derivatives are uniformly bounded, the partial derivatives themselves are Lipschitz continuous as well. This is a direct consequence of the well-known Mean Value Theorem.

We remark, that in the upcoming proofs we – without warning – often use the Fubini theorem in addition to Leibniz's integral rule. It can easily be checked that the conditions of those theorems under Assumption 6.1.1 are indeed satisfied.

We frequently exploit the Kolmogorov continuity of the X- and Y-processes reading as follows
$$\mathbb{E}\left[|X_t - X_r|^2 \right] \leq C|t - r|, \quad \mathbb{E}\left[|Y_t - Y_r|^2 \right] \leq C|t - r|. \tag{6.5}$$

On top of the notations above, we make use of the following widely known results from the numerical analysis of SDEs.

Theorem 6.1.1 (Strong Convergence of the Euler–Maruyama Scheme for SDEs)
Let the conditions of Assumption 1.2.1 hold. Additionally, assume that μ and σ are $\frac{1}{2}$-Hölder continuous in time with constant L. Then
$$\max_{0\leq i\leq N-1} \mathbb{E}\left[\sup_{t\in[t_i, t_{i+1}]} |X_t - X_i^\pi|^2 \right] \leq C \left(1 + |x_0|^2 \right) |\pi|, \tag{6.6}$$
with some constant C independent from the time grid.
In fact, for a fixed initial condition, we have $C|\pi|$ on the right-hand side.

Proof. See, e.g., [12, Theorem 5.3.1]. □

Finally, we frequently rely on the following classical inequalities, which we state without proofs.

Proposition 6.1.1 (Young inequality)
Let $a, b \in \mathbb{R}$ and $\beta > 0$. Then the following line of inequalities holds
$$(1 - \beta)a^2 + (1 - 1/\beta)b^2 \leq (a + b)^2 \leq (1 + \beta)a^2 + (1 + 1/\beta)b^2. \tag{6.7}$$
In particular, we have $(a + b)^2 \leq 2(a^2 + b^2)$.

Proposition 6.1.2 (Discrete Grönwall lemma)
Let $\{p_n, q_n, r_n\}_n$ be positive sequences which satisfy for all $n \in \mathbb{N}$

$$p_n \leq (1 + q_n)p_{n+1} + r_n. \tag{6.8}$$

Then we have

$$\max_{0\leq i\leq N} p_i \leq \exp\left(\sum_{i=0}^{N-1} q_i\right)\left(p_N + \sum_{i=0}^{N-1} r_i\right). \tag{6.9}$$

In particular, if $\forall i : q_i = \mathcal{O}(1/N)$ this becomes

$$\max_{0\leq i\leq N} p_i \leq C\left(p_N + \sum_{i=0}^{N-1} r_i\right), \tag{6.10}$$

where C is independent of N.

6.2 Consistency of the One-Step Malliavin Scheme

We start proving the consistency of the proposed schemes by first giving bounds for the errors induced by the OSM scheme introduced in Equation 5.15. These bounds shall later be of good use in the corresponding proofs of the multi-step scheme. For the convenience of the reader, we split up the whole proof into smaller sub-lemmas, which we prove under different conditions. We glue these results together in the final theorem of the scheme stated in Theorem 6.2.1.

In the following, we use the following notations for the true targets of the regressions of Z_{t_i}, Y_{t_i}

$$V_i := \mathbb{E}_i\left[\nabla_x Y^\pi_{i+1} D_i X^\pi_{i+1} + \Delta t_i\left(\nabla_x f^\pi_{i+1} + \nabla_y f^\pi_{i+1}\nabla_x Y^\pi_{i+1} + \nabla_z f^\pi_{i+1}\nabla_x Z^\pi_{i+1}\right) D_i X^\pi_{i+1}\right], \tag{6.11}$$

$$U_i := \Delta t_i \vartheta_y f(t_i, X^\pi_i, U_i, Z^\pi_i) + \mathbb{E}_i\left[Y^\pi_{i+1} + \Delta t_i(1-\theta) f(t_{i+1}, X^\pi_{i+1}, Y^\pi_{i+1}, Z^\pi_{i+1})\right], \tag{6.12}$$

where we use the notation $\nabla_x f^\pi_{i+1} := \nabla_x f(t_{i+1}, X^\pi_{i+1}, Y^\pi_{i+1}, Z^\pi_{i+1})$ and similarly for the partial derivatives in y, z.

Due to Markovianity it is known that there exist deterministic functions $u_i : \mathbb{R}^d \to \mathbb{R}$ and $v_i : \mathbb{R}^d \to \mathbb{R}^d$ such that

$$U_i = u_i(X^\pi_i), \quad V^\pi_i = v_i(X^\pi_i). \tag{6.13}$$

We can parametrize the functions u_i and v_i by fully-connected, feedforward deep neural nets of the form $\mathcal{Y}(\cdot|\theta^y_i) := \mathcal{NN}(\cdot|\theta^y_i) : \mathbb{R}^d \to \mathbb{R}$ and $\mathcal{Z}(\cdot|\theta^z_i) := \mathcal{NN}(\cdot|\theta^z_i) : \mathbb{R}^d \to \mathbb{R}^d$. We define

$$\widehat{\theta}^z_i \in \arg\min_\eta \mathbb{E}\left[|v_i(X^\pi_i) - \mathcal{Z}(X^\pi_i|\eta)|^2\right], \tag{6.14}$$

$$\widehat{\theta}^y_i \in \arg\min_\eta \mathbb{E}\left[|u_i(X^\pi_i) - \mathcal{Y}(X^\pi_i|\eta)|^2\right], \tag{6.15}$$

giving the following estimations

$$Z^\pi_i := \mathcal{Z}(X^\pi_i|\widehat{\theta}^z_i), \quad Y^\pi_i := \mathcal{Y}(X^\pi_i|\widehat{\theta}^y_i). \tag{6.16}$$

Additionally, differentiating the resulting networks via automatic differentiation we take

$$\nabla_x Z^\pi_i := \nabla_x \mathcal{Z}(X^\pi_i|\widehat{\theta}^z_i), \quad \nabla_x Y^\pi_i := \nabla_x \mathcal{Y}(X^\pi_i|\widehat{\theta}^y_i). \tag{6.17}$$

Using all these notations, we define the regression errors of U_i, V_i by

$$\epsilon^z_i := \mathbb{E}\left[|v_i(X^\pi_i) - Z^\pi_i(X^\pi_i)|^2 + |\nabla_x v_i(X^\pi_i) - \nabla_x Z^\pi_i(X^\pi_i)|^2\right] \tag{6.18}$$

$$\epsilon^y_i := \mathbb{E}\left[|u_i(X^\pi_i) - Y^\pi_i(X^\pi_i)|^2 + |\nabla_x u_i(X^\pi_i) - \nabla_x Y^\pi_i(X^\pi_i)|^2\right]. \tag{6.19}$$

In the light of the Universal Approximation Theorem Theorem 3.2.2, these quantities can be made arbitrarily small even with shallow neural network approximations. The goal is to bound the following squared approximation error of the discrete, numerical scheme

$$\mathcal{E}\left[(Y^\pi, Z^\pi), (Y, Z)\right] := \max_{t_i\in\pi^N} \mathbb{E}\left[|Y_{t_i} - Y^\pi_i|^2\right] + \max_{t_j\in\pi^N} \mathbb{E}\left[|Z_{t_j} - Z^\pi_j|^2\right]. \tag{6.20}$$

With all these notations in power, we can now start giving bounds for the representation errors of quantities arising in the discrete scheme. First, a bound for the representation error of U_i in Equation 6.12 is proven. This lemma is similar to the one of Huré et. al in [9, Theorem 4.1] for the the Backward Deep BSDE method, with the differences that hereby we allow for a theta-discretization of the time integrals, and that the Z-process is approximated by a separate regression problem.

Lemma 6.2.1 (Representation Error of U_i – OSM)
Under the general conditions in Assumption 6.1.1, for sufficiently small equidistant time grids, the representation error of the Y-targets defined in Equation 6.12 satisfies for each $t_i \in \pi^N$

$$\mathbb{E}\left[|Y_{t_i} - U_i|^2\right] \le \left(1 + C\left(\vartheta_y^2 + (1-\vartheta_y)^2\right)\Delta t_i\right)\mathbb{E}\left[\left|Y_{t_{i+1}} - Y_{i+1}^\pi\right|^2\right] \\ + C(1-\vartheta_y)^2\Delta t_i\mathbb{E}\left[\left|Z_{t_{i+1}} - Z_{i+1}^\pi\right|^2\right] + C\vartheta_y^2\Delta t_i\mathbb{E}\left[|Z_{t_i} - Z_i^\pi|^2\right] + C|\pi|^2, \quad (6.21)$$

where C is a constant, independent of π^N.

Proof. In the following C always denotes a constant independent of the time partition, whose value may vary from line to line. By the dynamics of the BSDE and the definition of U_i we have

$$Y_{t_i} - U_i = \mathbb{E}_i\left[Y_{t_{i+1}} - Y_{i+1}^\pi\right] \\ + \vartheta_y\mathbb{E}_i\left[\int_{t_i}^{t_{i+1}} f(r, X_r, Y_r, Z_r) - f(t_i, X_i^\pi, U_i, Z_i^\pi)\mathrm{d}r\right] \\ + (1-\vartheta_y)\mathbb{E}_i\left[\int_{t_i}^{t_{i+1}} f(r, X_r, Y_r, Z_r) - f(t_{i+1}, X_{i+1}^\pi, Y_{i+1}^\pi, Z_{i+1}^\pi)\mathrm{d}r\right]. \quad (6.22)$$

Applying the Young inequality of the form $(a+b)^2 \le (1+\beta)a^2 + (1+1/\beta)b^2, \beta > 0$ yields

$$|Y_{t_i} - U_i|^2 \le (1+\beta)\left(\mathbb{E}_i\left[Y_{t_{i+1}} - Y_{i+1}^\pi\right]\right)^2 \\ + (1+1/\beta)\Bigg(\vartheta_y\mathbb{E}_i\left[\int_{t_i}^{t_{i+1}} f(r, X_r, Y_r, Z_r) - f(t_i, X_i^\pi, U_i, Z_i^\pi)\mathrm{d}r\right] \\ +(1-\vartheta_y)\mathbb{E}_i\left[\int_{t_i}^{t_{i+1}} f(r, X_r, Y_r, Z_r) - f(t_{i+1}, X_{i+1}^\pi, Y_{i+1}^\pi, Z_{i+1}^\pi)\mathrm{d}r\right]\Bigg)^2. \quad (6.23)$$

Using $(a+b)^2 \le a^2 + b^2$ together with the Jensen inequality gives

$$|Y_{t_i} - U_i|^2 \le (1+\beta)\mathbb{E}_i\left[\left|Y_{t_{i+1}} - Y_{i+1}^\pi\right|^2\right] \\ + 2(1+1/\beta)\vartheta_y^2\mathbb{E}_i\left[\left(\int_{t_i}^{t_{i+1}} f(r, X_r, Y_r, Z_r) - f(t_i, X_i^\pi, U_i, Z_i^\pi)\mathrm{d}r\right)^2\right] \\ + 2(1+1/\beta)(1-\vartheta_y)^2\mathbb{E}_i\left[\left(\int_{t_i}^{t_{i+1}} f(r, X_r, Y_r, Z_r) - f(t_{i+1}, X_{i+1}^\pi, Y_{i+1}^\pi, Z_{i+1}^\pi)\mathrm{d}r\right)^2\right]. \quad (6.24)$$

Let us now turn to the second term of the right-hand side above. Because of the Lipschitz and Hölder continuities of the driver we have that

$$|f(r, X_r, Y_r, Z_r) - f(t_i, X_i^\pi, U_i, Z_i^\pi)| \le L\left(|r - t_i|^{1/2} + |X_r - X_i^\pi| + |Y_r - U_i| + |Z_r - Z_i^\pi|\right), \quad (6.25)$$

which readily yields

$$\int_{t_i}^{t_{i+1}} f(r, X_r, Y_r, Z_r) - f(t_i, X_i^\pi, U_i, Z_i^\pi)\mathrm{d}r \\ \le L\int_{t_i}^{t_{i+1}} |r - t_i|^{1/2} + |X_r - X_i^\pi| + |Y_r - U_i| + |Z_r - Z_i^\pi|\mathrm{d}r. \quad (6.26)$$

Since $(a+b+c+d)^2 \leq 4(a^2+b^2+c^2+d^2)$, we get

$$\begin{aligned}
&\mathbb{E}_i\left[\left(\int_{t_i}^{t_{i+1}} f(r, X_r, Y_r, Z_r) - f(t_i, X_i^\pi, U_i, Z_i^\pi)\mathrm{d}r\right)^2\right] \\
&\qquad \leq 4L^2\left(\Delta t_i^3 + \mathbb{E}_i\left[\left(\int_{t_i}^{t_{i+1}} |X_r - X_i^\pi|\mathrm{d}r\right)^2\right]\right. \\
&\qquad\qquad \left. +\mathbb{E}_i\left[\left(\int_{t_i}^{t_{i+1}} |Y_r - U_i|\mathrm{d}r\right)^2\right] + \mathbb{E}_i\left[\left(\int_{t_i}^{t_{i+1}} |Z_r - Z_i^\pi|\mathrm{d}r\right)^2\right]\right). \qquad (6.27)
\end{aligned}$$

Taking expectations of the equation above, using the total law of probability and the linearity of the expectation operator, we subsequently gather

$$\begin{aligned}
&\mathbb{E}\left[\mathbb{E}_i\left[\left(\int_{t_i}^{t_{i+1}} f(r, X_r, Y_r, Z_r) - f(t_i, X_i^\pi, U_i, Z_i^\pi)\mathrm{d}r\right)^2\right]\right] \\
&\qquad \leq 4L^2\left(\Delta t_i^3 + \mathbb{E}\left[\left(\int_{t_i}^{t_{i+1}} |X_r - X_i^\pi|\mathrm{d}r\right)^2\right]\right. \\
&\qquad\qquad \left. +\mathbb{E}\left[\left(\int_{t_i}^{t_{i+1}} |Y_r - U_i|\mathrm{d}r\right)^2\right] + \mathbb{E}\left[\left(\int_{t_i}^{t_{i+1}} |Z_r - Z_i^\pi|\mathrm{d}r\right)^2\right]\right). \qquad (6.28)
\end{aligned}$$

Now, the Cauchy-Schwartz inequality for L^2-spaces implies that the integral terms on the right-hand side admit to

$$\begin{aligned}
\mathbb{E}\left[\left(\int_{t_i}^{t_{i+1}} |X_r - X_i^\pi|\mathrm{d}r\right)^2\right] &\leq \mathbb{E}\left[\int_{t_i}^{t_{i+1}} |X_r - X_i^\pi|^2\mathrm{d}r\right]\mathbb{E}\left[\int_{t_i}^{t_{i+1}} |1|^2\mathrm{d}r\right] \qquad (6.29)\\
&= \Delta t_i \mathbb{E}\left[\int_{t_i}^{t_{i+1}} |X_r - X_i^\pi|^2\mathrm{d}r\right], \qquad (6.30)
\end{aligned}$$

and similarly for the last two terms. This means that the upper bound can be loosened to

$$\begin{aligned}
&\mathbb{E}\left[\mathbb{E}_i\left[\left(\int_{t_i}^{t_{i+1}} f(r, X_r, Y_r, Z_r) - f(t_i, X_i^\pi, U_i, Z_i^\pi)\mathrm{d}r\right)^2\right]\right] \\
&\qquad \leq 4L^2\Delta t_i\left(\Delta t_i^2 + \mathbb{E}\left[\int_{t_i}^{t_{i+1}} |X_r - X_i^\pi|^2\mathrm{d}r\right]\right. \\
&\qquad\qquad \left. +\mathbb{E}\left[\int_{t_i}^{t_{i+1}} |Y_r - U_i|^2\mathrm{d}r\right] + \mathbb{E}\left[\int_{t_i}^{t_{i+1}} |Z_r - Z_i^\pi|^2\mathrm{d}r\right]\right). \qquad (6.31)
\end{aligned}$$

Identical steps for the third term in Equation 6.24 lead to

$$\begin{aligned}
&\mathbb{E}\left[\mathbb{E}_i\left[\left(\int_{t_i}^{t_{i+1}} f(r, X_r, Y_r, Z_r) - f(t_{i+1}, X_{i+1}^\pi, Y_{i+1}^\pi, Z_{i+1}^\pi)\mathrm{d}r\right)^2\right]\right] \\
&\qquad \leq 4L^2\Delta t_i\left(\Delta t_i^2 + \mathbb{E}\left[\int_{t_i}^{t_{i+1}} \left|X_r - X_{i+1}^\pi\right|^2\mathrm{d}r\right]\right. \\
&\qquad\qquad \left. +\mathbb{E}\left[\int_{t_i}^{t_{i+1}} \left|Y_r - Y_{i+1}^\pi\right|^2\mathrm{d}r\right] + \mathbb{E}\left[\int_{t_i}^{t_{i+1}} \left|Z_r - Z_{i+1}^\pi\right|^2\mathrm{d}r\right]\right). \qquad (6.32)
\end{aligned}$$

Taking expectations of Equation 6.24, and using the upper bounds Equation 6.31 and Equation 6.32

on the right-hand side, we can therefore collect

$$\begin{aligned}\mathbb{E}\left[|Y_{t_i}-U_i|^2\right] \leq (1+\beta)\mathbb{E}\left[\left|Y_{t_{i+1}}-Y^\pi_{i+1}\right|^2\right]&\\ +(1+1/\beta)\,\vartheta_y^2 8L^2\Delta t_i\left(\Delta t_i^2+\mathbb{E}\left[\int_{t_i}^{t_{i+1}}|X_r-X^\pi_i|^2\mathrm{d}r\right]\right.&\\ \left.+\mathbb{E}\left[\int_{t_i}^{t_{i+1}}|Y_r-U_i|^2\mathrm{d}r\right]+\mathbb{E}\left[\int_{t_i}^{t_{i+1}}|Z_r-Z^\pi_i|^2\mathrm{d}r\right]\right)&\\ +(1+1/\beta)\left(1-\vartheta_y\right)^2 8L^2\Delta t_i\left(\Delta t_i^2+\mathbb{E}\left[\int_{t_i}^{t_{i+1}}\left|X_r-X^\pi_{i+1}\right|^2\mathrm{d}r\right]\right.&\\ \left.+\mathbb{E}\left[\int_{t_i}^{t_{i+1}}\left|Y_r-Y^\pi_{i+1}\right|^2\mathrm{d}r\right]+\mathbb{E}\left[\int_{t_i}^{t_{i+1}}\left|Z_r-Z^\pi_{i+1}\right|^2\mathrm{d}r\right]\right).&\end{aligned} \tag{6.33}$$

Now, first of all notice that the Euler estimations of the forward equations give $\mathbb{E}\left[|X_r-X^\pi_i|^2\right] \leq \mathbb{E}\left[\sup_{r\in[t_i,t_{i+1}]}|X_r-X^\pi_i|^2\right] \leq C|\pi|$ by Equation 6.6 with some C independent of the time grid. Therefore, we get

$$\mathbb{E}\left[\int_{t_i}^{t_{i+1}}|X_r-X^\pi_i|^2\mathrm{d}r\right] \leq C|\pi|\Delta t_i, \tag{6.34}$$

$$\mathbb{E}\left[\int_{t_i}^{t_{i+1}}\left|X_r-X^\pi_{i+1}\right|^2\mathrm{d}r\right] \leq C|\pi|\Delta t_i. \tag{6.35}$$

Furthermore, we can rewrite the Y-terms using the Kolmogorov continuity of the Y-process

$$\mathbb{E}\left[\int_{t_i}^{t_{i+1}}|Y_r-U_i|^2\mathrm{d}r\right] \leq 2C|\pi|\Delta t_i+2\Delta t_i\mathbb{E}\left[|Y_{t_i}-U_i|^2\right]. \tag{6.36}$$

Analogously, for the $i+1$'th term we get

$$\mathbb{E}\left[\int_{t_i}^{t_{i+1}}\left|Y_r-Y^\pi_{i+1}\right|^2\mathrm{d}r\right] \leq 2C|\pi|\Delta t_i+2\Delta t_i\mathbb{E}\left[\left|Y_{t_{i+1}}-Y^\pi_{i+1}\right|^2\right]. \tag{6.37}$$

Additionally, building on the continuity result established by Theorem 2.5.2, we know that the Z-process itself is continuous admitting to the following inequality

$$\mathbb{E}\left[|Z_r-Z_t|^2\right] \leq C|r-t|. \tag{6.38}$$

With which, the Z terms can be upper bounded in the following way

$$\begin{aligned}\mathbb{E}\left[\int_{t_i}^{t_{i+1}}|Z_r-Z^\pi_i|^2\mathrm{d}r\right] &\leq C\int_{t_i}^{t_{i+1}}|r-t_i|\mathrm{d}r+2\Delta t_i\mathbb{E}\left[|Z_{t_i}-Z^\pi_i|^2\right] &(6.39)\\ &\leq C\Delta t_i^2+2\Delta t_i\mathbb{E}\left[|Z_{t_i}-Z^\pi_i|^2\right]. &(6.40)\end{aligned}$$

Furthermore, identically

$$\mathbb{E}\left[\int_{t_i}^{t_{i+1}}\left|Z_r-Z^\pi_{i+1}\right|^2\mathrm{d}r\right] \leq C\Delta t_i^2+2\Delta t_i\mathbb{E}\left[\left|Z_{t_{i+1}}-Z^\pi_{i+1}\right|^2\right]. \tag{6.41}$$

Plugging all these observations back into Equation 6.33 we get

$$\begin{aligned}\mathbb{E}\left[|Y_{t_i}-U_i|^2\right] \leq (1+\beta)\mathbb{E}\left[\left|Y_{t_{i+1}}-Y^\pi_{i+1}\right|^2\right]&\\ +8L^2\Delta t_i\left(1+1/\beta\right)\left[\vartheta_y^2\left(C|\pi|^2+2\mathbb{E}\left[|Y_{t_i}-U_i|^2\right]\Delta t_i+2\mathbb{E}\left[|Z_{t_i}-Z^\pi_i|^2\right]\Delta t_i\right)\right.&\\ \left.+\left(1-\vartheta_y\right)^2\left(C|\pi|^2+2\mathbb{E}\left[\left|Y_{t_{i+1}}-Y^\pi_{i+1}\right|^2\right]\Delta t_i+2\mathbb{E}\left[\left|Z_{t_{i+1}}-Z^\pi_{i+1}\right|^2\right]\Delta t_i\right)\right].&\end{aligned} \tag{6.42}$$

Let us now choose $\beta = \gamma \Delta t_i$ with some $\gamma > 0$. With this choice, for small enough time steps satisfying $\gamma \Delta t_i < 1$, the equation above becomes

$$\begin{aligned}\mathbb{E}\left[|Y_{t_i} - U_i|^2\right] \leq \left(1 + \left[1 + \frac{32L^2}{\gamma}(1-\vartheta_y)^2\right]\Delta t_i\right)\mathbb{E}\left[\left|Y_{t_{i+1}} - Y_{i+1}^\pi\right|^2\right] \\ + \frac{32L^2}{\gamma}(1-\vartheta_y)^2\Delta t_i \mathbb{E}\left[\left|Z_{t_{i+1}} - Z_{i+1}^\pi\right|^2\right] + \frac{32L^2}{\gamma}\vartheta_y^2 \Delta t_i \mathbb{E}\left[|Z_{t_i} - Z_i^\pi|^2\right] \\ + \frac{32L^2}{\gamma}\vartheta_y^2\Delta t_i \mathbb{E}\left[|Y_{t_i} - U_i|^2\right] + C|\pi|^2. \quad (6.43)\end{aligned}$$

Let us now choose $\gamma := 64L^2$. Since $\vartheta_y \in [0,1]$, we have that $\vartheta_y^2/2 < 1$ and therefore conclude[1]

$$\begin{aligned}\mathbb{E}\left[|Y_{t_i} - U_i|^2\right] \leq \left(1 + C\left(\vartheta_y^2 + (1-\vartheta_y)^2\right)\Delta t_i\right)\mathbb{E}\left[\left|Y_{t_{i+1}} - Y_{i+1}^\pi\right|^2\right] \\ + C(1-\vartheta_y)^2\Delta t_i \mathbb{E}\left[\left|Z_{t_{i+1}} - Z_{i+1}^\pi\right|^2\right] + C\vartheta_y^2\Delta t_i\mathbb{E}\left[|Z_{t_i} - Z_i^\pi|^2\right] + C|\pi|^2, \quad (6.44)\end{aligned}$$

which is what we needed to show, completing the proof. □

Remark 6.2.1 (Z-independence, Implicitness)
In order to conclude this result, let us make two important remarks.

1. *the coefficients of* $\mathbb{E}\left[|Z_{t_i} - Z_i^\pi|^2\right]$ *and* $\mathbb{E}\left[\left|Z_{t_{i+1}} - Z_{i+1}^\pi\right|^2\right]$ *imply that for a completely implicit or explicit scheme, the corresponding terms cancel out. On the other hand, for any choice of* $\vartheta_y \in [0,1]$ *the* Y_{i+1}^π *term remains to be present on the right-hand side, confirming the intuition of the BSDE dynamics;*

2. *the above lemma was stated under the general Assumption 6.1.1; in case the driver is independent of* Z *it is easy to see that both* Z *terms on the right-hand side can be dropped. This observation will be essential in the statement of the final consistency proof.*

Having given an upper bound for the representation error of U_i, we now proceed with giving a similar bound on the representation error of $\nabla_x U_i$, i.e. the spatial derivative of the Y-targets. This estimate will be a crucial input for the estimation error of the Z-process. It is important to highlight that the following result is only stated under the more restricted conditions of Assumption 6.1.2.

Lemma 6.2.2 (Representation Error of the Variation $\nabla_x Y_{t_i}$ – OSM)
Under Assumption 6.1.2, for sufficiently small equidistant time grids, the representation error of the derivative of the Y*-targets satisfies for each* $t_i \in \pi^N$

$$\begin{aligned}\mathbb{E}\left[|\nabla_x Y_{t_i} - \nabla_x U_i|^2\right] \leq \left(1 + C\left(\vartheta_y^2 + (1-\vartheta_y)^2\right)\Delta t_i\right)\mathbb{E}\left[\left|\nabla_x Y_{t_{i+1}} - \nabla_x Y_{i+1}^\pi\right|^2\right] \\ + C\vartheta_y^2\Delta t_i\mathbb{E}\left[|Y_{t_i} - U_i|^2\right] + C(1-\vartheta_y)^2\Delta t_i\mathbb{E}\left[\left|Y_{t_{i+1}} - Y_{i+1}^\pi\right|^2\right] + C\Delta t_i^2. \quad (6.45)\end{aligned}$$

where C *is a constant independent of* π^N.

Proof. In the following C always denotes a constant independent of the time partition, whose value may vary from line to line. In order to avoid repetition of previously seen arguments, we hereby only highlight those steps which significantly differ from the ones in the previous lemma.

By the dynamics of the variational process $\nabla_x Y_{t_i}$[2] Equation 2.26 and the definition of U_i Equa-

[1]This result easily follows by the Taylor expansions of $\frac{1+ax}{1-bx}$ and $\frac{ax}{1-bx}$ around $x = 0$.

[2]Here $\nabla_x Y_r \equiv \nabla_x Y_r^{(t_i,x)}$ naturally from the context. We drop the superscript corresponding to the initial conditions, in order to ease the notation.

tion 6.12, Leibniz's integral rule[3] gives[4]

$$\begin{aligned}\nabla_x Y_{t_i} - \nabla_x U_i \leq \mathbb{E}_i &\left[\nabla_x Y_{t_{i+1}} - \nabla_x Y^\pi_{i+1}\right] \\ &+ \vartheta_y \left(\mathbb{E}_i\left[\int_{t_i}^{t_{i+1}} \nabla_x f(r, X_r, Y_r)\nabla_x X_r - \nabla_x f(t_i, X_{t_i}, U_i)\nabla_x X_{t_i}\mathrm{d}r\right]\right. \\ &\left. + \mathbb{E}_i\left[\int_{t_i}^{t_{i+1}} \nabla_y f(r, X_r, Y_r)\nabla_x Y_r - \nabla_y f(t_i, X_{t_i}, U_i)^\tau \nabla_x U_i \mathrm{d}r\right]\right) \\ &+ (1-\vartheta_y)\left(\mathbb{E}_i\left[\int_{t_i}^{t_{i+1}} \nabla_x f(r, X_r, Y_r)\nabla_x X_r - \nabla_x f(t_{i+1}, X_{t_{i+1}}, Y^\pi_{i+1})\nabla_x X_{t_{i+1}}\mathrm{d}r\right]\right. \\ &\left. + \mathbb{E}_i\left[\int_{t_i}^{t_{i+1}} \nabla_y f(r, X_r, Y_r)\nabla_x Y_r - \nabla_y f(t_{i+1}, X_{t_{i+1}}, Y_{i+1}\pi)\nabla_x Y^\pi_{i+1}\mathrm{d}r\right]\right). \end{aligned} \tag{6.46}$$

Similar applications of the Young and Jensen inequalities as in Lemma 6.2.1 yield

$$\begin{aligned}|\nabla_x Y_{t_i} - \nabla_x U_i|^2 \leq (1+\beta)\mathbb{E}_i &\left[\left|\nabla_x Y_{t_{i+1}} - \nabla_x Y^\pi_{i+1}\right|^2\right] \\ &+ 2(1+1/\beta)\vartheta_y^2 \left(\mathbb{E}_i\left[\left|\int_{t_i}^{t_{i+1}} \nabla_x f(r, X_r, Y_r)\nabla_x X_r - \nabla_x f(t_i, X_{t_i}, U_i)\nabla_x X_{t_i}\mathrm{d}r\right|^2\right]\right. \\ &\left. + \mathbb{E}_i\left[\left|\int_{t_i}^{t_{i+1}} \nabla_y f(r, X_r, Y_r)\nabla_x Y_r - \nabla_y f(t_i, X_{t_i}, U_i)\nabla_x U_i \mathrm{d}r\right|^2\right]\right) \\ &+ 2(1+1/\beta)(1-\vartheta_y)^2 \\ \times &\left(\mathbb{E}_i\left[\left|\int_{t_i}^{t_{i+1}} \nabla_x f(r, X_r, Y_r)\nabla_x X_r - \nabla_x f(t_{i+1}, X_{t_{i+1}}, Y^\pi_{i+1})\nabla_x X_{t_{i+1}}\mathrm{d}r\right|^2\right]\right. \\ &\left. + \mathbb{E}_i\left[\left|\int_{t_i}^{t_{i+1}} \nabla_y f(r, X_r, Y_r)\nabla_x Y_r - \nabla_y f(t_{i+1}, X_{t_{i+1}}, Y^\pi_{i+1})\nabla_x Y^\pi_{i+1}\mathrm{d}r\right|^2\right]\right). \end{aligned} \tag{6.47}$$

Let us now turn to the integral terms above. The second term above in expectations leads to

$$\begin{aligned}\mathbb{E}&\left[\left|\int_{t_i}^{t_{i+1}} \nabla_x f(r, X_r, Y_r)\nabla_x X_r - \nabla_x f(t_i, X_{t_i}, U_i)\nabla_x X_{t_i}\mathrm{d}r\right|^2\right] \\ &\leq d\Delta t_i \mathbb{E}\left[\int_{t_i}^{t_{i+1}} |\nabla_x f(r, X_r, Y_r)\nabla_x X_r - \nabla_x f(t_i, X_{t_i}, U_i)\nabla_x X_{t_i}|^2 \mathrm{d}r\right], \end{aligned} \tag{6.48}$$

by the L^2 Cauchy-Schwartz inequality. Now, using the fact that under Assumption 6.1.2 X is an Arithmetic Brownian Motion whose flow – defined in Equation 2.12 – is constant I_d, the integrand is identically equal to

$$|\nabla_x f(r, X_r, Y_r)\nabla_x X_r - \nabla_x f(t_i, X_{t_i}, U_i)\nabla_x X_{t_i}|^2 = |\nabla_x f(r, X_r, Y_r) - \nabla_x f(t_i, X_{t_i}, U_i)|^2. \tag{6.49}$$

Since the driver is assumed to be twice continuously differentiable with uniformly bounded second derivatives, we also have by the Mean Value theorem, that the partial derivatives are also Lipschitz continuous. Additionally, under Assumption 6.1.2 the partial derivatives of the driver are also $\frac{1}{2}$-Hölder continuous from which it follows that

$$\begin{aligned}\mathbb{E}&\left[\left|\int_{t_i}^{t_{i+1}} \nabla_x f(r, X_r, Y_r)\nabla_x X_r - \nabla_x f(t_i, X_{t_i}, U_i)\nabla_x X_{t_i}\mathrm{d}r\right|^2\right] \\ &\leq 4dL^2 C\Delta t_i \mathbb{E}\left[\int_{t_i}^{t_{i+1}} |r-t_i|^{1/2} + |X_r - X_{t_i}|^2 + |Y_r - U_i|^2 \mathrm{d}r\right]. \end{aligned} \tag{6.50}$$

The Kolmogorov continuity of the forward and the Y-processes thus leads to

$$\mathbb{E}\left[\left(\int_{t_i}^{t_{i+1}} \nabla_x f(r, X_r, Y_r) - \nabla_x f(t_i, X_{t_i}, U_i)\mathrm{d}r\right)^2\right] \leq 4dL^2 C\Delta t_i\left(\Delta t_i^2 + \Delta t_i \mathbb{E}\left[|Y_{t_i} - U_i|^2\right]\right). \tag{6.51}$$

[3]It can easily be seen by the assumptions of standard parameters that the dominated convergence argument indeed holds.

[4]Here $\nabla_x X_r$ denotes the flow of the SDE defined in Equation 2.12, and in fact $\nabla_x X_r \equiv \nabla_x X_r^{(t_i,x)}$. We drop the superscript corresponding to the initial condition, in order to ease the notation.

Through identical steps, we also gather

$$\mathbb{E}\left[\left(\int_{t_i}^{t_{i+1}} \nabla_x f(r, X_r, Y_r) - \nabla_x f(t_{i+1}, X_{t_{i+1}}, Y_{i+1}^\pi)\mathrm{d}r\right)^2\right] \\ \leq 4dL^2 C\Delta t_i \left(\Delta t_i^2 + \Delta t_i \mathbb{E}\left[\left|Y_{t_i} - Y_{i+1}^\pi\right|^2\right]\right). \quad (6.52)$$

Let us now turn to the integral terms in Equation 6.47 containing partial derivatives with respect to y. First, notice that the integrands can be upper bounded in the following way

$$|\nabla_y f(r, X_r, Y_r)\nabla_x Y_r - \nabla_x f(t_i, X_{t_i}, U_i)\nabla_x U_i|^2 \\ \leq 2|\nabla_y f(r, X_r, Y_r) - \nabla_x f(t_i, X_{t_i}, U_i)|^2|\nabla_x Y_r|^2 + 2|\nabla_x f(t_i, X_{t_i}, U_i)|^2|\nabla_x Y_r - \nabla_x U_i|^2 \quad (6.53)$$

by the triangular inequality and the submultiplicative property of the Frobenius norm. Now, given by Lemma 2.5.1 – see Equation 2.29 in particular – we have that under the general conditions in Assumption 6.1.1 the variational process $\nabla_x Y_r$ is bounded, which leads to the following upper bound

$$\mathbb{E}\left[|\nabla_y f(r, X_r, Y_r) - \nabla_y f(t_i, X_{t_i}, U_i)|^2|\nabla_x Y_r|^2\right] \leq C\mathbb{E}\left[|\nabla_y f(r, X_r, Y_r) - \nabla_y f(t_i, X_{t_i}, U_i)|^2\right]. \quad (6.54)$$

At this point, using the uniform boundedness and Lipschitz continuity of $\nabla_y f(\cdot,\cdot,\cdot)$, on top of the continuity of the variational process given by Lemma 2.5.1, we conclude that the original integral term satisfies

$$\mathbb{E}\left[\left(\int_{t_i}^{t_{i+1}} \nabla_y f(r, X_r, Y_r)\nabla_x Y_r - \nabla_y f(t_i, X_{t_i}, U_i)\nabla_x U_i \mathrm{d}r\right)^2\right] \\ \leq 4L^2 dC\Delta t_i \left(\Delta t_i^2 + 2\Delta t_i \mathbb{E}\left[|Y_{t_i} - U_i|^2\right]\right) + 2L^2 d\Delta t_i \mathbb{E}\left[|\nabla_x Y_{t_i} - \nabla_x U_i|^2\right]. \quad (6.55)$$

Similarly, for the $i+1$'th term we have

$$\mathbb{E}\left[\left(\int_{t_i}^{t_{i+1}} \nabla_y f(r, X_r, Y_r)\nabla_x Y_r - \nabla_y f(t_{i+1}, X_{t_{i+1}}, Y_{i+1}^\pi)\nabla_x Y_{i+1}^\pi \mathrm{d}r\right)^2\right] \\ \leq 4L^2 dC\Delta t_i \left(\Delta t_i^2 + 2\Delta t_i \mathbb{E}\left[\left|Y_{t_{i+1}} - Y_{i+1}^\pi\right|^2\right]\right) + 2L^2 d\Delta t_i \mathbb{E}\left[\left|\nabla_x Y_{t_{i+1}} - \nabla_x Y_{i+1}^\pi\right|^2\right]. \quad (6.56)$$

Plugging Equation 6.51, Equation 6.52, Equation 6.55 and Equation 6.56 back into Equation 6.47 in expectations, choosing $\beta := \gamma\Delta t_i < 1$, we therefore get

$$\mathbb{E}\left[|\nabla_x Y_{t_i} - \nabla_x U_i|^2\right] \leq (1 + \gamma\Delta t_i)\mathbb{E}_i\left[\left|\nabla_x Y_{t_{i+1}} - \nabla_x Y_{i+1}^\pi\right|^2\right] \\ + \frac{16L^2 d}{\gamma}\left(C\vartheta_y^2\Delta t_i \mathbb{E}\left[|Y_{t_i} - U_i|^2\right] + C(1-\vartheta_y)^2\Delta t_i \mathbb{E}\left[\left|Y_{t_{i+1}} - Y_{i+1}^\pi\right|^2\right]\right. \\ \left. + \vartheta_y^2\Delta t_i \mathbb{E}\left[|\nabla_x Y_{t_i} - \nabla_x U_i|^2\right] + (1-\vartheta_y)^2\Delta t_i \mathbb{E}\left[\left|\nabla_x Y_{t_{i+1}} - \nabla_x Y_{i+1}^\pi\right|^2\right] + C\Delta t_i^2\right). \quad (6.57)$$

Now, if we choose $\gamma := 32L^2 d$, for time steps satisfying $\Delta t_i < \max\left[\frac{1}{32L^2 d}, 1\right]$ we conclude

$$\mathbb{E}\left[|\nabla_x Y_{t_i} - \nabla_x U_i|^2\right] \leq \left(1 + C\left(\vartheta_y^2 + (1-\vartheta_y)^2\right)\Delta t_i\right)\mathbb{E}\left[\left|\nabla_x Y_{t_{i+1}} - \nabla_x Y_{i+1}^\pi\right|^2\right] \\ + C\vartheta_y^2\Delta t_i \mathbb{E}\left[|Y_{t_i} - U_i|^2\right] + C(1-\vartheta_y)^2\Delta t_i \mathbb{E}\left[\left|Y_{t_{i+1}} - Y_{i+1}^\pi\right|^2\right] + C\Delta t_i^2. \quad (6.58)$$

This is what we needed to show, concluding the proof. □

Thanks to the Malliavin chain rule, the lemma above directly implies an approximation error bound for the Malliavin derivatives estimated through Equation 6.1, which is stated below.

Corollary 6.2.1 (Approximation Error of $D_{t_i}Y_{t_{i+1}}$ – OSM)
Under the assumptions of Lemma 6.2.2 we also have

$$\mathbb{E}\left[\left|D_{t_i}Y_{t_{i+1}} - D_i Y_{i+1}^\pi\right|^2\right] \leq C\left(1 + \left(\vartheta_y^2 + (1-\vartheta_y)^2\right)\Delta t_i\right)\mathbb{E}\left[\left|\nabla_x Y_{t_{i+1}} - \nabla_x Y_{i+1}^\pi\right|^2\right] \\ + C\vartheta_y^2\Delta t_i \mathbb{E}\left[|Y_{t_i} - U_i|^2\right] + C(1-\vartheta_y)^2\Delta t_i \mathbb{E}\left[\left|Y_{t_{i+1}} - Y_{i+1}^\pi\right|^2\right] + C\Delta t_i^2 + C\epsilon_i^y, \quad (6.59)$$

where C is a constant independent of the time grid.

Proof. In what follows C always denotes a constant, independent of the time partition whose value may vary from line to line.

By the Malliavin chain rule and the approximation defined in Equation 6.1, we get that the approximation error of $D_{t_i}Y_{t_{i+1}}$ admits to

$$D_{t_i}Y_{t_{i+1}} - D_iY^{\pi}_{i+1} = \nabla_x Y_{t_{i+1}} D_{t_i}X_{t_{i+1}} - \nabla_x Y^{\pi}_{i+1} D_i X^{\pi}_{i+1}. \tag{6.60}$$

For ABM forward dynamics the Malliavin derivative is analytically solvable and it is also constant according to Equation 2.9, which thus gives

$$\mathbb{E}\left[\left|D_{t_i}Y_{t_{i+1}} - D_iY^{\pi}_{i+1}\right|^2\right] \leq C\mathbb{E}\left[\left|\nabla_x Y_{t_{i+1}} - \nabla_x Y^{\pi}_{i+1}\right|^2\right]. \tag{6.61}$$

The result then follows from Lemma 6.2.2 and the definition of the regression error Equation 6.19. □

Having obtained an error estimate for the estimation of the Malliavin derivative through the Malliavin chain rule, let us now turn to the representation error of the Z-process.

Lemma 6.2.3 (Representation Error of Z_{t_i} – OSM)
Under Assumption 6.1.2, for sufficiently small equidistant time grids, the representation error of the Z-targets defined in Equation 6.11 satisfies for each $t_i \in \pi^N$

$$\mathbb{E}\left[\left|Z_{t_i} - V_i\right|^2\right] \;\leq\; (1 + C\Delta t_i)\,\mathbb{E}\left[\left|D_{t_i}Y_{t_{i+1}} - D_iY^{\pi}_{i+1}\right|^2\right] + C\Delta t_i^2 + C\Delta t_i\mathbb{E}\left[\left|Y_{t_{i+1}} - Y^{\pi}_{i+1}\right|^2\right], \tag{6.62}$$

where C is a constant independent of π^N.

Proof. In the following C always denotes a constant independent of the time partition, whose value may vary from line to line. In order to avoid the repetition of arguments, hereby we only highlight the steps, which significantly differ from those of the previous lemmas.

Using the Malliavin representation of the Z-process given by Theorem 2.5.1, we have the following inequality

$$\begin{aligned}Z_{t_i} - V_i = \mathbb{E}_i&\left[D_{t_i}Y_{t_{i+1}} - D_iY^{\pi}_{i+1}\right]\\ &+ \mathbb{E}_i\left[\int_{t_i}^{t_{i+1}} \nabla_x f(r, X_r, Y_r)D_{t_i}X_r - \nabla_x f(t_{i+1}, X_{t_{i+1}}, Y^{\pi}_{i+1})D_iX^{\pi}_{i+1}\mathrm{d}r\right]\\ &+ \mathbb{E}_i\left[\int_{t_i}^{t_{i+1}} \nabla_y f(r, X_r, Y_r)D_{t_i}Y_r - \nabla_y f(t_{i+1}, X_{t_{i+1}}, Y^{\pi}_{i+1})D_iY^{\pi}_{i+1}\mathrm{d}r\right]\\ &+ \mathbb{E}_i\left[\int_{t_i}^{t_{i+1}} \nabla_z f(r, X_r, Y_r)D_{t_i}Z_r - \nabla_z f(t_{i+1}, X_{t_{i+1}}, Y^{\pi}_{i+1})D_iZ^{\pi}_{i+1}\mathrm{d}r\right].\end{aligned} \tag{6.63}$$

Under Assumption 6.1.2, the driver is independent of Z and the forward process's Malliavin derivative is analytically solvable. These observations, together with the application of the Young- and Jensen inequalities with some $\alpha > 0$ as in Lemma 6.2.1 yield

$$\begin{aligned}\left|Z_{t_i} - V_i\right|^2 \leq (1+\alpha)\,&\mathbb{E}_i\left[\left|D_{t_i}Y_{t_{i+1}} - D_iY^{\pi}_{i+1}\right|^2\right]\\ &+ 4\,(1 + 1/\alpha)\left(\mathbb{E}_i\left[\left|\int_{t_i}^{t_{i+1}} \nabla_x f(r, X_r, Y_r)D_{t_i}X_r + \nabla_x f(t_{i+1}, X_{t_{i+1}}, Y^{\pi}_{i+1})D_{t_i}X_{t_{i+1}}\mathrm{d}r\right|^2\right]\right.\\ &\left.+\mathbb{E}_i\left[\left|\int_{t_i}^{t_{i+1}} \nabla_y f(r, X_r, Y_r)D_{t_i}Y_r + \nabla_y f(t_{i+1}, X_{t_{i+1}}, Y^{\pi}_{i+1})D_iY^{\pi}_{i+1}\mathrm{d}r\right|^2\right]\right).\end{aligned} \tag{6.64}$$

Let us first focus on the first term of the second term. Taking expectations gives

$$\begin{aligned}\mathbb{E}&\left[\left|\int_{t_i}^{t_{i+1}} \nabla_x f(r, X_r, Y_r)D_{t_i}X_r - \nabla_x f(t_{i+1}, X_{t_{i+1}}, Y^{\pi}_{i+1})D_{t_i}X_{t_{i+1}}\mathrm{d}r\right|^2\right]\\ &\leq d\Delta t_i\mathbb{E}\left[\int_{t_i}^{t_{i+1}} \left|\nabla_x f(r, X_r, Y_r)D_{t_i}X_r - \nabla_x f(t_{i+1}, X_{t_{i+1}}, Y^{\pi}_{i+1})D_{t_i}X_{t_{i+1}}\right|^2\mathrm{d}r\right],\end{aligned} \tag{6.65}$$

by the L^2 Cauchy–Schwartz inequality. The integrand can be rewritten as

$$\begin{aligned}\nabla_x f(r, X_r, Y_r) D_{t_i} X_r - \nabla_x f(t_{i+1}, X_{t_{i+1}}, Y^\pi_{i+1}) D_{t_i} X_{t_{i+1}} \\ = \left(\nabla_x f(r, X_r, Y_r) - \nabla_x f(t_{i+1}, X_{t_{i+1}}, Y^\pi_{i+1})\right) D_{t_i} X_r \\ + \nabla_x f(t_{i+1}, X_{t_{i+1}}, Y^\pi_{i+1})(D_{t_i} X_r - D_{t_i} X_{t_{i+1}}). \quad (6.66)\end{aligned}$$

In expectations, this means that

$$\begin{aligned}\mathbb{E}\left[\left|\nabla_x f(r, X_r, Y_r) D_{t_i} X_r - \nabla_x f(t_{i+1}, X_{t_{i+1}}, Y^\pi_{i+1}) D_{t_i} X_{t_{i+1}}\right|^2\right] \\ \leq 2L^2 C \mathbb{E}\left[\left|\nabla_x f(r, X_r, Y_r) - \nabla_x f(t_{i+1}, X_{t_{i+1}}, Y^\pi_{i+1})\right|^2\right] + 2L^2 C |t_{i+1} - r|, \quad (6.67)\end{aligned}$$

where we used that the Malliavin derivative of ABM – given by Equation 2.9 – is bounded and continuous in time. Therefore, using the standard Lipschitz continuity argument for $\nabla_x f(\cdot, \cdot, \cdot)$ we conclude that the integral term of X admits to

$$\begin{aligned}\mathbb{E}\left[\left(\int_{t_i}^{t_{i+1}} \nabla_x f(r, X_r, Y_r) D_{t_i} X_r - \nabla_x f(t_{i+1}, X_{t_{i+1}}, Y^\pi_{i+1}) D_{t_i} X_{t_{i+1}} \mathrm{d}r\right)^2\right] \\ \leq 4L^2 d C \Delta t_i \left(\Delta t_i^2 + \Delta t_i \mathbb{E}\left[\left|Y_{t_{i+1}} - Y^\pi_{i+1}\right|^2\right]\right). \quad (6.68)\end{aligned}$$

Let us now turn to the second term of the second term in Equation 6.64.

$$\begin{aligned}\mathbb{E}\left[\left(\int_{t_i}^{t_{i+1}} \nabla_y f_r D_{t_i} Y_r - \nabla_x f^\pi_{i+1} D_i Y^\pi_{i+1} \mathrm{d}r\right)^2\right] \\ \leq d \Delta t_i \mathbb{E}\left[\int_{t_i}^{t_{i+1}} \left|\nabla_x f_r D_{t_i} Y_r - \nabla_x f^\pi_{i+1} D_i Y^\pi_{i+1}\right|^2 \mathrm{d}r\right] \quad (6.69)\end{aligned}$$

by the Cauchy-Schwartz inequality. Applying similar tricks to the integrand as before, we get that

$$\begin{aligned}\left|\nabla_y f(r, X_r, Y_r) D_{t_i} Y_r - \nabla_x f(t_{i+1}, X_{t_{i+1}}, Y^\pi_{i+1}) D_i Y^\pi_{i+1}\right|^2 \\ \leq 4 \Big(\left|\nabla_y f(r, X_r, Y_r) - \nabla_y f(t_{i+1}, X_{t_{i+1}}, Y^\pi_{i+1})\right|^2 |D_{t_i} Y_r|^2 \\ + \left|\nabla_y f(t_{i+1}, X_{t_{i+1}}, Y^\pi_{i+1})\right|^2 \left|D_{t_i} Y_r - D_{t_i} Y_{t_{i+1}}\right|^2 \\ + \left|\nabla_y f(t_{i+1}, X_{t_{i+1}}, Y^\pi_{i+1})\right|^2 \left|D_{t_i} Y_{t_{i+1}} - D_i Y^\pi_{i+1}\right|^2 \Big). \quad (6.70)\end{aligned}$$

In expectations this comes down to

$$\begin{aligned}\mathbb{E}\left[\left|\nabla_y f(r, X_r, Y_r) D_{t_i} Y_r - \nabla_x f(t_{i+1}, X_{t_{i+1}}, Y^\pi_{i+1}) D_i Y^\pi_{i+1}\right|^2\right] \\ \leq 4C \Big(\mathbb{E}\left[\left|\nabla_y f(r, X_r, Y_r) - \nabla_y f(t_{i+1}, X_{t_{i+1}}, Y^\pi_{i+1})\right|^2\right] + L^2 |t_{i+1} - r| \\ + L^2 \mathbb{E}\left[\left|D_{t_i} Y_{t_{i+1}} - D_i Y^\pi_{i+1}\right|^2\right] \Big), \quad (6.71)\end{aligned}$$

as the partial derivatives are uniformly bounded by L; the Malliavin derivative $D_{t_i} Y_r$ is bounded due to the given ABM dynamics by Equation 2.27; and by the continuity given by Theorem 2.5.2 we have that $\mathbb{E}\left[\left|D_s Y_t - D_s Y_r\right|^2\right] \leq C|t - r|$. Therefore, using the Lipschitz and Hölder continuities of the partial derivative of the driver, we conclude that the Y-integral term can be upper bounded by

$$\begin{aligned}\mathbb{E}\left[\left(\int_{t_i}^{t_{i+1}} \nabla_y f_r D_{t_i} Y_r - \nabla_x f^\pi_{i+1} D_i Y^\pi_{i+1} \mathrm{d}r\right)^2\right] \\ \leq 4L^2 d \Delta t_i \left(C \Delta t_i^2 + C \Delta t_i \mathbb{E}\left[\left|Y_{t_{i+1}} - Y^\pi_{i+1}\right|^2\right] + \Delta t_i \mathbb{E}\left[\left|D_{t_i} Y_{t_{i+1}} - D_i Y^\pi_{i+1}\right|^2\right]\right). \quad (6.72)\end{aligned}$$

Finally, substituting Equation 6.68 and Equation 6.72 back into Equation 6.64, choosing $\alpha := \gamma \Delta t_i < 1$ we get

$$\begin{aligned}\mathbb{E}\left[\left|Z_{t_i} - V_i\right|^2\right] \leq (1 + \gamma \Delta t_i) \mathbb{E}\left[\left|D_{t_i} Y_{t_{i+1}} - D_i Y^\pi_{i+1}\right|^2\right] \\ + \frac{16 L^2 d}{\gamma} (1 + \gamma \Delta t_i) \Big(C \Delta t_i^2 + C \Delta t_i \mathbb{E}\left[\left|Y_{t_{i+1}} - Y^\pi_{i+1}\right|^2\right] \\ + \Delta t_i \mathbb{E}\left[\left|D_{t_i} Y_{t_{i+1}} - D_i Y^\pi_{i+1}\right|^2\right] \Big). \quad (6.73)\end{aligned}$$

Choosing $\gamma := 32L^2 d$, we obtain

$$\mathbb{E}\left[|Z_{t_i} - V_i|^2\right] \leq (1 + C\Delta t_i)\,\mathbb{E}\left[\left|D_{t_i}Y_{t_{i+1}} - D_iY^\pi_{i+1}\right|^2\right] + C\Delta t_i^2 + C\Delta t_i\mathbb{E}\left[\left|Y_{t_{i+1}} - Y^\pi_{i+1}\right|^2\right], \quad (6.74)$$

what was needed to be shown. □

We have by now gathered all the relevant sub-results with respect to the representation errors arising in the proposed one-step scheme, and can state the theorem establishing the consistency of the OSM scheme under Assumption 6.1.2. The following theorem is basically only assembling the lemmas proven above.

Theorem 6.2.1 (Consistency of the One-Step Malliavin Scheme)
Under Assumption 6.1.2, with sufficiently small time steps of an equidistant time grid satisfying the conditions of Lemma 6.2.1, Lemma 6.2.2 and Lemma 6.2.3, for any choice of $\vartheta_y \in [0,1]$ we have that

$$\max_{0\leq i\leq N} \mathbb{E}\left[|Y_{t_i} - Y^\pi_i|^2\right] \leq C|\pi| + C\sum_{n=0}^{N-1}\epsilon^y_n, \quad (6.75)$$

$$\max_{0\leq i\leq N} \mathbb{E}\left[|Z_{t_i} - Z^\pi_i|^2\right] \leq C|\pi| + C\sum_{n=0}^{N-1}\left(\epsilon^y_n + \epsilon^z_n\right), \quad (6.76)$$

with some constant C independent of π^N. Consequently

$$\mathcal{E}\left[(Y^\pi, Z^\pi), (Y, Z)\right] \leq C|\pi| + C\sum_{n=0}^{N-1}\left(\epsilon^y_n + \epsilon^z_n\right). \quad (6.77)$$

Proof. In what follows C always denotes a constant independent of the time partition, whose value may vary from line to line.

We proceed in four steps. First, taking advantage of the scheme in Equation 5.15 being separated; and the driver's independence from Z under Assumption 6.1.2, we give an error estimate for the worst time step's approximation error in Y_{t_i}. Second, we give an upper bound for the approximation error of the variational process $\nabla_x Y_{t_i}$. Then, we use this upper bound for the approximation error of $D_{t_i}Y_{t_{i+1}}$. Finally, we plug these error estimates into the approximation error of Z_{t_i}.

Step 1: Approximation error of Y_{t_i}.

Under the assumptions, the results of Lemma 6.2.1 still hold, in fact – in light of Remark 6.2.1 – the upper bound for the representation error of U_i in Equation 6.12 becomes

$$\mathbb{E}\left[|Y_{t_i} - U_i|^2\right] \leq \left(1 + C\left(\vartheta_y^2 + (1-\vartheta_y)^2\right)\Delta t_i\right)\mathbb{E}\left[\left|Y_{t_{i+1}} - Y^\pi_{i+1}\right|^2\right] + C\Delta t_i^2. \quad (6.78)$$

Using $\vartheta_y^2, (1-\vartheta_y)^2 < 1$, this can be loosened to

$$\mathbb{E}\left[|Y_{t_i} - U_i|^2\right] \leq (1 + C\Delta t_i)\,\mathbb{E}\left[\left|Y_{t_{i+1}} - Y^\pi_{i+1}\right|^2\right] + C\Delta t_i^2. \quad (6.79)$$

Splitting up the representation error into approximation and regression errors, we have

$$\mathbb{E}\left[|Y_{t_i} - U_i|^2\right] \leq 2\mathbb{E}\left[|Y_{t_i} - Y^\pi_i|^2\right] + 2\mathbb{E}\left[|Y^\pi_i - U_i|^2\right] = 2\mathbb{E}\left[|Y_{t_i} - Y^\pi_i|^2\right] + \epsilon^y_i, \quad (6.80)$$

with the definition of the regression error given in Equation 6.19. Plugging this back into Equation 6.79 we get

$$\mathbb{E}\left[|Y_{t_i} - Y^\pi_i|^2\right] \leq (1 + C\Delta t_i)\,\mathbb{E}\left[\left|Y_{t_{i+1}} - Y^\pi_{i+1}\right|^2\right] + C\Delta t_i^2 + C\epsilon^\pi_i. \quad (6.81)$$

Now, applying the Discrete Grönwall lemma Proposition 6.1.2, with the analytical terminal condition $Y_{t_N} = g(X_{t_N}) = Y^\pi_N$, this becomes

$$\max_{0\leq i\leq N} \mathbb{E}\left[|Y_{t_i} - Y^\pi_i|^2\right] \leq C|\pi| + C\sum_{n=0}^{N-1}\epsilon^y_n, \quad (6.82)$$

as for an equidistant time grid $\forall i : \Delta t_i \equiv T/N$. This proves Equation 6.75.

Step 2: Approximation error of $\nabla_x Y_{t_i}$.

The conditions of Lemma 6.2.2 are satisfied, therefore by Equation 6.45 we have

$$\mathbb{E}\left[|\nabla_x Y_{t_i} - \nabla_x U_i|^2\right] \leq \left(1 + C\left(\vartheta_y^2 + (1-\vartheta_y)^2\right)\Delta t_i\right)\mathbb{E}_i\left[\left|\nabla_x Y_{t_{i+1}} - \nabla_x Y_{i+1}^\pi\right|^2\right] + C\vartheta_y^2\Delta t_i\mathbb{E}\left[|Y_{t_i} - U_i|^2\right] + C(1-\vartheta_y)^2\Delta t_i\mathbb{E}\left[\left|Y_{t_{i+1}} - Y_{i+1}^\pi\right|^2\right] + C\Delta t_i^2. \quad (6.83)$$

Since $\vartheta_y^2, (1-\vartheta_y)^2 < 1$, this can be further loosened to

$$\mathbb{E}\left[|\nabla_x Y_{t_i} - \nabla_x U_i|^2\right] \leq (1 + C\Delta t_i)\,\mathbb{E}_i\left[\left|\nabla_x Y_{t_{i+1}} - \nabla_x Y_{i+1}^\pi\right|^2\right] + C\Delta t_i\mathbb{E}\left[|Y_{t_i} - U_i|^2\right] + C\Delta t_i\mathbb{E}\left[\left|Y_{t_{i+1}} - Y_{i+1}^\pi\right|^2\right] + C\Delta t_i^2. \quad (6.84)$$

Splitting up $Y_{t_i} - U_i$ and $\nabla_x Y_{t_i} - \nabla_x U_i$ to approximation and regression errors, this leads to

$$\mathbb{E}\left[|\nabla_x Y_{t_i} - \nabla_x Y_i^\pi|^2\right] \leq (1 + C\Delta t_i)\,\mathbb{E}\left[\left|\nabla_x Y_{t_{i+1}} - \nabla_x Y_{i+1}^\pi\right|^2\right] + C\Delta t_i\mathbb{E}\left[|Y_{t_i} - Y_i^\pi|^2\right] + C\Delta t_i\mathbb{E}\left[\left|Y_{t_{i+1}} - Y_{i+1}^\pi\right|^2\right] + C\Delta t_i^2 + C\epsilon_i^y. \quad (6.85)$$

Using the bound Equation 6.75 obtained in the previous step, we have that

$$\max\left[\mathbb{E}\left[|Y_{t_i} - Y_i^\pi|^2\right], \mathbb{E}\left[\left|Y_{t_{i+1}} - Y_{i+1}^\pi\right|^2\right]\right] \leq C\Delta t_i + C\sum_{n=0}^{N-1}\epsilon_n^y. \quad (6.86)$$

Plugging this back into the inequality above yields

$$\mathbb{E}\left[|\nabla_x Y_{t_i} - \nabla_x Y_i^\pi|^2\right] \leq (1 + C\Delta t_i)\,\mathbb{E}\left[\left|\nabla_x Y_{t_{i+1}} - \nabla_x Y_{i+1}^\pi\right|^2\right] + C\Delta t_i\sum_{n=0}^{N-1}\epsilon_n^y + C\epsilon_i^y + C\Delta t_i^2. \quad (6.87)$$

Under Assumption 6.1.2, we gather the derivative of the terminal condition analytically $\nabla_x Y_{t_N} = \nabla_x g(X_{t_N}) = \nabla_x Y_N^\pi$, and therefore obtain by the Discrete Grönwall lemma Proposition 6.1.2

$$\max_{0\leq i\leq N-1}\mathbb{E}\left[|\nabla_x Y_{t_i} - \nabla_x Y_i^\pi|^2\right] \leq C\Delta t_i + C\sum_{n=0}^{N-1}\epsilon_n^y. \quad (6.88)$$

Step 3: Approximation error of $D_{t_i}Y_{t_{i+1}}$.

As a consequence, by Equation 6.59 in Corollary 6.2.1 we also have

$$\mathbb{E}\left[\left|D_{t_i}Y_{t_{i+1}} - D_i Y_{i+1}^\pi\right|^2\right] \leq C\left(1 + \left(\vartheta_y^2 + (1-\vartheta_y)^2\right)\Delta t_i\right)\mathbb{E}\left[\left|\nabla_x Y_{t_{i+1}} - \nabla_x Y_{i+1}^\pi\right|^2\right] + C\vartheta_y^2\Delta t_i\mathbb{E}\left[|Y_{t_i} - U_i|^2\right] + C(1-\vartheta_y)^2\Delta t_i\mathbb{E}\left[\left|Y_{t_{i+1}} - Y_{i+1}^\pi\right|^2\right] + C\Delta t_i^2 + C\epsilon_i^y. \quad (6.89)$$

Using $\vartheta_y^2, (1-\vartheta_y)^2 < 1$ and plugging the approximation error bound Equation 6.75 and Equation 6.88 in above gives

$$\mathbb{E}\left[\left|D_{t_i}Y_{t_{i+1}} - D_i Y_{i+1}^\pi\right|^2\right] \leq C(1 + \Delta t_i)\left(\Delta t_i + \sum_{n=0}^{N-1}\epsilon_n^y\right) + C\sum_{n=0}^{N-1}\epsilon_n^y + C\Delta t_i^2. \quad (6.90)$$

Since the right-hand side above is independent from the index i, this holds for any $0 \leq i \leq N-1$ and we obtain

$$\max_{0\leq i\leq N-1}\mathbb{E}\left[\left|D_{t_i}Y_{t_{i+1}} - D_i Y_{i+1}^\pi\right|^2\right] \leq C\Delta t_i + C\sum_{n=0}^{N-1}\epsilon_n^y. \quad (6.91)$$

Step 4: Approximation error of Z_{t_i}.

The conditions of Lemma 6.2.3 are satisfied and therefore by Equation 6.62 we have that

$$\mathbb{E}\left[|Z_{t_i} - V_i|^2\right] \leq (1 + C\Delta t_i)\,\mathbb{E}\left[\left|D_{t_i}Y_{t_{i+1}} - D_i Y_{i+1}^\pi\right|^2\right] + C\Delta t_i^2 + C\Delta t_i\mathbb{E}\left[\left|Y_{t_{i+1}} - Y_{i+1}^\pi\right|^2\right]. \quad (6.92)$$

We can loosen the right-hand side by substituting the maximum errors established by Equation 6.75 and Equation 6.91, which yields

$$\mathbb{E}\left[|Z_{t_i} - V_i|^2\right] \leq C(1+\Delta t_i)\left(\Delta t_i + \sum_{n=0}^{N-1} \epsilon_n^y\right) + C\sum_{n=0}^{N-1}\epsilon_n^y + C\Delta t_i^2. \tag{6.93}$$

Splitting up the representation error of Z_{t_i} to approximation and regression errors, furthermore using that $\epsilon_n^z \geq 0$ for all n, we get

$$\mathbb{E}\left[|Z_{t_i} - Z_i^\pi|^2\right] \leq C(1+\Delta t_i)\left(\Delta t_i + \sum_{n=0}^{N-1} \epsilon_n^y\right) + C\sum_{n=0}^{N-1}(\epsilon_n^y + \epsilon_n^z) + C\Delta t_i^2. \tag{6.94}$$

In case of equidistant time partitions, the right-hand side now does not depend on index i anymore, and therefore we conclude

$$\max_{0\leq i\leq N-1} \mathbb{E}\left[|Z_{t_i} - Z_i^\pi|^2\right] \leq C\Delta t_i + C\sum_{n=0}^{N-1}(\epsilon_n^y + \epsilon_n^z), \tag{6.95}$$

which is the form Equation 6.76 that we needed to show.

Combining Equation 6.75 and Equation 6.76 gives the upper bound Equation 6.77. This concludes the proof.

□

6.3 Consistency of the Multi-Step Malliavin Scheme

After completing the consistency error analysis of the one-step scheme, it is now time to turn to the Multi-Step Malliavin scheme proposed in Equation 5.22. Similarly as in the previous section, we proceed by proving a sequence of lemmas, which altogether shall prove the final consistency theorem stated in Theorem 6.3.1. First of all, however, let us introduce some further notations. Most importantly, the true targets of the regression problems in the multi-step scheme are defined as

$$\begin{aligned} Q_i &:= \mathbb{E}_i\left[\nabla_x g(X_N^\pi) D_i X_N^\pi + f_{i+1}^{D,\pi} D_i X_{i+1}^\pi \Delta t_i\right] \\ &\quad + \mathbb{E}_i\left[\sum_{j=i+1}^{N-1}\left(\vartheta_z f_j^{D,\pi} D_i X_j^\pi + (1-\vartheta_z) f_{j+1}^{D,\pi} D_i X_{j+1}^\pi\right)\Delta t_j\right] \end{aligned} \tag{6.96}$$

$$\begin{aligned} P_i &:= \vartheta_y f(t_i, X_i^\pi, P_i, Z_i^\pi)\Delta t_i + \mathbb{E}_i\left[g(X_N^\pi)\right] + \mathbb{E}_i\left[(1-\vartheta_y) f(t_{i+1}, X_{i+1}^\pi, Y_{i+1}^\pi, Z_{i+1}^\pi)\Delta t_i\right] \\ &\quad + \mathbb{E}_i\left[\sum_{j=i+1}^{N-1}\left(\vartheta_y f_j^\pi + (1-\vartheta_y) f_{j+1}^\pi\right)\Delta t_j\right], \end{aligned} \tag{6.97}$$

where we used the usual abbreviation defined in Equation 5.13.

In addition, let us also define the following auxiliary processes

$$\overline{U}_i := \vartheta_y f(t_i, X_i^\pi, \overline{U}_i, Z_i^\pi)\Delta t_i + \mathbb{E}_i\left[\overline{U}_{i+1}\right] + \mathbb{E}_i\left[(1-\vartheta_y) f(t_{i+1}, X_{i+1}^\pi, \overline{U}_{i+1}, Z_{i+1}^\pi)\Delta t_i\right], \tag{6.98a}$$

$$\begin{aligned} \widehat{P}_i &:= \vartheta_y f(t_i, X_i^\pi, Y_i^\pi, Z_i^\pi)\Delta t_i + \mathbb{E}_i\left[g(X_N^\pi)\right] + \mathbb{E}_i\left[(1-\vartheta_y) f(t_{i+1}, X_{i+1}^\pi, Y_{i+1}^\pi, Z_{i+1}^\pi)\Delta t_i\right] \\ &\quad + \mathbb{E}_i\left[\sum_{j=i+1}^{N-1}\left(\vartheta_y f_j^\pi + (1-\vartheta_y) f_{j+1}^\pi\right)\Delta t_j\right]. \end{aligned} \tag{6.98b}$$

It is worth to highlight that by the tower property, $\widehat{P}$ satisfies the following iterated formula

$$\widehat{P}_i = \vartheta_y f(t_i, X_i^\pi, Y_i^\pi, Z_i^\pi)\Delta t_i + \mathbb{E}_i\left[\widehat{P}_{i+1} + (1-\vartheta_y) f(t_{i+1}, X_{i+1}^\pi, Y_{i+1}^\pi, Z_{i+1}^\pi)\Delta t_i\right]. \tag{6.99}$$

Due to Markovianity we know that there exist deterministic functions $p_i : \mathbb{R}^d \to \mathbb{R}$ and $q_i : \mathbb{R}^d \to \mathbb{R}^d$ such that

$$P_i = p_i(X_i^\pi), \quad Q_i = q_i(X_i^\pi). \tag{6.100}$$

We parametrize the functions p_i and q_i by fully-connected, feedforward deep neural nets of the form $\mathcal{Y}(\cdot|\theta_i^y) := \mathcal{NN}(\cdot|\theta_i^y) : \mathbb{R}^d \to \mathbb{R}$ and $\mathcal{Z}(\cdot|\theta_i^z) := \mathcal{NN}(\cdot|\theta_i^z) : \mathbb{R}^d \to \mathbb{R}^d$. We take estimations

$$\widehat{\theta}_i^z \in \arg\min_{\eta} \mathbb{E}\left[|q_i(X_i^\pi) - \mathcal{Z}(X_i^\pi|\eta)|^2\right], \tag{6.101}$$

$$\widehat{\theta}_i^y \in \arg\min_{\eta} \mathbb{E}\left[|p_i(X_i^\pi) - \mathcal{Y}(X_i^\pi|\eta)|^2\right], \tag{6.102}$$

leading to approximations

$$Z_i^\pi := \mathcal{Z}(X_i^\pi|\widehat{\theta}_i^z), \quad Y_i^\pi := \mathcal{Y}(X_i^\pi|\widehat{\theta}_i^y). \tag{6.103}$$

Additionally, differentiating the resulting networks via automatic differentiation we take

$$\nabla_x Z_i^\pi := \nabla_x \mathcal{Z}(X_i^\pi|\widehat{\theta}_i^z), \quad \nabla_x Y_i^\pi := \nabla_x \mathcal{Y}(X_i^\pi|\widehat{\theta}_i^y), \tag{6.104}$$

Using all these notations, we define the regression errors induced by P_i, Q_i by

$$\delta_i^z := \mathbb{E}\left[|q_i(X_i^\pi) - Z_i^\pi(X_i^\pi)|^2 + |\nabla_x q_i(X_i^\pi) - \nabla_x Z_i^\pi(X_i^\pi)|^2\right] \tag{6.105}$$

$$\delta_i^y := \mathbb{E}\left[|p_i(X_i^\pi) - Y_i^\pi(X_i^\pi)|^2 + |\nabla_x p_i(X_i^\pi) - \nabla_x Y_i^\pi(X_i^\pi)|^2\right]. \tag{6.106}$$

In the light of the Universal Approximation Theorem Theorem 3.2.2, these quantities can be made arbitrarily small even with shallow feedforward neural network approximations. The goal is to bound the following squared approximation error of the discrete numerical scheme

$$\mathcal{E}\left[(Y^\pi, Z^\pi), (Y, Z)\right] := \max_{t_i \in \pi^N} \mathbb{E}\left[|Y_{t_i} - Y_i^\pi|^2\right] + \max_{t_j \in \pi^N} \mathbb{E}\left[\left|Z_{t_j} - Z_j^\pi\right|^2\right]. \tag{6.107}$$

With the use of these notations, let us now prove similar representation lemmas as before. The following proof is very similar to that of the Backward Deep Multistep method in [50] with the differences that hereby we apply a theta-discretization, and that the Z-process is approximated in a separate regression task. The main idea of the proof is to split up the representation error to a sum of one-step recursive errors induced by the auxiliary processes in Equation 6.98. In order to ease the presentation, we only state the following results under the driver's independence of Z. It is worth to note, however, that the bounds for the representation error of the Y-process established in Lemma 6.3.1 – similarly to the one-step case – would also hold in the general setup of Assumption 6.1.1.

Lemma 6.3.1 (Representation Error of Y_{t_i} – MSM)
Under Assumption 6.1.2, for sufficiently small time steps and any choice of $\vartheta_z, \vartheta_y \in [0,1]$, we have

$$\max_{0 \le i \le N} \mathbb{E}\left[|Y_{t_i} - P_i|^2\right] \le C\Delta t_i + C\Delta t_i \sum_{n=0}^{N-1} \delta_i^y, \tag{6.108}$$

where C is a constant independent of the time grid.

Proof. In what follows C always denotes a constant independent of the time partition, whose value may vary from line to line.

First, notice that the representation error of P_i can be split up to

$$\mathbb{E}\left[|Y_{t_i} - P_i|^2\right] \le 4\left(\mathbb{E}\left[\left|Y_{t_i} - \overline{U}_i\right|^2\right] + \mathbb{E}\left[\left|\overline{U}_i - \widehat{P}_i\right|^2\right] + \mathbb{E}\left[\left|\widehat{P}_i - P_i\right|^2\right]\right), \tag{6.109}$$

where $\overline{U}, \widehat{P}$ are the auxiliary processes introduced in Equation 6.98. Having this in mind, we proceed in three steps and upper bound each term's contribution on the right hand side separately.

<u>Step 1: Error of $\mathbb{E}\left[\left|Y_{t_i} - \overline{U}_i\right|^2\right]$.</u>

First of all, let us highlight that $\overline{U}_i$ is "almost" identical to Equation 6.12, with the only exception that the targets projected on $\mathcal{F}_{t_i}$ are not the regressed values at t_{i+1} but the true conditional expectations. Therefore, this error is equivalent to the discretization error of the one-step scheme. Hence, through identical steps as seen in Lemma 6.2.1 we can obtain

$$\mathbb{E}\left[\left|Y_{t_i} - \overline{U}_i\right|^2\right] \le (1 + C\Delta t_i)\,\mathbb{E}\left[\left|Y_{t_{i+1}} - \overline{U}_{i+1}\right|^2\right] + C\Delta t_i^2, \tag{6.110}$$

in case of Z independent drivers. In order to avoid repetition, we omit this proof.

Consequently, if we collect the – under Assumption 6.1.2 – analytical terminal conditions $Y_{t_N} = g(X_{t_N}) = \overline{U}_N$, by the Discrete Grönwall lemma, we get

$$\max_{0\leq i\leq N} \mathbb{E}\left[\left|Y_{t_i} - \overline{U}_i\right|^2\right] \leq C\Delta t_i. \tag{6.111}$$

This provides the first term's error contribution in Equation 6.109.

Step 2: Error of $\mathbb{E}\left[\left|\widehat{P}_i - P_i\right|^2\right]$.

For technical reasons, let us first upper bound the third term's contribution in Equation 6.109. By the definition of the auxiliary process in Equation 6.98b and the true target in Equation 6.97 we have that

$$\widehat{P}_i - P_i = \vartheta_y \Delta t_i \left[f(t_i, X_i^\pi, Y_i^\pi) - f(t_i, X_i^\pi, P_i)\right]. \tag{6.112}$$

By the Lipschitz continuity of the driver it therefore follows that

$$\mathbb{E}\left[\left|\widehat{P}_i - P_i\right|^2\right] \leq \vartheta_y^2 \Delta t_i^2 \mathbb{E}\left[\left|Y_i^\pi - P_i\right|^2\right] = \vartheta_y^2\Delta t_i^2 \delta_i^y, \tag{6.113}$$

where we used the definition of the regression error given in Equation 6.106. This establishes the error contribution of the third term.

Step 3: Error of $\mathbb{E}\left[\left|\overline{U}_i - \widehat{P}_i\right|^2\right]$.

We proceed by giving an error bound for the second term in Equation 6.109. First, notice that by the definition of the auxiliary processes in Equation 6.98 and the recursive conditional expectation formula for $\widehat{P}_i$ in Equation 6.99 we have that

$$\begin{aligned}\overline{U}_i - \widehat{P}_i = \mathbb{E}_i\left[\overline{U}_{i+1} - \widehat{P}_{i+1}\right] + \Delta t_i \vartheta_y \left[f(t_i, X_i^\pi, \overline{U}_i) - f(t_i, X_i^\pi, Y_i^\pi)\right] \\ + (1-\vartheta_y)\Delta t_i \mathbb{E}_i\left[f(t_{i+1}, X_{i+1}^\pi, \overline{U}_{i+1}) - f(t_{i+1}, X_{i+1}^\pi, Y_{i+1}^\pi)\right].\end{aligned} \tag{6.114}$$

Similar applications of the Jensen and Young inequalities as seen in Lemma 6.2.1 on top of the Lipschitz continuity of the driver, therefore lead to

$$\begin{aligned}\mathbb{E}\left[\left|\overline{U}_i - \widehat{P}_i\right|^2\right] &\leq (1+\beta)\mathbb{E}\left[\left|\overline{U}_{i+1} - \widehat{P}_{i+1}\right|^2\right] \\ &+ 4L^2\Delta t_i^2 (1+1/\beta)\left(\vartheta_y^2 \mathbb{E}\left[\left|\overline{U}_i - Y_i^\pi\right|^2\right] + (1-\vartheta_y)^2\mathbb{E}\left[\left|\overline{U}_{i+1} - Y_{i+1}\right|^2\right]\right).\end{aligned} \tag{6.115}$$

We can use the upper bound

$$\left|\overline{U}_i - Y_i^\pi\right|^2 \leq 4\left(\left|\overline{U}_i - \widehat{P}_i\right|^2 + \left|\widehat{P}_i - P_i\right|^2 + \left|P_i - Y_i^\pi\right|^2\right), \tag{6.116}$$

which in expectations gives

$$\mathbb{E}\left[\left|\overline{U}_i - Y_i^\pi\right|^2\right] \leq 4\left(\mathbb{E}\left[\left|\overline{U}_i - \widehat{P}_i\right|^2\right] + (1 + \vartheta_y^2\Delta t_i^2)\delta_i^y\right), \tag{6.117}$$

using the definition of the regression error Equation 6.106 and Equation 6.113. A similar result holds for the $i+1$'th term.

Substituting these back into Equation 6.115, choosing $\beta := \gamma\Delta t_i < 1$, $\gamma := 64L^2$, similarly as seen in Lemma 6.2.1, we subsequently gather

$$\mathbb{E}\left[\left|\overline{U}_i - \widehat{P}_i\right|^2\right] \leq (1 + C\Delta t_i)\mathbb{E}\left[\left|\overline{U}_{i+1} - \widehat{P}_i\right|^2\right] + C\Delta t_i(\delta_i^y + \delta_{i+1}^y). \tag{6.118}$$

At this point, we can apply the Discrete Grönwall lemma Proposition 6.1.2 and with the analytical terminal condition $Y_{t_N} = g(X_{t_N}) = \overline{U}_N = Y_N^\pi$ provided by Assumption 6.1.2, conclude that

$$\max_{0\leq i\leq N} \mathbb{E}\left[\left|\overline{U}_i - \widehat{P}_i\right|^2\right] \leq C\Delta t_i \sum_{n=0}^{N-1} \delta_i^y. \tag{6.119}$$

This concludes the third step of the proof.

Finally, plugging Equation 6.111, Equation 6.113 and Equation 6.119 back in Equation 6.109, we collect

$$\max_{0\leq i\leq N} \mathbb{E}\left[\left|Y_{t_i}-P_i\right|^2\right] \leq C\Delta t_i + C\Delta t_i \sum_{n=0}^{N-1}\delta_i^y, \tag{6.120}$$

which is what we needed to show. □

The above lemma establishes the representation error of the multi-step scheme's estimations for Y, induced by discrete conditional expectations. In what follows, we gather a similar result for the representation error of the Z-process under Assumption 6.1.2.

Lemma 6.3.2 (Representation Error of ∇Y_{t_i} – MSM)
Under Assumption 6.1.2, for sufficiently small equidistant time steps and any choice of $\vartheta_z, \vartheta_y \in [0,1]$, we have that

$$\max_{0\leq i\leq N} \mathbb{E}\left[\left|\nabla_x Y_{t_i}-\nabla_x P_i\right|^2\right] \leq C\Delta t_i + C\Delta t_i \sum_{n=0}^{N-1}\delta_i^y, \tag{6.121}$$

where C is a constant independent of the time grid.

Proof. In what follows C always denotes a constant independent of the time partition whose value may vary from line to line.

Similarly as in Lemma 6.3.1 we split up the representation error to the following three terms

$$\mathbb{E}\left[\left|\nabla_x Y_{t_i}-\nabla_x P_i\right|^2\right] \leq 4\Bigg(\mathbb{E}\left[\left|\nabla_x Y_{t_i}-\nabla_x \overline{U}_i\right|^2\right] + \mathbb{E}\left[\left|\nabla_x \overline{U}_i-\nabla_x \widehat{P}_i\right|^2\right] \\ + \mathbb{E}\left[\left|\nabla_x \widehat{P}_i-\nabla_x P_i\right|^2\right]\Bigg). \tag{6.122}$$

We proceed in three steps and bound each term's contribution on the right-hand side.

<u>Step 1: Error of $\mathbb{E}\left[\left|\nabla_x Y_{t_i}-\nabla_x \overline{U}_i\right|^2\right]$.</u>

Through identical steps as seen in Lemma 6.2.2 we collect – analogously to Equation 6.45

$$\mathbb{E}\left[\left|\nabla_x Y_{t_i}-\nabla_x \overline{U}_i\right|^2\right] \leq \left(1+C\left(\vartheta_y^2+(1-\vartheta_y)^2\right)\Delta t_i\right)\mathbb{E}\left[\left|\nabla_x Y_{t_{i+1}}-\nabla_x \overline{U}_{i+1}\right|^2\right] \\ + C\vartheta_y^2\Delta t_i \mathbb{E}\left[\left|Y_{t_i}-\overline{U}_i\right|^2\right] + C(1-\vartheta_y)^2\Delta t_i\mathbb{E}\left[\left|Y_{t_{i+1}}-\overline{U}_{i+1}\right|^2\right] + C\Delta t_i^2. \tag{6.123}$$

By the result obtained in Equation 6.111, this for any choice of $\vartheta_y\in[0,1]$ yields

$$\mathbb{E}\left[\left|\nabla_x Y_{t_i}-\nabla_x \overline{U}_i\right|^2\right] \leq (1+C\Delta t_i)\,\mathbb{E}\left[\left|\nabla_x Y_{t_{i+1}}-\nabla_x \overline{U}_{i+1}\right|^2\right] + C\Delta t_i^2. \tag{6.124}$$

Using the – under Assumption 6.1.2 – analytical terminal condition $\nabla_x Y_{t_N} = \nabla_x g(X_{t_N}) = \nabla_x \overline{U}_N$ we thus by the Discrete Grönwall lemma Proposition 6.1.2 conclude

$$\max_{0\leq i\leq N} \mathbb{E}\left[\left|\nabla_x Y_{t_i}-\overline{U}_i\right|^2\right] \leq C\Delta t_i. \tag{6.125}$$

<u>Step 2: Error of $\mathbb{E}\left[\left|\nabla_x \widehat{P}_i-\nabla_x P_i\right|^2\right]$.</u>

For technical reasons, let us proceed with the third term's contribution in Equation 6.122. Differentiating Equation 6.112 we easily get

$$\nabla_x \widehat{P}_i - \nabla_x P_i = \vartheta_y\Delta t_i\left[\nabla_x f(t_i, X_{t_i}, Y_i^\pi) - \nabla_x f(t_i, X_{t_i}, P_i)\right] \\ + \vartheta_y \Delta t_i\left[\nabla_y f(t_i, X_{t_i}, Y_i^\pi)\nabla_x Y_i^\pi - \nabla_y f(t_i, X_{t_i}, P_i)\nabla_x P_i\right]. \tag{6.126}$$

By the usual arguments of Lipschitz continuity this yields

$$\mathbb{E}\left[\left|\nabla_x \widehat{P}_i-\nabla_x P_i\right|^2\right] \leq 2C\vartheta_y^2\Delta t_i^2\left(\mathbb{E}\left[\left|Y_i^\pi-P_i\right|^2\right] + \mathbb{E}\left[\left|\nabla_x Y_i^\pi-\nabla_x P_i\right|^2\right]\right), \tag{6.127}$$

which translates to

$$\mathbb{E}\left[\left|\nabla_x \widehat{P}_i - \nabla_x P_i\right|^2\right] \le C\vartheta_y^2 \Delta t_i^2 \delta_i^y. \tag{6.128}$$

Step 3: Error of $\mathbb{E}\left[\left|\nabla_x \overline{U}_i - \nabla_x \widehat{P}_i\right|^2\right]$.

Finally, let us upper bound the second term's contribution in Equation 6.122. Differentiating Equation 6.114, using Leibniz's integral rule gives

$$\begin{aligned}\nabla_x \overline{U}_i - \nabla_x \widehat{P}_i = \mathbb{E}_i\left[\nabla_x \overline{U}_{i+1} - \nabla_x \widehat{P}_{i+1}\right] + \Delta t_i \vartheta_y \left[f(t_i, X_i^\pi, \overline{U}_i) - f(t_i, X_i^\pi, Y_i^\pi)\right] \\ + (1-\vartheta_y)\Delta t_i \mathbb{E}_i\left[f(t_{i+1}, X_{i+1}^\pi, \overline{U}_{i+1}) - f(t_{i+1}, X_{i+1}^\pi, Y_{i+1}^\pi)\right].\end{aligned} \tag{6.129}$$

Through the usual arguments of Lipschitz continuity we therefore gather

$$\begin{aligned}\mathbb{E}\left[\left|\nabla_x \overline{U}_i - \nabla_x \widehat{P}_i\right|^2\right] \le (1+\gamma\Delta t_i)\,\mathbb{E}\left[\left|\nabla_x \overline{U}_{i+1} - \nabla_x \widehat{P}_{i+1}\right|^2\right] \\ \frac{4L^2}{\gamma}(1+\gamma\Delta t_i)\Delta t_i \left(C\mathbb{E}\left[\left|\overline{U}_i - Y_i^\pi\right|^2\right] + \vartheta_y^2 \mathbb{E}\left[\left|\nabla_x \overline{U}_i - \nabla_x Y_i^\pi\right|^2\right]\right. \\ \left. +(1-\vartheta_y)^2 \mathbb{E}\left[\left|\nabla_x \overline{U}_{i+1} - \nabla_x Y_{i+1}^\pi\right|^2\right]\right).\end{aligned} \tag{6.130}$$

Now, we can use the upper bound

$$\left|\nabla_x \overline{U}_i - \nabla_x Y_i^\pi\right|^2 \le 4\left(\left|\nabla_x \overline{U}_i - \nabla_x \widehat{P}_i\right|^2 + \left|\nabla_x \widehat{P}_i - \nabla_x P_i\right|^2 + \left|\nabla_x P_i - \nabla_x Y_i^\pi\right|^2\right), \tag{6.131}$$

which in expectations gives

$$\mathbb{E}\left[\left|\nabla_x \overline{U}_i - \nabla_x Y_i^\pi\right|^2\right] \le 4\left(\mathbb{E}\left[\left|\nabla_x \overline{U}_i - \nabla_x \widehat{P}_i\right|^2\right] + (1 + C\vartheta_y^2 \Delta t_i^2)\delta_i^y\right), \tag{6.132}$$

with the use of Equation 6.128 and the definition of the regression error given in Equation 6.106. Substituting this back into Equation 6.130, with a choice of $\gamma = 8L^2$ thus gives for every $\vartheta_y \in [0,1]$

$$\mathbb{E}\left[\left|\nabla_x \overline{U}_i - \nabla_x \widehat{P}_i\right|^2\right] \le (1 + C\Delta t_i)\,\mathbb{E}\left[\left|\nabla_x \overline{U}_{i+1} - \nabla_x \widehat{P}_{i+1}\right|^2\right] + C\Delta t_i(\delta_i^y + \delta_{i+1}^y). \tag{6.133}$$

Taking advantage of the analytical terminal condition under Assumption 6.1.2, we have that $Y_{t_N} = \overline{U}_N = \widehat{P}_N$, and therefore applying the Discrete Grönwall lemma Proposition 6.1.2 yields

$$\max_{0\le i\le N} \mathbb{E}\left[\left|\nabla_x \overline{U}_i - \nabla_x \widehat{P}_i\right|^2\right] \le C\Delta t_i + C\Delta t_i \sum_{n=0}^{N-1} \delta_n^y. \tag{6.134}$$

This concludes the third step of the proof.

Let us now turn back to Equation 6.122. Plugging Equation 6.125, Equation 6.128 and Equation 6.134 into the right-hand side leads to

$$\max_{0\le i\le N} \mathbb{E}\left[\left|\nabla_x Y_{t_i} - \nabla_x P_i\right|^2\right] \le C\Delta t_i + C\Delta t_i \sum_{n=0}^{N-1} \delta_i^y, \tag{6.135}$$

concluding the proof. □

As in Corollary 6.2.1, the above lemma gives a natural upper bound for the estimations of the Malliavin derivative. This is stated in the following corollary.

Corollary 6.3.1 (Approximation Error of $D_{t_i} Y_s$ – MSM)
Under the assumptions of Lemma 6.3.2, we also have that for any $0 \le i < j \le N$

$$\mathbb{E}\left[\left|D_{t_i} Y_{t_j} - D_i Y_j^\pi\right|^2\right] \le C\Delta t_i + C\delta_j^y + C\Delta t_i \sum_{n=0}^{N-1} \delta_i^y, \tag{6.136}$$

where C is a constant independent of the time grid.

Proof. In what follows C always denotes a constant independent of the time partition whose value may vary from line to line.

Using the Malliavin chain rule and the definition of the approximation defined in Equation 6.1, we have that the approximation error of $D_{t_i}Y_{t_j}$ for any $i<j$ admits to

$$D_{t_i}Y_{t_j} - D_iY_j^\pi = \nabla_x Y_{t_j} D_{t_i}X_{t_j} - \nabla_x Y_j^\pi D_i X_j^\pi. \tag{6.137}$$

Under ABM dynamics this readily reads as

$$\mathbb{E}\left[\left|D_{t_i}Y_{t_j} - D_iY_j^\pi\right|^2\right] \leq C\mathbb{E}\left[\left|\nabla_x Y_{t_j} - \nabla_x Y_j^\pi\right|^2\right], \tag{6.138}$$

by the boundedness of the Malliavin derivative of Arithmetic Brownian Motions in Equation 2.9. The result then follows from Lemma 6.3.2 and the definition of the regression error in Equation 6.106. □

Using Lemma 6.3.2 and Corollary 6.3.1, we can prove an upper bound for the representation error of the Z-process induced by the discrete scheme in Equation 6.96. This is collected in the following lemma.

Lemma 6.3.3 (Representation Error of Z_{t_i} – MSM)
Under Assumption 6.1.2, for sufficiently small equidistant time steps and any choice of $\vartheta_z, \vartheta_y \in [0,1]$, we have that

$$\max_{0\leq i\leq N} \mathbb{E}\left[|Z_{t_i} - Q_i|^2\right] \leq C\Delta t_i + C\Delta t_i \sum_{n=0}^{N-1}\delta_n^y, \tag{6.139}$$

where C is a constant independent of the time grid.

Proof. In what follows C always denotes a constant independent of the time grid, whose value may vary from line to line.

Most of the work has already been done by Corollary 6.3.1. In order to see this, let us expand the error term by the definition of Q_i in Equation 6.96 and the Z_{t_i} given by Theorem 2.5.1, which yields

$$\begin{aligned} Z_{t_i} - Q_i = \mathbb{E}_i\left[\nabla_x g(X_{t_N})D_{t_i}X_{t_N} - \nabla_x g(X_N^\pi)D_iX_N^\pi\right] + \mathbb{E}_i\left[\int_{t_i}^{t_{i+1}} f_r^D D_{t_i}X_r - f_i^{D,\pi}D_iX_{i+1}^\pi \mathrm{d}r\right] \\ + \vartheta_z \mathbb{E}_i\left[\sum_{j=i+1}^{N-1}\int_{t_j}^{t_{j+1}} f_r^D D_{t_i}X_r - f_j^{D,\pi}D_iX_j^\pi \mathrm{d}r\right] \\ + (1-\vartheta_z)\mathbb{E}_i\left[\sum_{j=i+1}^{N-1}\int_{t_j}^{t_{j+1}} f_r^D D_{t_i}X_r - f_{j+1}^{D,\pi}D_iX_{j+1}^\pi \mathrm{d}r\right], \end{aligned} \tag{6.140}$$

where we used the definition of $f_j^{D,\pi}$ given by Equation 5.13. Under Assumption 6.1.2, we immediately observe that due to the analytical terminal condition $\nabla_x g(X_{t_N}) \equiv \nabla_x g(X_N^\pi)$ and also $D_{t_i}X_{t_N} \equiv D_iX_N^\pi$, thus the first term cancels out. By the usual arguments of the Young and Jensen inequalities, exploiting the fact that $\vartheta_z < 1$ this leads to

$$\mathbb{E}\left[|Z_{t_i} - Q_i|^2\right] \leq C\mathbb{E}\left[\left|\sum_{j=i}^{N-1}\int_{t_j}^{t_{j+1}} f_r^D D_{t_i}X_r - f_{j+1}^{D,\pi}D_{t_i}X_{t_{j+1}}\mathrm{d}r\right|^2\right] \tag{6.141}$$

$$\leq C(N-1-i)\mathbb{E}\left[\sum_{j=i}^{N-1}\left|\int_{t_j}^{t_{j+1}} f_r^D D_{t_i}X_r - f_{j+1}^{D,\pi}D_{t_i}X_{t_{j+1}}\mathrm{d}r\right|^2\right], \tag{6.142}$$

where we used $(a+b)^2 \leq 2(a^2+b^2)$ for a summation of n-many terms. The Cauchy–Schwartz inequality thus now leads to

$$\mathbb{E}\left[|Z_{t_i} - Q_i|^2\right] \leq C(N-1-i)\sum_{j=i}^{N-1}\Delta t_j \mathbb{E}\left[\int_{t_j}^{t_{j+1}}\left|f_r^D D_{t_i}X_r - f_{j+1}^{D,\pi}D_{t_i}X_{t_{j+1}}\right|^2\mathrm{d}r\right]. \tag{6.143}$$

Let us now turn to the integrands of the summation terms. By the definition of f_r^D in Equation 5.13 we have that

$$\begin{aligned} f_r^D D_{t_i}X_r - f_{j+1}^{D,\pi}D_{t_i}X_{t_{j+1}} = \nabla_x f(r, X_r, Y_r)D_{t_i}X_r - \nabla_x f(t_{i+1}, X_{t_{i+1}}, Y_{i+1}^\pi)D_{t_i}X_{t_{i+1}} \\ + \nabla_y f(r, X_r, Y_r)D_{t_i}Y_r - \nabla_y f(t_{i+1}, X_{t_{i+1}}, Y_{i+1}^\pi)D_iY_{i+1}^\pi, \end{aligned} \tag{6.144}$$

with which we gather the upper bound

$$\begin{aligned}\mathbb{E}\left[\left|f_r^D D_{t_i} X_r - f_{j+1}^{D,\pi} D_{t_i} X_{t_{j+1}}\right|^2\right] \\ &\leq 2\mathbb{E}\left[\left|\nabla_x f(r, X_r, Y_r) D_{t_i} X_r - \nabla_x f(t_{j+1}, X_{t_{j+1}}, Y_{j+1}^\pi) D_{t_i} X_{t_{j+1}}\right|^2\right] \\ &\quad + 2\mathbb{E}\left[\left|\nabla_y f(r, X_r, Y_r) D_{t_i} Y_r - \nabla_y f(t_{j+1}, X_{t_{j+1}}, Y_{j+1}^\pi) D_i Y_{j+1}^\pi\right|^2\right]. \end{aligned} \tag{6.145}$$

Similarly as seen in Lemma 6.2.3 – see Equation 6.68 in particular – we obtain the following upper bound for the first term

$$\begin{aligned}\mathbb{E}\left[\left|\nabla_x f(r, X_r, Y_r) D_{t_i} X_r - \nabla_x f(t_{j+1}, X_{t_{j+1}}, Y_{j+1}^\pi) D_{t_i} X_{t_{j+1}}\right|^2\right] \\ \leq C\left(\Delta t_j + \mathbb{E}\left[\left|Y_{t_{j+1}} - Y_{j+1}^\pi\right|^2\right]\right), \end{aligned} \tag{6.146}$$

by the boundedness of $D_{t_i} X_r$ for ABM. Furthermore, we can rewrite the second term of Equation 6.145 in the form $\nabla_y f_r D_{t_i} Y_r - \nabla_y f_{j+1}^\pi D_i Y_{j+1}^\pi = (\nabla_y f_r - \nabla_y f_{j+1}^\pi) D_{t_i} Y_r + \nabla_y f_{i+1}^\pi (D_{t_i} Y_r - D_i Y_{j+1}^\pi)$ and – similarly to Equation 6.71 in Lemma 6.2.3 – use that for ABM dynamics under Assumption 6.1.2 the Malliavin derivative is bounded given by Equation 2.27. On top of the continuity of the Malliavin derivative $\mathbb{E}\left[\left|D_{t_i} Y_r - D_{t_i} Y_{t_{j+1}}\right|^2\right] \leq C|t_{j+1} - r|$ provided by Theorem 2.5.2, this therefore implies

$$\begin{aligned}\mathbb{E}\left[\left|\nabla_y f(r, X_r, Y_r) D_{t_i} Y_r - \nabla_y f(t_{j+1}, X_{t_{j+1}}, Y_{j+1}^\pi) D_i Y_{j+1}^\pi\right|^2\right] \\ \leq C\left(\Delta t_j + \mathbb{E}\left[\left|Y_{t_{j+1}} - Y_{j+1}^\pi\right|^2\right] + \mathbb{E}\left[\left|D_{t_i} Y_{t_{j+1}} - D_i Y_{j+1}^\pi\right|^2\right]\right). \end{aligned} \tag{6.147}$$

Splitting up the approximation error to representation and regression errors, with the use of the upper bound Equation 6.108 established in Lemma 6.3.1 we know that

$$\mathbb{E}\left[\left|Y_{t_{j+1}} - Y_{j+1}^\pi\right|^2\right] \leq C\Delta t_j + C\Delta t_j \sum_{n=0}^{N-1} \delta_n^y + C\delta_{j+1}^y. \tag{6.148}$$

Similarly, using the error bound Equation 6.136 established in Corollary 6.3.1 we collect

$$\mathbb{E}\left[\left|D_{t_i} Y_{t_{j+1}} - D_i Y_{j+1}^\pi\right|^2\right] \leq C\Delta t_j + C\delta_{j+1}^y + C\Delta t_j \sum_{n=0}^{N-1} \delta_n^y. \tag{6.149}$$

Plugging Equation 6.148 and Equation 6.149 estimates back into Equation 6.146 and Equation 6.147 respectively, and then into Equation 6.145 we get

$$\mathbb{E}\left[\left|f_r^D D_{t_i} X_r - f_{j+1}^{D,\pi} D_{t_i} X_{t_{j+1}}\right|^2\right] \leq C\Delta t_j \left(1 + \sum_{n=0}^{N-1} \delta_n^y\right) + C\delta_{j+1}^y. \tag{6.150}$$

Plugging this back into Equation 6.143 subsequently yields

$$\begin{aligned}\mathbb{E}\left[\left|Z_{t_i} - Q_i\right|^2\right] &\leq C(N-1-i) \sum_{j=i}^{N-1} \left[\Delta t_j^3 \left(1 + \sum_{n=0}^{N-1} \delta_n^y\right) + \Delta t_j \delta_{j+1}^y\right] \\ &\leq C(N-1-i)^2 \left[\Delta t_j^3 \left(1 + \sum_{n=0}^{N-1} \delta_n^y\right) + \Delta t_j \delta_{j+1}^y\right]. \end{aligned} \tag{6.151}$$

Now, exploiting the fact that $i < N$ and $\forall j : \Delta t_j := T/N$ in an equidistant time grid, we conclude

$$\mathbb{E}\left[\left|Z_{t_i} - Q_i\right|^2\right] \leq C\Delta t_i + C\Delta t_i \sum_{n=0}^{N-1} \delta_n^y, \tag{6.152}$$

concluding the proof. □

This concludes the sequence of lemmas that we needed show in order to be able to prove the final consistency theorem of the multi-step scheme, which is stated below.

Theorem 6.3.1 (Consistency of the Multi-Step Malliavin Scheme)
Under Assumption 6.1.2, with sufficiently small time steps of an equidistant time grid, for any choice of $\vartheta_y, \vartheta_z \in [0, 1]$, *we have that*

$$\max_{0\le i\le N} \mathbb{E}\left[|Y_{t_i} - Y_i^\pi|^2\right] \le C|\pi| + C\sum_{n=0}^{N-1} \delta_n^y, \tag{6.153}$$

$$\max_{0\le i\le N} \mathbb{E}\left[|Z_{t_i} - Z_i^\pi|^2\right] \le C|\pi| + C\sum_{n=0}^{N-1} \left(|\pi|\delta_n^y + \delta_n^z\right), \tag{6.154}$$

with some constant C *independent of the* π^N. *Consequently,*

$$\mathcal{E}\left[(Y^\pi, Z^\pi), (Y, Z)\right] \le C|\pi| + C\sum_{n=0}^{N-1} \left(\epsilon_n^y + \epsilon_n^z\right). \tag{6.155}$$

Proof. The results naturally follow from Equation 6.108 in Lemma 6.3.1 and Equation 6.139 in Lemma 6.3.3. Combining Equation 6.153 and Equation 6.154 gives Equation 6.155. This concludes the proof. □

6.4 Comparison of Results

Having obtained an approximation error bound for both the one- and multi-step schemes, let us now briefly evaluate the results qualitatively.

First of all, observe that the right-hand sides of both Equation 6.77 and Equation 6.155 are $\mathcal{O}(|\pi|)$ functions, in case the regression errors are $\mathcal{O}(|\pi|)$ functions as well. For the latter, one can argue – motivated by the Universal Approximation Theorem in Theorem 3.2.2 – that the regression errors can indeed be made arbitrarily small with neural network parametrizations as neural networks are dense function approximators in Sobolev spaces. Second, it is worth to highlight the differences between the two bounds. We see that for the multi-step scheme, in Equation 6.154, the regression errors δ_n^y induced by the conditional expectation regressions for the Y-process are $\mathcal{O}(1/N)$ smaller than in case of the one-step scheme in Equation 6.76. This difference confirms the implication of other results in the literature, where the interdependence between regression errors are mitigated via taking multiple time steps into account – see, e.g., [21], [50]. Furthermore, we see that neither scheme's approximation error in Y – Equation 6.75, Equation 6.153 – depends on the regression errors $(\epsilon_i^z, \delta_i^z)$ of Z. It is important to notice that this behaviour is solely due to the conditions set in Assumption 6.1.2 – namely because of the driver's independence from the control process – and that under general conditions it would not be the case.

Nevertheless, even under these more restricted conditions, it is of interest to see how the results in Theorem 6.2.1 and Theorem 6.3.1 compare to the error bounds given for the Backward Deep BSDE schemes introduced in section 4.4. As hinted in that section, the authors in [9] proved a similar consistency theorem for the Backward Deep BSDE method, which is stated as follows.

Theorem 6.4.1 (Consistency of the Backward Deep BSDE Method – Theorem 4.1, [9])
Under certain regularity assumptions, for small enough mesh sizes, there exists a constant C *such that*

$$\begin{aligned}\max_{0\le i\le N} \mathbb{E}\left[|Y_{t_i} - Y_i^\pi|^2\right] + \mathbb{E}\left[\sum_{i=0}^{N-1}\int_{t_i}^{t_{i+1}} |Z_t - Z_i^\pi|^2 \mathrm{d}t\right] \\ \le C\left(\mathbb{E}\left[|g(X_T) - g(X_N^\pi)|^2\right] + |\pi| + \varepsilon^Z(\pi^N) + \sum_{i=0}^{N-1}\left(N\epsilon_i^y + \epsilon_i^z\right)\right).\end{aligned} \tag{6.156}$$

Proof. See [9, Pg. 10, Theorem 4.1]. □

Similarly, for the so called Backward Multistep Deep BSDE method, the authors in [50] provide an error analysis for a multi-step scheme.

Theorem 6.4.2 (Consistency of the Deep Backward Multistep Method – Theorem 4.1, [50])
Under certain regularity assumptions, for small enough mesh sizes we have that

$$\begin{aligned}\max_{0\le i\le N} \mathbb{E}\left[|Y_{t_i} - Y_i^\pi|^2\right] + \mathbb{E}\left[\sum_{i=0}^{N-1}\int_{t_i}^{t_{i+1}} |Z_t - Z_i^\pi|^2 \mathrm{d}t\right] \\ \le C\left(\mathbb{E}\left[|g(X_T) - g(X_N^\pi)|^2\right] + |\pi| + \varepsilon^Z(\pi^N) + \sum_{i=0}^{N-1}\left(\epsilon_i^y + |\pi|\epsilon_i^z\right)\right).\end{aligned} \tag{6.157}$$

Proof. See [50, Pg. 17, Theorem 4.1]. □

Let us compare these results against the approximation errors proven in Theorem 6.2.1 and Theorem 6.3.1. The first big difference which we shall not leave unmentioned is that these two theorems bound a strictly weaker error measure, where the approximation error of the Z-process is not bounded by the supremum norm but rather only by an integral norm projected on each step's time interval. This advantage of the hereby proposed schemes is entirely coming from the Malliavin problem formulation, which – through Theorem 2.5.1 – enables us to give bounds for the control process in the supremum norm. This difference in formulation is the reason for the absence of $\varepsilon^Z(|\pi|)$ – defined in Equation 6.2 – in the bounds of Theorem 6.2.1 and Theorem 6.3.1 as well. However, as shown in [5], $\varepsilon^Z(|\pi|) = \mathcal{O}(|\pi|)$ in case the terminal condition is also a Lipschitz continuous function. Therefore this does not induce an additional order of magnitude in the error compared to the hereby proposed schemes, as under Assumption 2.5.1 the terminal condition is guaranteed to be Lipschitz continuous.

Hence, we can conclude that Theorem 6.4.1 and Theorem 6.4.2 with the conditions under which our consistency bounds were proven – stated in Assumption 6.1.2 – would read as

$$\max_{0\le i\le N} \mathbb{E}\left[|Y_{t_i} - Y_i^\pi|^2\right] + \mathbb{E}\left[\sum_{i=0}^{N-1}\int_{t_i}^{t_{i+1}} |Z_t - Z_i^\pi|^2 \mathrm{d}t\right] \le C\left(|\pi| + \sum_{i=0}^{N-1}(N\epsilon_i^y + \epsilon_i^z)\right), \tag{6.158}$$

$$\max_{0\le i\le N} \mathbb{E}\left[|Y_{t_i} - Y_i^\pi|^2\right] + \mathbb{E}\left[\sum_{i=0}^{N-1}\int_{t_i}^{t_{i+1}} |Z_t - Z_i^\pi|^2 \mathrm{d}t\right] \le C\left(|\pi| + \sum_{i=0}^{N-1}(\epsilon_i^y + |\pi|\epsilon_i^z)\right). \tag{6.159}$$

This means, that the bounds of both OSM and MSM are $\mathcal{O}(1/N)$ better in the cumulative regression error of Y than in the Backward Deep BSDE scheme – on top of bounding a strictly stronger norm. On the other hand, we also see that they are $\mathcal{O}(N)$ worse in the cumulative regression error of Z than in the Deep Backward Multi-Step Method. Nevertheless, we can observe that the proposed schemes are both an order of magnitudes better in the regression errors of ϵ_i^y, δ_i^y for the approximation error in Z. However, it is important to keep in mind, that everything we have said above only holds under the stronger conditions of Assumption 6.1.2.

6.5 Role of Restrictions in the Additional Assumptions

Before concluding the analysis, let us have a few words on the assumptions we made. As we have seen, all results corresponding to the error schemes of the Z-process have been stated under the more restricted set of assumptions collected in Assumption 6.1.2. In particular, we assumed three further conditions on top of the ones in Assumption 6.1.1 providing convergence for standard numerical schemes: in the driver's independence of the Z-process; in the underlying forward diffusion being ABM; and in the Hölder continuities of the partial derivatives of the driver. The last one – similarly to the difference between Assumption 2.5.1 and Assumption 6.1.1 – only provides convergence for discrete approximation schemes and can be considered standard. Below, we motivate the first two restrictions and explain why they were needed to be made for the analysis.

6.5.1 Arithmetic Brownian Motion Diffusion

Let us start with the assumptions on the forward diffusion. The need for the assumption of ABM dynamics is twofold: first through a closed-form solution for X_t and D_sX_t – via Equation 1.12 and Equation 2.9 – it eliminates the necessity of solving the underlying forward SDE and the linear SDEs of its Malliavin derivatives numerically; second, given specifically for ABM, its flow process and Malliavin derivatives are uniformly bounded. In what follows we elaborate on these two observations.

Closed-Form Solution

The need for an analytically solvable forward dynamics is not merely due to the numerical solution of the underlying diffusion. Indeed, as it is stated in Theorem 6.1.1, the Euler-scheme – defined in Equation 4.6 – is consistent with the true solution of the forward diffusion and could therefore be used to gather estimates of an analytically not solvable forward process. In that case – see the right-hand side of Equation 6.12 in Lemma 6.2.1 – we would have an additional term in our error bound depending on the approximation error of the non-analytical terminal condition $g(X_{t_N}) - g(X_N^\pi)$ in the scheme of Y.

In fact, the real difficulty with an Itô-process whose solution needs to be obtained numerically lies in the estimations of its Malliavin derivatives, which is a fundamental input to the proposed schemes.

Regardless the fact that under standard smoothness assumptions – see Assumption 2.5.1 – on the FBSDE's Malliavin differentiability, the Malliavin derivative of X does exist and exhibits all the properties we relied on during the analysis – see Theorem 2.4.1 –, we still encounter a problem with respect to the accuracy of their numerical approximations, since they hierarchically depend on the approximation errors of the forward process itself. Indeed, as we have seen in chapter 4 – see Equation 4.8 in particular – the same Euler scheme can be applied to the forward SDE of $\{D_{t_i}X\}, s \in [0,T]$. However, this forward SDE starts off from an initial condition $\sigma(t_i, X_i^\pi)$, which already depends on the accuracy of the numerical approximation of X_{t_i}. Because of this non-fixed, *noisy* initial condition, it is challenging to obtain rigorous theoretical guarantees for the approximations of each Malliavin derivative's trajectory in the time grid.

Since in the upcoming chapter's numerical experiments we only present results on problems with forward dynamics which possess a closed-form solution – such as Arithmetic Brownian Motions – we thus decided to restrict ourselves to the special case of analytically solvable forward diffusions. However, this relaxation may also be motivated by the uncoupledness of the FBSDE system Equation 1.56. In fact, because of the forward process being independent from the solutions of the backward equation, the simulation of the forward diffusion can be done *offline*, independently from the backward solution. This in fact provides a possibility to solve the forward dynamics Equation 1.56a on a magnitudes much finer time grid – say, e.g., with 10^3N grid points – and pass those finer approximations to the backward stage of the proposed algorithms. Relying on Theorem 6.1.1, this would guarantee a much higher order of convergence for the forward scheme and may also mitigate the error stemming from the numerical approximations of its Malliavin derivatives. Nevertheless, the proper error formulation of this intuition is out of the scope of this work, and is left for future research.

Boundedness

Another step where we were relying on the assumption of Arithmetic Brownian Motion dynamics was the splitting of chain rule like products. In fact, when taking the expectations of products of non-negative random variables in Equation 6.49, Equation 6.54 and Equation 6.67, we readily used that for ABM dynamics both the flow process and the Malliavin derivative are constant and therefore uniformly bounded. This property was also fundamental in the approximation errors shown for $D_{t_i}Y_{t_{i+1}}$. We have a strong belief that this assumption can be relaxed to more general not necessarily bounded forward diffusions with the help of classical inequalities such as the Hölder and Doob's maximal inequality. However, as in the upcoming section we only show numerical results for ABM dynamics we decided to eliminate this difficulty for the purpose of this work, and left the proof under general diffusions for future research.

6.5.2 Drivers Independent of Z

Finally, let us argue why the assumption of drivers with $\nabla_z f(\cdot,\cdot,\cdot,\cdot) = 0$ was needed. As we have seen, this additional assumption was not used for the representation error bound of Y proven in Lemma 6.2.1. It does, however, play a crucial role in the approximations of the Z-process. In fact, in Lemma 6.2.3 – and similarly in its multi-step counterpart – when we gave an upper bound for the square of the time integrand corresponding to $\nabla_y f$ – see, in particular, Equation 6.71 –, we used the fact that the Malliavin derivative of Y, exhibits the following two essential properties guaranteed by Theorem 2.5.2 and Lemma 2.5.1

1. boundedness: $|D_{t_i}Y_r|^2 < \infty$;
2. continuity: $\mathbb{E}\left[|D_{t_i}Y_r - D_{t_i}Y_u|^2\right] \le C|r-u|$.

The first property is provided by the assumption of ABM dynamics which we discussed above. The second, relates to the continuity of the Malliavin derivatives. Had we allowed for drivers depending on Z, we would have had to deal with a similar expression as in Equation 6.71 only containing the partial derivatives in z. This expression would have included terms depending on the random variable $\left|D_{t_i}Z_{t_{i+1}} - D_{t_i}Z_r\right|^2$. Similarly as in the steps of Lemma 6.2.3, in order to obtain a final upper bound for the integrands, we would have needed the continuity of $D_{t_i}Z$. This, however, is not guaranteed by the first-order Malliavin derivatives of the solutions, as Theorem 2.5.1 does not establish the continuity of the Malliavin derivative of Z.

If one wants to guarantee such properties of the $\{D_sZ\}_{s\in[0,T]}$, then an additional layer of stochasticity becomes needed. In fact, it can be shown – see, e.g., [57, Theorem 3.14] – that, under stronger conditions, similar results to that of Theorem 2.5.1 hold for the *second-order Malliavin derivatives* of the solution pair of the BSDE Equation 1.56b. Furthermore, it can also be shown that the second-order Malliavin derivative $D_{t_i}^2Y$ admits to a version such that $D_{t_i}^2Y = D_{t_i}Z$, almost surely. Recently, Izumi in [57] has shown that the higher-order Malliavin derivatives of the solution pair of the BSDE also satisfy linear BSDEs themselves and admit to similar relations as in Theorem 2.5.1. However, to the best of our knowledge, the continuity properties of these processes are yet to be fully understood, and therefore we decided to eliminate this difficulty through the assumption of Z-independent drivers. Nevertheless, given the interesting developments of the field in this direction – see also [58] –, extending the error bounds of Theorem 6.2.1 and Theorem 6.3.1 to the case of general Z-dependent drivers will be in the central scope of future research.

Chapter 7

Numerical Experiments

To demonstrate the numerical performance of the One-Step and Multi-Step Malliavin schemes, in the following section we present numerical results of the proposed algorithms on several high-dimensional problems and compare them to existing deep learning based BSDE solvers. We recall that the goal of both OSM and MSM is to provide more regularity for the control process Z by applying direct training on it through Malliavin calculus. Therefore, in what follows we only present results on the one-step Backward Deep BSDE method – defined by the loss in Equation 4.24 – as according to our numerical experiments it provides better control estimates than its multi-step counterpart – defined by Equation 4.26. We remark that due to its significantly worse performance at $t > 0$ we exclude the Forward Deep BSDE method from the presentation of high-dimensional problems and restrict the analysis to backward type methods which are designed to solve the FBSDE throughout the whole time horizon.

The chapter is built up as follows. First, we fix some final notations corresponding to error measures of the discrete schemes. Thereafter, we briefly explain the custom implementation with which the results were obtained and discuss the hyperparameter selection used in the experiments. We highlight that in our empirical experience OSM and MSM showed less sensitivity to the selection of these parameters than other Deep BSDE methods which is in line with the intuition that smaller learning problems should yield easier optimization. Nevertheless, in order to provide a fair comparison we decided to choose values best suited for the Backward Deep BSDE method and present results on a common set of hyperparameters. The discussion of the implementation is followed by the presentation of a non-trivial one-dimensional problem, where we demonstrate that the proposed schemes are indeed sensible. Thereafter, we turn to the main goal of this thesis and tackle high-dimensional FBSDEs up to $d = 10$ dimensions. First we consider the famous Hamilton–Jacobi–Bellman equation with linear randomness and quadratic control. We derive a *"semi-analytical"* expression which traces back the solution to the numerical integration of a set of ordinary differential equations (ODE). Using this reference solution we measure the accuracy of the proposed algorithms over the whole time window. Ultimately, in the last section we consider an FBSDE with a complex structure, whose driver is not everywhere differentiable, and for which the Backward Deep BSDE method is known to fail to provide accurate approximations in $d = 10$ dimensions – see [9, Section 5.2]. We demonstrate that OSM and MSM manage to overcome this problem, giving an order of magnitude better control estimates. In both high-dimensional experiments the robustness of the considered algorithms is tested by taking the averages and standard deviations of each scheme calculated over five independent runs on the same Monte Carlo test sample.

7.1 Preliminaries

In what follows we are considering the set of equations given by an FBSDE and its corresponding Malliavin derivatives collected in Equation 5.1. We denote the unique pair of triples of random processes solving Equation 5.1 by (X, Y, Z) and $\big(\{D_sX, D_sY, D_sZ\}_{s\in[0,T]}\big)$. We denote discrete partitions of the form

$$\pi^N := \{0 = t_0 < t_1 < \cdots < t_N = T\}, \tag{7.1}$$

whose mesh-size is denoted by $\sup_{t_i\in\pi^N} |t_{i+1} - t_i|$. In order to ease the notation, we put $X_n^\pi, Y_n^\pi, Z_n^\pi$, $D_mX_n^\pi, D_mY_n^\pi, D_mZ_n^\pi$ for the discrete approximations of their continuous counterparts X_{t_n}, Y_{t_n}, Z_{t_n}, $D_{t_m}X_{t_n}, D_{t_m}Y_{t_n}, D_{t_m}Z_{t_n}$ for each $0 \le m, n \le N$. We often use the absolute and relative mean squared errors induced by the discrete schemes, which read as follows

$$\mathcal{E}_i^{y,\pi} := \mathbb{E}\left[|Y_{t_i} - Y_i^\pi|^2\right], \qquad \mathcal{E}_i^{z,\pi} := \mathbb{E}\left[|Z_{t_i} - Z_i^\pi|^2\right], \tag{7.2}$$

$$\rho_i^{y,\pi} := \frac{\mathbb{E}\left[|Y_{t_i} - Y_i^\pi|^2\right]}{\mathbb{E}\left[|Y_{t_i}|^2\right]}, \qquad \rho_i^{z,\pi} := \frac{\mathbb{E}\left[|Z_{t_i} - Z_i^\pi|^2\right]}{\mathbb{E}\left[|Z_{t_i}|^2\right]}. \tag{7.3}$$

It is worth to notice that the error measure defined in the left-hand side of Equation 6.20 used in Theorem 6.2.1 and Theorem 6.3.1, corresponds to $\mathcal{E} = \max_i \mathcal{E}_i^{y,\pi} + \max_j \mathcal{E}_j^{z,\pi}$. We remark that in case $\mathbb{E}\left[|Z_{t_i}|^2\right] = 0$ or $\mathbb{E}\left[|Z_{t_i}|^2\right] = 0$ we report on their absolute errors instead – see Figure 7.5 in particular. Due to the nature of the considered algorithms, the approximation quality at $t_0 = 0$ is exceptionally important which is measured by the following absolute and relative L^2-errors[1] respectively

$$\text{abs. error } Y_0^\pi := |Y_0 - Y_0^\pi|, \qquad \text{abs. error } Z_0^\pi := |Z_0 - Z_0^\pi|, \tag{7.4}$$

$$\text{rel. error } Y_0^\pi := \frac{|Y_0 - Y_0^\pi|}{|Y_0|}, \qquad \text{rel. error } Z_0^\pi := \frac{|Z_0 - Z_0^\pi|}{|Z_0|}. \tag{7.5}$$

We remark that because of the hereby considered, fixed deterministic initial conditions for the forward SDE, these latter errors are not probabilistic. Finally, on top of the errors above we also consider the relative estimated integral errors which are defined by

$$\nu^{y,\pi} := \frac{\mathbb{E}\left[\sum_{i=0}^{N-1} \operatorname{trapz}\left(\int_{t_i}^{t_{i+1}} |Y_t - Y_i^\pi|^2 \mathrm{d}t\right)\right]}{\mathbb{E}\left[\sum_{i=0}^{N-1} \operatorname{trapz}\left(\int_{t_i}^{t_{i+1}} |Y_t^2| \mathrm{d}t\right)\right]}, \quad \nu^{z,\pi} := \frac{\mathbb{E}\left[\sum_{i=0}^{N-1} \operatorname{trapz}\left(\int_{t_i}^{t_{i+1}} |Z_t - Z_i^\pi|^2 \mathrm{d}t\right)\right]}{\mathbb{E}\left[\sum_{i=0}^{N-1} \operatorname{trapz}\left(\int_{t_i}^{t_{i+1}} |Z_t^2| \mathrm{d}t\right)\right]}, \tag{7.6}$$

where trapz() denotes the trapezoidal rule. These discrete errors measure the *average approximation quality* over the whole time horizon.

7.2 Implementation Details

Before presenting the upcoming numerical experiments let us have a few words on the program which the results below were obtained with. First, we briefly explain the code structure. Thereafter, we cover how the hyperparameters (commonly set for all problems below) were chosen, and what impact their choice has on the numerical results. Finally, we discuss the chosen discretization for OSM and MSM.

7.2.1 Code and Platform

Apart from the results of the Forward Deep BSDE method, all results below were obtained with a custom implementation. The Forward Deep BSDE code base was taken from the public github repository of the main author in [7]. The Backward Deep BSDE, OSM and MSM solvers were implemented individually. The implementation is written in TensorFlow 2.0. Automatic differentiation is done by the `.gradient()`, `.jacobian()` methods for `tf.GradientTape()` in eager execution. In order to conserve efficiency all functions are decorated with `@tf.function` decorators exploiting the performance boost of static graph execution. For more information on the structure of TensorFlow we refer to the official guide and the tutorials therein. The code base of this project will be made publicly available after the completion of the project under the author's github repository. All experiments below were run under Google Colaboratory on a randomly assigned cloud of GPUs.

7.2.2 Hyperparameter Selection

The hyperparameters of the considered models can be split into three different categories. We discuss these separately.

[1]Although the notation might be slightly confusing here, we remark that these errors indeed correspond to L^2-norms. Recall from section 1.1 that throughout the whole work $|\cdot|$ denotes the Frobenius norm, which in case of vectors coincides with the Euclidean norm.

Network Architecture

We choose a fully-connected feedforward neural network of the form Equation 3.14 where batch normalization layers – see algorithm 3 – are inserted before the input layer and in between each pair of hidden layers. In order to adhere to good machine learning practices suggested by [28], there is no batch normalization applied in between the last hidden and the output layers. In accordance with other deep learning based BSDE methods, we choose networks with depth of two hidden layers $L = 2$, each containing $N^\ell = 100 + d, \ell = 1, 2$ many neurons depending on the dimensionality of the problem being considered. Since smooth differentiability is a fundamental property of the hereby proposed algorithms, we choose a tanh activation function. We remark that although our empirical experiments showed slightly increased accuracy with the use of ELU (defined in Equation 3.4), we decided to present results with tanh, since the latter satisfies the conditions of both UAT theorems (Theorem 3.2.1, Theorem 3.2.2) presented in chapter 3. As in this work we are only concerned with fixed, deterministic initial conditions $X_0 = x_0$, similarly to the Forward Deep BSDE model, we parametrize the processes at $t_0 = 0$ by single trainable parameters. Finally, we remark that for the results on the Forward Deep BSDE method we used ReLU activations (defined in Equation 3.3) according to the suggestions of the paper which it was proposed in [7].

Training

We choose a Monte Carlo sample size for the underlying forward diffusion (and its Malliavin derivatives) of size $M = 2^{17} = 131072 = \mathcal{O}(10^5)$. The choice for this to be a power of 2 is motivated by better parallelization properties on GPUs. We remark that in case of each upcoming experiment, the forward diffusion is an Arithmetic Brownian Motion whose dynamics – together with its Malliavin derivatives – is analytically solvable by Equation 1.12 and Equation 2.9. The case of forward diffusions without closed form expressions is left for future research. In each experiment we perform mini-batch training with mini-batches of size $2^{10} = 1024$, which means that in each epoch $2^7 = 128$ iterations steps are taken. We use 100 epochs for the training of each time step that implies that in total the optimizer is shown $100 \times 2^{17} = \mathcal{O}(10^7)$ trajectories of the forward dynamics. The optimization step is done with the Adam optimizer introduced in algorithm 2. The decay parameters are chosen according to the default suggestions in [36].

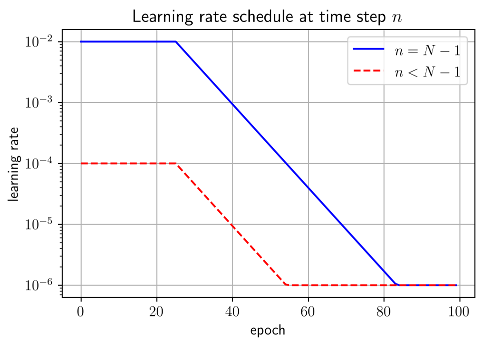


Figure 7.1: Numerical Experiments – Learning Rate Schedule. Learning rate schedule defined in Equation 7.7. Blue curve: time step $n = N - 1$. Red curve: time step $n < N - 1$ initialized by transfer learning. Parameters set according to Table 7.1.

A crucial input of the optimization is the choice of the learning rate. In our empirical experience, this has shown to have a big impact on the accuracy of the estimations, and therefore it needs to be chosen carefully in order to avoid the phenomenon of rolling errors in the backward recursive algorithms. Hereby, we choose an adaptive learning rate schedule proposed by Chen and Wan in [10]. This changes the learning rate according to the following formula for each epoch in $0 \leq e \leq$ total-epochs

$$\eta^{(e)} \leftarrow \eta^{(0)} \times \max\left[10^{-6}; \text{decay-rate}^{\max[\min\{(e-\text{total-epochs}/4)/(350\times\text{total-epochs}/600);1\},0]}\right], \tag{7.7}$$

where η^0 is the chosen initial learning rate and the schedule is floored at 10^{-6}. We choose a decay rate of 10^{-4} for each upcoming problem. For the initial learning rate, in accordance with the Backward Deep BSDE method, we select $\eta^{(0)} = 10^{-2}$ for the $N - 1$'th time steps corresponding to the first regression problem in each model.

Thereafter, we initialize the subsequent time steps' parametrizations by the "optimal" parameter set of their adjacent future neighbour according to $\theta_n \leftarrow \widehat{\theta}_{n+1}$ for each $n = 1, \ldots, N - 2$. Due to this transfer learning trick, we bump the initial learning rate down for previous time steps in the time grid to $\eta^{(0)} = 10^{-4}$. The resulting learning rate schedules are depicted in Figure 7.1. We remark that in our empirical findings, the aforementioned transfer learning trick contributes significantly to the accuracy at time steps $n < N - 1$, confirming the intuition that, due to continuity, the solutions at adjacent time steps should not be far off from each other. This empirical finding is also confirmed in [10, Figure 4]. Finally, we remark that in order to apply a similar logic at t_0 we initialize the values of the trainable parameters in each model by their corresponding explicit schemes.

	hyperparameter	value
network architecture	L	2
	N^ℓ, $\ell = 1, \dots, L$	$100 + d$
	activation	tanh (FWDBSDE: ReLU)
training	optimizer	Adam (default momentums)
	learning rate	defined in Equation 7.7
	initial learning rate	$10^{-2}, n = N-1$; $10^{-4}, n < N-1$
	decay rate	10^{-4}
	epochs	100
	batch size	1024
	M – Monte Carlo sample size	$2^{17} = 131072$
discretization	$\vartheta_y = \vartheta_z$	1/2
	Δt_i, $i = 0, \dots, N-1$	T/N

Table 7.1: Numerical Experiments – Common Hyperparameters

Discretization

Ultimately, we remark that in the following examples we only consider equidistant time partitions whose mesh size is $\Delta t_i = T/N$. For the parameters ϑ_y, ϑ_z in OSM and MSM, we choose the Crank–Nicolson scheme $\vartheta_y = \vartheta_z = 1/2$. We highlight that according to our empirical findings, the Crank–Nicolson yields slightly better accuracy than the completely implicit scheme $\vartheta_y = 1$ and both perform significantly better than the explicit scheme $\vartheta_y = 0$. Investigation on the optimally chosen theta-discretization is left for future research. Finally, in what follows we only present results for a fixed, fine enough time partition and do not analyze the $N \to \infty$ limit behaviour. This is in accordance with other methods in the literature, argued mostly be the computational complexity of deep learning models. Our offline empirical experiments suggest that while increasing the time horizon T and the number of discretization points N the Forward Deep BSDE method diverges or does not fit into memory. For the Backward Deep BSDE method's, the previously described rolling error phenomenon becomes more apparent and accuracy is lost towards $t = 0$. We have experienced that OSM and MSM are prone to similar effects, although to a smaller extent. This coincides with the intuition that one by separating the estimations of the Y- and Z-processes derives smaller and easier learning problems. Nevertheless, the thorough the investigation of the vanishing mesh-size behaviour is left for future research.

7.3 One-Dimensional Problem

Taking advantage of the possibility for insightful visualizations in one-dimensional problems we first present results on the following FBSDE system which is taken from [43]

$$\mu = 0, \quad \sigma = 1, \tag{7.8a}$$
$$f(t,x,y,z) = yz - z + 2.5y - \sin(t+x)\cos(t+x) - 2\sin(t+x), \tag{7.8b}$$
$$g(x) = \sin(T+x). \tag{7.8c}$$

Notice that the underlying forward diffusion reduces to a shifted Brownian motion. The analytical solution of the FBSDE is given by

$$X_t = X_0 + W_t, \quad (Y_t, Z_t) = (\sin(t+X_t), \cos(t+X_t)), \quad 0 \le t \le T. \tag{7.9}$$

The equation is considered with terminal time $T = 0.5$, the forward diffusion starting off from $X_0 = 1$ and we take $N = 20$ discretization points in the time partition.

Since $d = 1$, one can evaluate the approximation accuracy at each point time by comparing the fitted deterministic mappings $(t_n, X_{t_n}) \mapsto (Y_n^\pi, Z_n^\pi)$ against their corresponding reference curves given by Equation 7.9. This is done in Figure 7.2 where such mappings are studied in case of OSM, MSM and the Backward Deep BSDE method for the two most important learning problems in the time sequence.[2] In the top row we see the approximation accuracy at the time step closest to termination for both Y and Z. In all backward recursive algorithms this is the very first approximation step – after the collection of the

[2]We remark that due to the higher orders in the errors of the Forward Deep BSDE scheme's approximations, we excluded its result from this presentation.

analytical terminal conditions – which is subsequently an input for all previous time steps' regressions. Hence, its accuracy is fundamental with respect to the overall approximation quality of each numerical scheme. Accordingly, we can see in the top left plot of Figure 7.2 that all the three algorithms manage to capture the deterministic mapping $\left(t_{N-1}, X_{t_{N-1}}\right) \mapsto Y_{t_{N-1}}$ accurately over the discovered spatial area. Nevertheless, we can also see in the top right plot that the approximation accuracy for $Z_{t_{N-1}}$ is less good at the scarcely discovered regions of space: namely the two edges of the curves. These correspond to areas in the space domain where the forward diffusion seldom arrives and whose *importance* is less accentuated during the training. Notwithstanding, we see that OSM and MSM both perform slightly better at these sparsely sampled regions than the Backward Deep BSDE method which can be contributed to the direct training applied on Z through the Malliavin problem formulation.

Moreover, in the bottom graphs of Figure 7.2 we see the approximation quality at $t = t_1$, i.e. the last time step in the recursive time sequence where deterministic mappings are fitted. Here, the left plot shows that although all algorithms capture accurate approximations for Y in the most densely discovered spatial area of the domain, their errors also significantly increase during the backward recursion at the edges of the curve. The one-step schemes OSM and Backward Deep BSDE method seem to perform slightly better at these extremes than MSM. However, on the right part of the bottom row in Figure 7.2 we see an opposite behaviour for Z. Namely, the Backward Deep BSDE method's approximations roll a large error in the control around the edges of the spatial region and the quality of the approximations significantly deteriorates during the training of the time sequence. This confirms the observations in section 4.4. On the contrary, we see that OSM improves the approximation quality at Z_{t_1} and MSM performs even better. This incites hope that the direct training on the control process through Malliavin calculus can indeed mitigate the problem of worsening control estimates.

This phenomenon is further illustrated by Figure 7.3. Here we can see the relative mean squared errors of the approximations over the whole discretized time window. In the left plot we see that the Backward Deep BSDE method rolls an increasing error in the Y-process towards $t = 0$. On the other hand OSM and MSM mitigate this impact and around time steps closest to $t = 0$ their approximation accuracy is nearly an order of magnitude better than that of the Backward Deep BSDE method. In the right-hand side we see a very similar pattern for the control process. However, on top of OSM and MSM performing much better at the rolling errors throughout the recursive time window, we can also observe that their approximation quality at t_{N-1} is significantly better than the Backward Deep BSDE's, giving further confirmation of the meaningfulness of the Malliavin formulation. Moreover, let us also highlight the difference between the magnitudes of relative errors the Backward Deep BSDE method gives for the Y- and Z-processes. Indeed, we see that the approximation error of the control (green curve on the right-hand side of Figure 7.3) is roughly an order of magnitude larger compared to the left-hand side corresponding to the Y-process. This is further empirical confirmation that the Backward Deep BSDE method is not suited to deal with the control problem of the BSDE. This phenomenon seems is mitigated to a great extent in case of OSM and MSM.

Additionally, the approximation quality over the whole time horizon is also depicted in Figure 7.4, where we can see approximated against analytical trajectories of each model for two independent realizations of the underlying Brownian motion. In the top group we see the pathwise trajectories of the Y-process. These plots show that OSM and MSM both perform better throughout the whole time horizon than the Deep BSDE solvers. It is worth to notice the large inaccuracies of the Forward Deep BSDE method for $t \geq 0$ which indeed confirms that the forward algorithm only manages to give accurate estimations at $t = 0$. One can see a similar pattern in the pathwise trajectories of the Z-process, where the approximation and reference curves for OSM and MSM coincide with their analytical reference, whereas in case of the Backward Deep BSDE method we see a significant gap between the curves at time steps close to termination already.

We conclude the discussion of the one-dimensional problem by the precise numerical error figures collected in Table 7.2. We highlight that out of the four algorithms OSM performs the best with respect to the approximation error in Y_0. On the other hand MSM excels in the approximation error of Z_0. The proposed algorithms also perform better than the Deep BSDE solvers in the maximum approximation errors $\max_i \rho_i^{y,\pi}, \max_i \rho_i^{z,\pi}$ of both Y and Z. We highlight that the average approximation error $\nu^{z,\pi}$ of the Z-process is nearly an order of magnitude better for the proposed algorithms than for their Deep BSDE counterparts.

7.4 Hamilton-Jacobi-Bellman Equation

Having demonstrated the proposed algorithms relevance in a one-dimensional problem, we now turn to the main focus of this work and investigate high-dimensional equations. First, we consider the so-called Hamilton–Jacobi–Bellman equation which has its roots in dynamic programming and is a fundamental

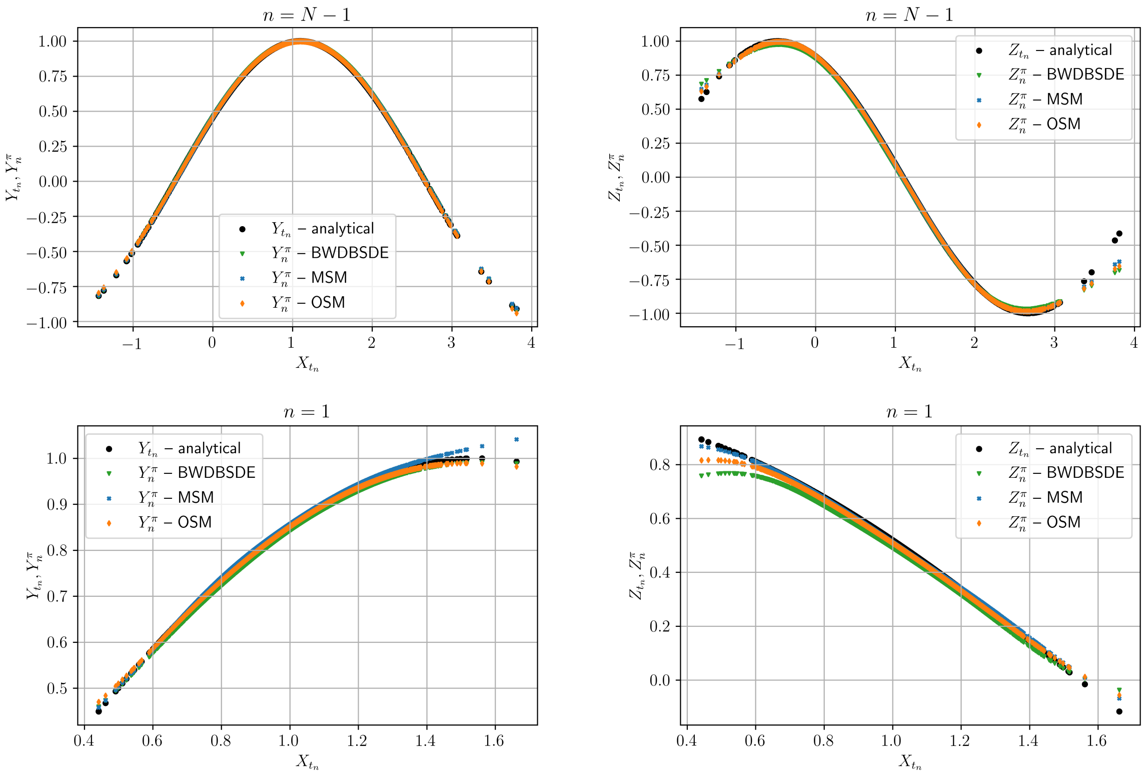


Figure 7.2: One-Dimensional Problem – Deterministic Mappings. Top row: estimations at $t = t_{N-1}$. Left: Y^π_{N-1}; right: Z^π_{N-1}. Bottom row: estimations at $t = t_1$. Left: Y^π_1; right: Z^π_1. Evaluated on a test sample of size $M = 2^{12}$. ($d = 1$, $T = 0.5$, $X_0 = 1$, $N = 20$).

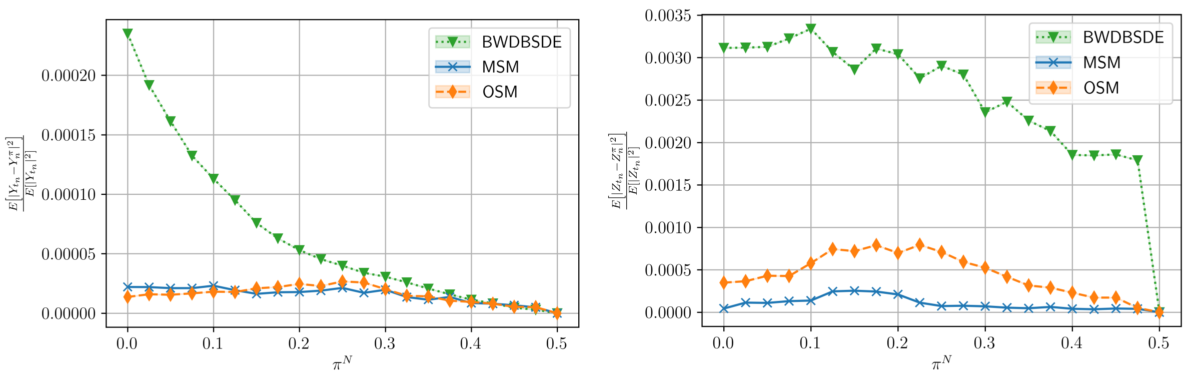
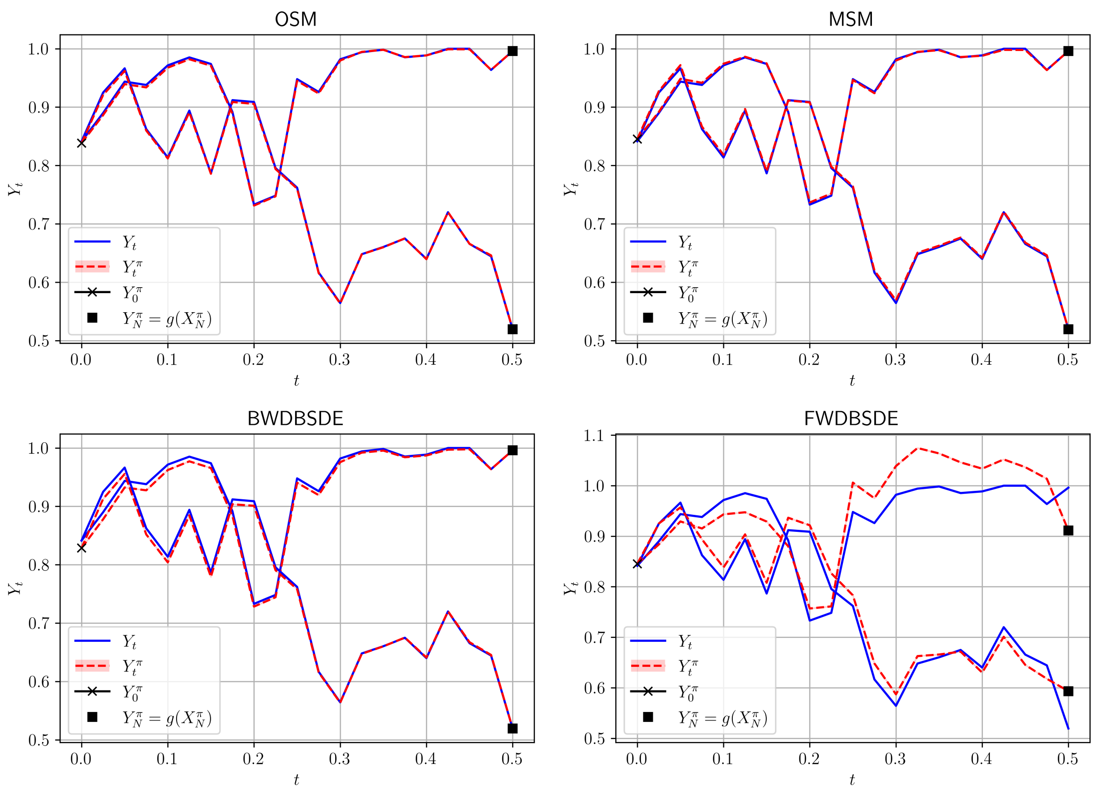
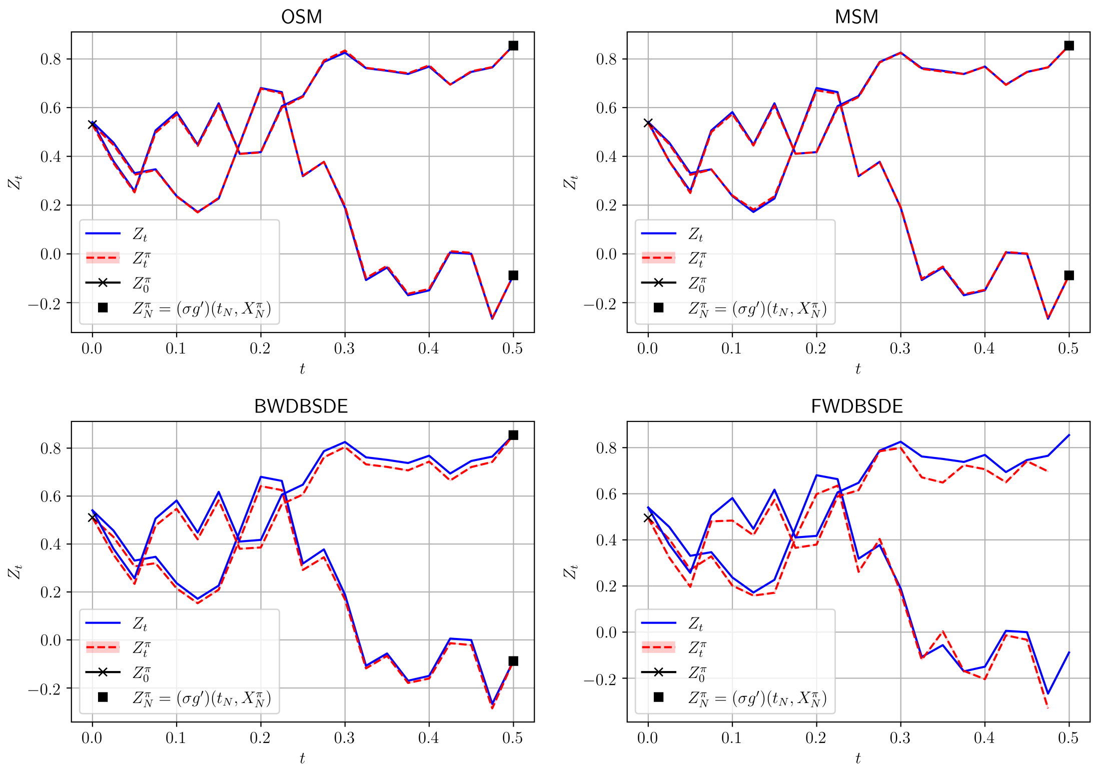


Figure 7.3: One-Dimensional Problem – Relative Errors. Comparison of relative mean squared errors at each time step. Left: Y_{t_n}; right: Z_{t_n}. Calculated on an independent Monte Carlo test sample of size $M = 2^{17}$. ($d = 1$, $T = 0.5$, $X_0 = 1$, $N = 20$).



(a) Y_t



(b) Z_t

Figure 7.4: One-Dimensional Problem – Pathwise Trajectories. Analytical and approximated trajectories for two random realizations of the underlying Brownian motion. Top group: Y_t; bottom group: Z_t. Within each group, from top to bottom, left to right: One-Step Malliavin, Multi-Step Malliavin, Backward Deep BSDE, Forward Deep BSDE. ($d = 1$, $T = 0.5$, $X_0 = 1$, $N = 20$).

	OSM	MSM	BWDBSDE	FWDBSDE
Y_0^π	8.3836E-01	8.4542E-01	8.2857E-01	8.4545E-01
rel. error Y_0^π	3.6985E-03	4.6943E-03	1.5331E-02	4.7293E-03
Z_0^π	5.3022E-01	5.3674E-01	5.1015E-01	4.9501E-01
rel. error Z_0^π	1.8670E-02	6.5883E-03	5.5801E-02	8.3826E-02
$\max_i \rho_i^{y,\pi}$	2.6724E-05	2.3105E-05	2.3503E-04	2.5036E-02
$\max_i \rho_i^{z,\pi}$	7.9368E-04	2.5373E-04	3.3403E-03	NaN
$\nu^{y,\pi}$	1.6229E-05	1.5776E-05	6.2536E-05	2.4161E-03
$\nu^{z,\pi}$	4.4425E-04	1.0238E-04	2.5376E-03	NaN

Table 7.2: One-Dimensional Problem – Numerical Results. Comparison between OSM, MSM, Backward Deep BSDE and the Forward Deep BSDE method. Best performances highlighted in blue. Calculated on an independent Monte Carlo test sample of size $M = 2^{17}$. The reference solution is $(Y_0, Z_0) = (\sin(1), \cos(1)) \simeq (0.8414709848, 0.54030230586)$ up to 10-digit accuracy. ($d = 1$, $T = 0.5$, $X_0 = 1$, $N = 20$.)

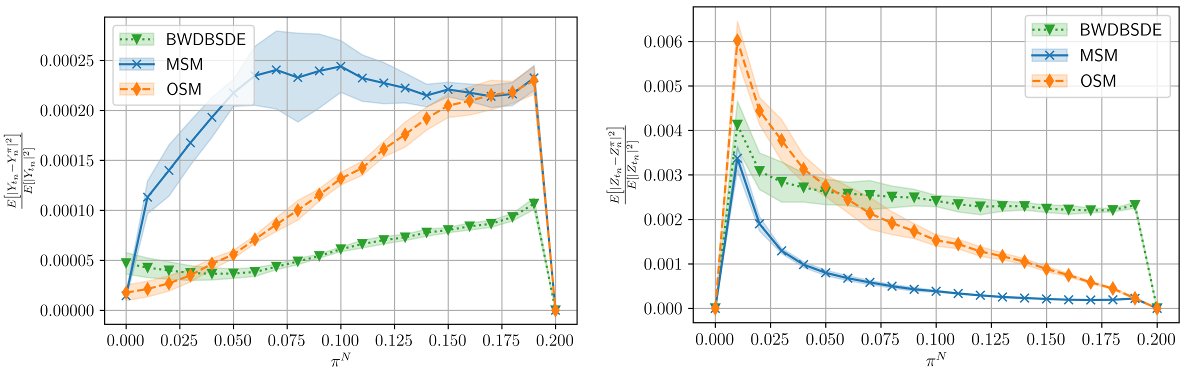


Figure 7.5: Hamilton–Jacobi–Bellman – Relative Errors. Comparison of the means and standard deviations of relative mean squared errors at each time step over 5 independent runs of the algorithms. Left: Y_{t_n}; right: Z_{t_n}. Calculated on an independent Monte Carlo test sample of size $M = 2^{17}$. ($d = 10$, $T = 0.2$, $X_0 = 1_d$, $N = 20$). *(Remark: since* $\mathbb{E}\left[|Z_0|^2\right] = 0$ *for the reference solution, the first point of the right figure is replaced with the absolute error instead.)*

	OSM		MSM		BWDBSDE	
	mean	std	mean	std	mean	std
Y_0^π	2.9508	3.3462E-03	2.9278	1.4344E-03	2.9589	2.4922E-03
rel. error Y_0^π	4.0469E-03	1.1386E-03	3.7933E-03	4.8807E-04	6.8077E-03	8.4800E-04
abs. error Z_0^π	2.8224E-02	5.9020E-03	1.0496E-02	1.8431E-03	1.6231E-02	5.7578E-04
$\max_i \rho_i^{y,\pi}$	2.2999E-04	1.3814E-05	2.6134E-04	2.6361E-05	1.0672E-04	5.7766E-06
$\max_i \rho_i^{z,\pi}$	6.0213E-03	4.3501E-04	3.3738E-03	2.7089E-04	4.1265E-03	5.3368E-04
$\nu^{y,\pi}$	1.3488E-04	4.8917E-06	2.0124E-04	1.1244E-05	6.3589E-05	3.1082E-06
$\nu^{z,\pi}$	9.5775E-04	6.3483E-05	2.8542E-04	1.1835E-05	2.1473E-03	8.2628E-05
time (sec.)	2385.1363	26.0981	2575.4532	12.9191	1172.8388	17.6732

Table 7.3: Hamilton–Jacobi–Bellman – Numerical Results. Comparison between OSM, MSM and Backward Deep BSDE method. Means and standard deviations taken over 5 independent runs of each algorithm. Best performances highlighted in blue. Calculated on an independent Monte Carlo test sample of size $M = 2^{17}$. The reference solution is $Y_0 \simeq 2.9389333435$, $Z_0 \simeq 0.0000000000 \times 1_d$ up to 10-digit accuracy. ($d = 10$, $T = 0.2$, $X_0 = 1_d$, $N = 20$.)

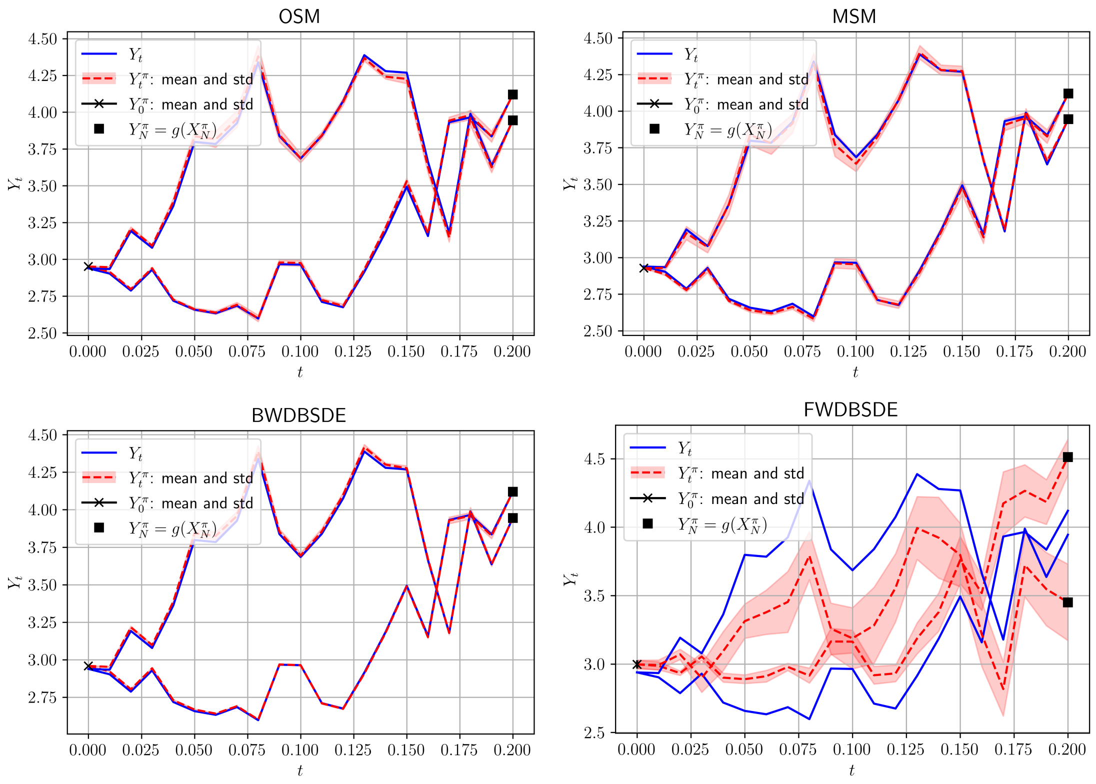


Figure 7.6: Hamilton–Jacobi–Bellman – Pathwise Trajectories of Y_t. Analytical and approximated trajectories for two random realizations of the underlying Brownian motion. Means and standard deviations taken over 5 independent runs of each algorithm. From top to bottom, left to right: One-Step Malliavin, Multi-Step Malliavin, Backward Deep BSDE, Forward Deep BSDE. ($d = 10$, $T = 0.2$, $X_0 = 1_d$, $N = 20$).

problem of stochastic control. In what follows we analyze a formulation of the equation with linear randomness and quadratic control (HJBLQ), i.e. a quadratic terminal condition. This equation poses a significant challenge in numerical approximations as its terminal condition is unbounded and its driver scales quadratically in the control process. Subsequently, the resulting PDE reads as follows

$$\frac{\partial u}{\partial t} + \Delta u - |\nabla u(t,x)|^2 = 0, \quad u(T,x) = |x|^2. \tag{7.10}$$

By the Feynman–Kac relations given by Theorem 1.2.2 we collect the parameters of the corresponding FBSDE system in the form

$$\mu = 0, \qquad \sigma = \sqrt{2} \tag{7.11a}$$

$$f(t,x,y,z) = \frac{1}{2}|z|^2 = \frac{1}{2}\sum_{i=1}^{d} z_i^2, \qquad g(x) = |x|^2 = \sum_{i=1}^{d} x_i^2. \tag{7.11b}$$

We remark that unlike in other Deep BSDE related papers – see, e.g., [7], [59] or [39] – we consider a different, quadratic terminal condition. The reason for this is to have a *semi-analytical* solution, callable over the whole spatial domain, which can be used as a reference to evaluate the accuracy of the considered algorithms. The form of this reference solution is established by the following proposition which creates a link between the solution of the PDE in Equation 7.10 and a system of ordinary differential equations (ODE).

Proposition 7.4.1
Consider Equation 7.10 and assume that the solution satisfies $u \in C^{1,2}([0,T]\times\mathbb{R}^d;\mathbb{R})$. Then the solution can be represented in the following form

$$u(t,x) = x^T P(t)x + x^T Q(t) + R(t), \tag{7.12}$$

where $P:[0,T]\to\mathbb{R}^{d\times d}, Q:[0,T]\to\mathbb{R}^d, R:[0,T]\to\mathbb{R}$ are continuously differentiable functions of time which satisfy the following set of ODEs

$$\begin{aligned}\dot P(t) - \left[P(t)+P^T(t)\right]^2 &= 0,\\ \dot Q(t) - 2\left[P(t)+P^T(t)\right]Q(t) &= 0,\\ \dot R(t) + \mathrm{Tr}\left[P(t)+P^T(t)\right] - |Q(t)|^2 &= 0,\\ P(T) = I_d, Q(T) = 0, R(T) &= 0,\end{aligned} \tag{7.13}$$

where I_d is the d-dimensional identity matrix and 0_d is a d-dimensional vector full of zeroes.

Proof. The proof can be obtained by elementary differentiation of the ansatz $u(t,x)$ in Equation 7.12. Indeed, we collect

$$\frac{\partial u}{\partial t}(t,x) = x^T\dot P(t)x + x^T\dot Q + \dot R(t), \tag{7.14}$$

$$\nabla u(t,x) = \left(P(t)+P^T(t)\right)x + Q(t), \tag{7.15}$$

$$\Delta u(t,x) = \mathrm{Tr}\left[P(t)+P^T(t)\right]. \tag{7.16}$$

Taking the Euclidean norm of the gradient of the ansatz gives

$$|\nabla u(t,x)|^2 = [\nabla u(t,x)]^T[\nabla u(t,x)] = x^T\left[P(t)+P^T(t)\right]^2 x + 2x^T\left[P(t)+P^T(t)\right]Q(t) + |Q(t)|^2. \tag{7.17}$$

Substituting Equation 7.14, Equation 7.17 and Equation 7.16 back into the ansatz of Equation 7.12, we get

$$\begin{aligned}\mathcal{L}(u) &:= \frac{\partial u}{\partial t}(t,x) + \Delta u(t,x) - |\nabla u(t,x)|^2 = x^T\dot P(t)x + x^T\dot Q(t) + \dot R(T) + \mathrm{Tr}\left[P(t)+P^T(t)\right]\\ &- x^T\left[P(t)+P^T(t)\right]^2 x + 2x^T\left[P(t)+P^T(t)\right]Q(t) + |Q(t)|^2\\ &= x^T\left(\dot P(t) - \left[P(t)+P^T(t)\right]^2\right)x + x^T\left(\dot Q(t) - 2\left[P(t)+P^T(t)\right]Q(t)\right)\\ &+ \left(\dot R(t) + \mathrm{Tr}\left[P(t)+P^T(t)\right] - |Q(t)|^2\right).\end{aligned} \tag{7.18}$$

From which we derive that $\mathcal{L}(u)(t,x)=0,\ \forall t,x\in[0,T]\times\mathbb{R}^d$ if and only if all the terms on the right-hand side above are identically equal to zero. Additionally, at terminal time the ansatz reads as

$$u(T,x)=x^TP(T)x+x^TQ(T)+R(T), \tag{7.19}$$

which only satisfies the terminal condition of the PDE for each x in space if $P(T)=I_d, Q(T)=0_d, R(T)=0$. We conclude that the ansatz in Equation 7.12 satisfies the PDE in Equation 7.10 if the following set of equations are all satisfied

$$\begin{aligned}\dot{P}(t)-\left[P(t)+P^T(t)\right]^2&=0,\\ \dot{Q}(t)-2\left[P(t)+P^T(t)\right]Q(t)&=0,\\ \dot{R}(t)+\operatorname{Tr}\left[P(t)+P^T(t)\right]-|Q(t)|^2&=0,\\ P(T)=I_d,Q(T)=0,R(T)&=0.\end{aligned} \tag{7.20}$$

This concludes the proof. □

As a consequence, we have that although the Hamilton-Jacobi-Bellman does not have closed-form analytical solution, one can integrate out the corresponding set of ordinary differential equations in Equation 7.13 numerically on a much finer time grid up to machine precision and substitute the resulting estimations for the time dependent coefficients $P(t),Q(t),R(t)$ back into the ansatz of Equation 7.12. Subsequently, one can gather approximations of almost arbitrary accuracy for each point in space and time. We implemented the set of ODEs in Equation 7.13, and using scipy's odeint function integrated it out on a hundred thousand times finer equidistant time grid with 10^5N points than that of the backward equation. We took the resulting approximations for the coefficients $\{P(t_i),Q(t_i),R(t_i)\}_{0\le i\le 10^5N}$ as ground truths and used their values at each point in the backward equation's time domain[3] to approximate the representation given by Equation 7.12. This is the reference solution against which we compare the numerical estimations obtained with the considered algorithms. Therefore, the *"semi-analytical"* solution of the Hamilton–Jacobi–Bellman equation can be written in the form

$$X_t=\sqrt{2}W_t, \tag{7.21a}$$
$$Y_t=u(t,X_t)=X_t^TP(t)X_t+X_t^TQ(t)+R(t), \tag{7.21b}$$
$$Z_t=\sigma\nabla u(t,X_t)=\sqrt{2}\left(\left[P(t)+P^T(t)\right]X_t+Q(t)\right), \tag{7.21c}$$

where P,Q,R are numerical solutions to the system of ODEs specified in Equation 7.13. The equation is considered in $d=10$ dimensions, with $T=0.2$, $X_0=1_d=(1,\dots,1)$ and $N=20$ discretization points.

In what follows, in order to be able to empirically demonstrate the robustness of the algorithms, we do not merely present results on one trained model according to OSM, MSM and the Backward Deep BSDE solver, but we run each algorithm five times, independently from each other, and calculate the means and standard deviations of these runs on a sixth independent Monte Carlo sample of size $M=2^{17}$. This way we are able to investigate the robustness of the machine learning training phase and empirically discuss how much impact does the stochastic optimization – performed on a stochastic sample – have on the final accuracy and stability of the considered methods.

Unfortunately, due to the high-dimensionality, we cannot directly evaluate the quality of deterministic mappings $(t_n,X_{t_n})\mapsto(Y_{t_n},Z_{t_n})$ visually and we are therefore restricted to the presentation of their mean squared errors instead. This is depicted in Figure 7.5 where the mean of the relative mean squared errors and their standard deviations over 5 independent runs of all the three considered backward algorithms are collected. In the left-hand side we see the error figures of the Y-process over the discretized time window. Here we can observe that the proposed algorithms OSM and MSM perform approximately two times worse at the approximation of the Y-process close to terminal time. Subsequently, going backwards in time, the approximation error of the Backward Deep BSDE method slowly decays, the one of MSM stalls whereas in case of OSM we see a steep decrease. In fact, for the first four points in time OSM reaches a smaller error figure than that of the Backward Deep BSDE method, giving more accurate approximations around $t=0$. Additionally, we see that MSM also exhibits a sharp decrease in the approximation error of the Y-process around $t=0.075$ and by $t=0$ it gives a smaller relative error than the Backward Deep BSDE method. However, we must also notice that the approximations of the Backward Deep BSDE method are more robust in the Y-process, especially compared to MSM which around the middle of the time window displays a large standard deviation in the approximations over independent runs of the algorithm. In this regard both one-step schemes are better and more stable than the multi-step scheme.

[3]Notice that since both time partitions are equidistant, the backward equations time partition is a subset of the ODE's time partition and thus there is no need for interpolation for the time dependent coefficients.

On the other hand, we see an opposite behaviour in the approximation errors of the Z-process which are depicted on the right-hand side of Figure 7.5. First, we highlight that through the direct training provided on Z the error figures at time step $N-1$ are an order of magnitude better for both OSM and MSM than that of the Backward Deep BSDE method. Additionally, one can also observe that as we go back towards $t=0$, the error figures of the Backward Deep BSDE method and OSM increase whereas the ones of MSM stay nearly constant, giving almost an order of magnitude better approximations throughout the whole time window. We remark that MSM exhibits a much better performance at the approximation error of Z than OSM which indeed is in line with the theoretical findings of the previous chapter. In fact, as seen in Theorem 6.2.1 and Theorem 6.3.1, for MSM the cumulative regression error in the Y-process has a $\mathcal{O}(|\pi|)=\mathcal{O}(1/N)$ better coefficient than in case of OSM – see Equation 6.154 against Equation 6.76 in particular. This is confirmed by the numerical experiments above, where we see that the increase in the relative mean squared errors of the Z-process is much flatter for the Multi-Step scheme. We remark that the approximation errors at $t=0$ are better in both proposed schemes than for the Backward Deep BSDE method. Finally, it is worth to mention that the approximations for the Z-process of OSM and MSM show more robustness than in case of the Backward Deep BSDE method. Indeed, we see that the curve corresponding to the Backward Deep BSDE method has a widening standard deviation going towards $t=0$, whereas in case of OSM and especially MSM these error areas are much narrower, suggesting that the proposed schemes depend a lot less on the randomness during training.

Similarly as in the one-dimensional case, we depicted the mean approximated pathwise trajectories and their standard deviations over five independent runs of each algorithm for two independent realizations of the underlying Brownian motion in Figure 7.6. We remark that such plots – due to the high-dimensionality of the problem – are not available in case of the Z-process. One can notice that OSM, MSM and the Backward Deep BSDE method give accurate and stable pathwise estimations, whereas the Forward Deep BSDE method does not manage to capture the true dynamics.

We conclude the discussion of the Hamilton–Jacobi–Bellman equation with the precise numerical results collected in Table 7.3 for all the three algorithms. Similarly as before, here the columns "mean" and "std" correspond to five independent runs of each algorithm evaluated on the same test sample. First of all, it is worth to notice that both OSM and MSM give almost two times better approximations for Y_0 than the Backward Deep BSDE method, whereas all the three methods yield very similar error figures for Z_0 with MSM performing the best. On the other hand, the Backward Deep BSDE method gives almost two times better and more robust estimates than the hereby proposed algorithms for the maximum relative approximation error $\max_i \rho_i^{y,\pi}$ and the average relative approximation error $\nu^{y,\pi}$ of the Y-process. However, we can also see that there is an opposite behaviour in the errors of the Z-process, where in case of MSM the average approximation error $\nu^{z,\pi}$ is almost an order of magnitude better and more robust than that of the Backward Deep BSDE method. We see similar, but slightly worse performance in case of OSM. These observations are in line with the previous findings in Figure 7.5 and suggest that OSM and MSM indeed give more accurate control estimates, however only with a trade-off for somewhat worse approximations in the Y-process.

7.5 Unbounded Solution, Complex Structure

Our last numerical example is an FBSDE system with ABM dynamics, unbounded solution and a complex structure where the driver depends on the product of the backward processes and also scales quadratically in the Y-process. This example is taken from [9, Section 5.2], where they demonstrate that the Backward Deep BSDE method does not converge to the true solution in $d=10$ dimensions. In what follows we show empirically that both OSM and MSM do, giving an order of magnitude better approximations than the standard Backward Deep BSDE method. The FBSDE is given by the following parameters

$$\mu = 0, \quad \sigma = \frac{1}{\sqrt{d}}, \tag{7.22a}$$

$$f(t,x,y,z) = k(x) + \frac{1}{2\sqrt{d}} y \langle 1_d | z \rangle + \frac{1}{2} y^2, \tag{7.22b}$$

$$g(x) = \cos\left(\sum_{i=1}^{d} i x_i\right). \tag{7.22c}$$

The function $k : \mathbb{R}^d \to \mathbb{R}$ above is defined in such a way that the solution of the corresponding PDE admits to

$$u(t,x) = \frac{T-t}{d} \sum_{i=1}^{d} \left[\sin(x_i)\,\mathbb{1}_{x_i<0}(x_i) + x_i \mathbb{1}_{x_1 \geq 0}\right] + \cos\left(\sum_{i=1}^{d} i x_i\right). \tag{7.23}$$

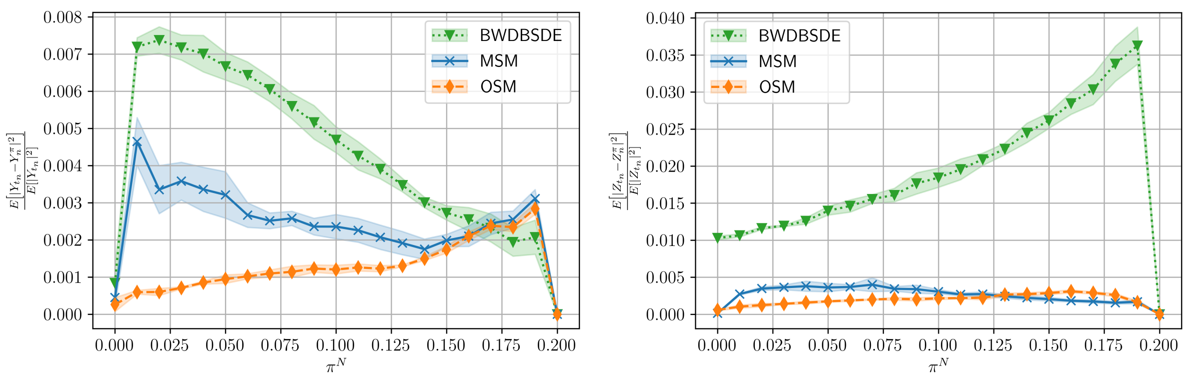


Figure 7.7: Unbounded Solution, Complex Structure – Relative Errors. Comparison of the means and standard deviations of relative mean squared errors at each time step over 5 independent runs of the algorithms. Left: Y_{t_n}; right: Z_{t_n}. Calculated on an independent Monte Carlo test sample of size $M = 2^{17}$. ($d = 10$, $T = 0.2$, $X_0 = 1_d$, $N = 20$).

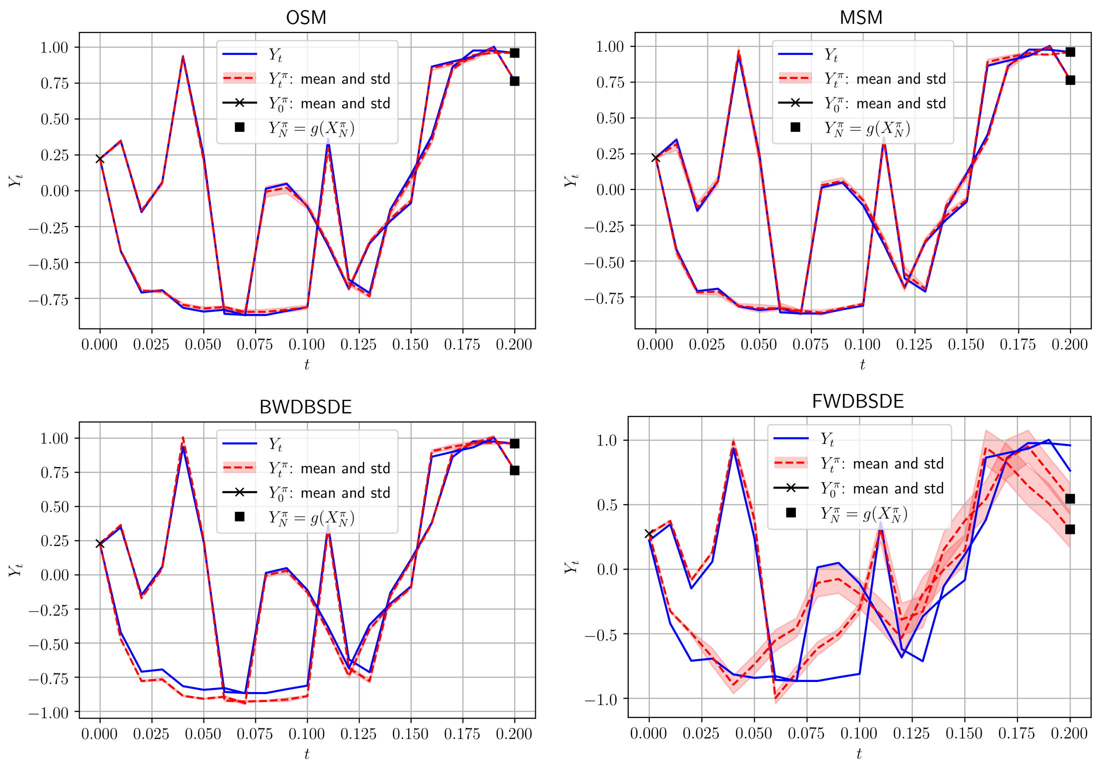


Figure 7.8: Unbounded Solution, Complex Structure – Pathwise Trajectories of Y_t. Analytical and approximated trajectories for two random realizations of the underlying Brownian motion. Means and standard deviations taken over 5 independent runs of each algorithm. From top to bottom, left to right: One-Step Malliavin, Multi-Step Malliavin, Backward Deep BSDE, Forward Deep BSDE. ($d = 10$, $T = 0.2$, $X_0 = 1_d$, $N = 20$).

By the virtue of the generalized Feynman–Kac relations in Theorem 1.2.2, the analytical solution of the FBSDE is thus

$$X_t = X_0 + \frac{1}{\sqrt{d}} W_t, \tag{7.24}$$

$$(Y_t, Z_t) = \left(u(t, X_t), \frac{1}{\sqrt{d}} \nabla_x u(t, X_t) \right). \tag{7.25}$$

It is worth to notice that neither the solution, nor the driver are continuously differentiable at $\{x \colon \exists 0 \leq i \leq d \quad x_i = 0\}$. In the implementation we therefore take the one-sided derivatives when calculating $\nabla_x f$. The equation is considered in $d = 10$ dimensions, with $T = 0.2$, $X_0 = 1_d = (1, \dots, 1)$ and $N = 20$ discretization points.

Similarly as in the case of HJBLQ, in order to evaluate the quality of the approximations of the mapping $(t_n, X_{t_n}) \mapsto (Y_n^\pi, Z_n^\pi)$ we consider the relative mean squared errors depicted in Figure 7.7. From the plot on the left-hand side we can immediately see that the Backward Deep BSDE method indeed fails to capture the true dynamics of the Y-process and rolls an increasing error towards $t = 0$. In fact, at time step t_1 its relative error roughly reaches the magnitude of $\mathcal{O}(10^{-2})$. On the contrary, OSM and MSM both show significantly better convergence and yield nearly three times better error figures. In particular, we observe that OSM exhibits a strictly decreasing error behaviour and by time steps around $t = 0$ it reaches an accuracy almost ten times better than the Backward Deep BSDE method.

Moreover, we can see that in case of the approximations for the Z-process depicted in the right-hand side of Figure 7.7, OSM and MSM show even bigger dominance over the Backward Deep BSDE method. In fact, we deduce that through the direct training applied on the Z-process through the Malliavin formulation, a whole order of magnitude is gained in terms of relative mean squared accuracy. Additionally, OSM and MSM are more stable, giving similar error figures throughout the whole time horizon whereas the Backward Deep BSDE method has an exceptionally large peak in the relative errors close to terminal time. The final accuracy in the control process is an order of magnitude better for both OSM and MSM. Finally, we can also extract from Figure 7.7 that the hereby proposed algorithms – and OSM especially – yield more robust estimations for the backward processes, as the five independent trainings of the model gave error figures narrowly distributed around their respective means. This implies that by splitting up the regression tasks of the backward processes for such a complex problem as in Equation 7.22, one gathers more stable algorithms which are less prone to the inherent stochasticity involved. Similarly as in the previous examples, we collected the pathwise estimations throughout the whole time window for two independent realizations of the underlying Brownian motion in Figure 7.8. Here, on top of seeing further confirmation of the better performance of OSM and MSM, we see that the Forward Deep BSDE method does not manage to capture the dynamics of the BSDE and shows poor accuracy even at $t = 0$.

In order to conclude the discussion on the FBSDE given by Equation 7.22, we finally collect the precise numerical error measures in Table 7.4. We can see that the proposed algorithms show better accuracy and more robustness than the Backward Deep BSDE method in every considered metric. In particular, we highlight that the relative L^2 approximation errors at $t_0 = 0$ are an order of magnitude better for both OSM and MSM than in case of the Backward Deep BSDE method, which exhibits poor $\mathcal{O}(10^{-1})$ relative error in the Z-process. This phenomenon is not restricted to the initial time step, as – suggested by the figures of $\nu^{z,\pi}$ – it can be seen that the average error figures are also an order of magnitude better in the proposed algorithms. These observations are in line with the visual conclusion drawn from Figure 7.7. Additionally, we see a similar pattern in the approximations of the Y-process, where OSM and MSM perform nearly twice as much better than the standard backward algorithm. We conclude that through gathering better estimations for the control process the One-Step and Multi-Step Malliavin schemes proposed in this thesis manage to deal with the very complex FBSDE given by Equation 7.22, where the Backward Deep BSDE method fails.

7.6 Summary of the Results

Ultimately, to conclude the presentation of numerical experiments, let us summarize the empirical findings above. We have seen that both deep learning based schemes OSM and MSM proposed in this thesis result in more accurate approximations for the Z-process than their Deep BSDE counterparts for high-dimensional equations which has been the main goal behind their formulation. Nevertheless, we have also seen that this additionally gained accuracy comes with a trade-off in estimations of the Y-process, where in case of the Hamilton–Jacobi–Bellman equation the Backward Deep BSDE method performed better over the majority of the time horizon than both of the proposed schemes. Motivated by Figure 7.5

	OSM		MSM		BWDBSDE	
	mean	std	mean	std	mean	std
Y_0^π	2.2262E-01	3.5502E-03	2.2229E-01	4.7006E-03	2.2835E-01	1.6379E-03
rel. error Y_0^π	1.4299E-02	7.4833E-03	1.9745E-02	7.6514E-03	2.8023E-02	7.3736E-03
rel. error Z_0^π	2.4236E-02	2.0412E-03	1.1196E-02	3.6085E-03	1.0145E-01	1.3677E-03
$\max_i \rho_i^{y,\pi}$	2.8399E-03	1.5082E-04	4.7348E-03	4.7556E-04	7.3814E-03	3.5887E-04
$\max_i \rho_i^{z,\pi}$	3.1281E-03	1.8654E-04	4.1591E-03	8.0439E-04	3.6252E-02	2.5125E-03
$\nu^{y,\pi}$	1.3533E-03	2.7770E-05	2.5549E-03	1.6672E-04	4.5282E-03	1.9526E-04
$\nu^{z,\pi}$	1.9527E-03	6.1691E-05	2.6506E-03	1.7378E-04	1.8975E-02	8.0768E-04
time (sec.)	2533.0447	142.5024	2799.7629	136.1857	1326.4776	87.4576

Table 7.4: Unbounded Solution, Complex Structure – Numerical Results. Comparison between OSM, MSM and Backward Deep BSDE method. Means and standard deviations taken over 5 independent runs of each algorithm. Best performances highlighted in blue. Calculated on an independent Monte Carlo test sample of size $M = 2^{17}$. The reference solution is $Y_0 \simeq 0.2221267563$, $Z_0 \simeq$[0.3224749004, 0.6386252454, 0.9547755904, 1.2709259355, 1.5870762805, 1.9032266255, 2.2193769706, 2.5355273156, 2.8516776606, 3.1678280057] up to 10-digit accuracy. ($d = 10$, $T = 0.2$, $X_0 = 1_d$, $N = 20$.)

the relative error of OSM in Y is smaller for the time steps around t_0, a possible remediation of this phenomenon would be to use a hybrid of OSM and MSM where Y is approximated by a one-step scheme and Z by a multi-step one. Whether such a fusion of the two discretizations would manage to improve the results is an open question left for future research.

Moreover, we have also seen empirical evidence on the error bounds established in Theorem 6.2.1 and Theorem 6.3.1. Namely, in case of the HJBLQ equation – see the right figure in Figure 7.5 –, we could observe that MSM reaches a much smaller increasing error figure in the approximation of the Z-process. This is in accordance with the theoretical bounds obtained in Equation 6.76 and Equation 6.154, where in case of MSM the cumulative regression errors of the Y-process contribute an $\mathcal{O}(1/N)$ factor less to the worst approximation error for the control process.

Finally, we need to mention one more aspect which we have not discussed before, namely the execution time of the training of each algorithm. Not surprisingly, out of the four algorithms considered in this chapter the Forward Deep BSDE method is trained the fastest, however, as it has been shown it only gives accurate approximations at $t_0 = 0$. Comparing OSM, MSM against Backward Deep BSDE method, one would intuitively assume that by splitting up the regression tasks of the Y- and Z-processes and performing twice as many regressions at each time step, the proposed algorithms should be roughly twice as much slower. This is confirmed by the last rows in Table 7.3 and Table 7.4. We emphasize that these numbers do not provide a precise fair comparison, as all experiments were run under Google Colaboratory on randomly assigned GPUs – i.e. there is no guarantee that the compared experiments were run on the same hardware. Nonetheless, we report on their figures to give comparative ballpark estimates on the execution times. A possible mitigation of this phenomenon would be not to perform full epoch training for the interior time steps initialized by the transfer learning trick. In fact, results from the literature suggest – see, e.g., [52] – that in such a recursive backward scheme when transfer learning is applied as in Equation 5.19, a few iterations over the whole dataset may suffice to provide accurate approximations. We remark that our offline numerical experiments confirmed this result, however also note that the thorough testing and implementation of this idea are left for future research.

Discussion

In this chapter we conclude the discussion of the novel proposed methods. First we summarize our findings and collect the main results of this work. Finally, we lay out future directions for further development of the proposed algorithms, and list other problems where such approaches may turn out to be useful.

7.7 Concluding Summary

In this thesis we considered a general FBSDE system given by Equation 1.56. We proposed a novel approach which takes the evolution of the Z-process provided by Malliavin calculus into account using deep learning. The work can be summarized as follows.

In chapter 1 we gave an introduction to the concept of backward stochastic differential equations. We motivated BSDEs as a non-linear extension to forward SDEs given by the martingale representation theorem. We showed the well-posedness of the problem in Theorem 1.4.1 under general assumptions on the underlying randomness. Thereafter we explained the concept of forward backward stochastic differential equations and presented their inherent connection with second-order parabolic PDEs through the generalized Feynman–Kac relations in Theorem 1.5.2.

In chapter 2 we explained the core concepts of Malliavin calculus, mostly focusing on the derivative operator. We established a connection between SDEs and their Malliavin derivatives and showed in Theorem 2.4.1 that the Malliavin derivative of the solution of a forward SDE satisfies a linear SDE itself. In a similar fashion we demonstrated an analogous connection for BSDEs and established that the Malliavin derivatives of the solution pair of BSDEs satisfy a linear BSDE themselves given by Theorem 2.5.1. This theorem was the basis of the algorithmic formulations in chapter 5, as it provides a natural BSDE dynamics for the Z-process which additionally is also continuous provided by Theorem 2.5.2.

This was followed by chapter 4 where we explained the main ideas behind classical numerical approaches to solve an FBSDE system. We derived classical recursive conditional expectation schemes for BSDEs in the Euler scheme Equation 4.13 and the theta-scheme Equation 4.16. We covered the concept of Least-Squares Monte Carlo regression which was the numerical method of this work to approximate conditional expectations. Finally, we explained the key ideas behind the recently proposed class of BSDE solvers built on deep learning, developed to deal with high-dimensional FBSDE problems. We covered the main advantages and drawbacks of these algorithms and discussed that neither of them manages to yield accurate approximations for the Z process over the whole time horizon.

Motivated by this observation, we derived the novel numerical methods proposed in this work, and explained the discrete One-Step Malliavin and Multi-Step Malliavin schemes in chapter 5. These algorithms are built on neural network LSMC regression and approximate separate conditional expectations for the Y- and Z-processes recursively, backwards in time. Both algorithms exploit the linear BSDE dynamics of the Z-process given by Malliavin calculus. The Malliavin derivatives (DY, DZ) in Equation 2.19 are approximated by the Malliavin chain rule that is enabled by the fact that neural networks are differentiable, universal function approximators which are dense in Sobolev spaces provided by Theorem 3.2.2.

Subsequently, in chapter 6 we analyzed the convergence of the proposed schemes and proved that they are consistent, meaning that their approximation errors vanish in the limit of infinitely small mesh-sizes. In the first part of the chapter we proved a consistency formula for the OSM scheme where the representation errors corresponding to the Y- and Z-processes were split up to different lemmas. We remark that the error bounds corresponding to the control process, and consequently the final error bounds in Theorem 6.2.1 were only stated under rather strict assumptions stated in Assumption 6.1.2 for special forward diffusion dynamics. Thereafter, we extended these results to the consistency of MSM and in Theorem 6.3.1 showed that one by taking multiple discretization points into account gains an $\mathcal{O}(1/N)$ convergence factor in the approximation error of the Z process through the cumulative regression errors of the Y-process. To conclude the chapter we compared the error bounds we obtained to the ones established for Backward Deep BSDE methods.

Finally, in chapter 7 we presented numerical experiments demonstrating the performance of the proposed algorithms on multiple high-dimensional problems. We have observed empirical confirmation of the error bounds obtained in Theorem 6.2.1 and Theorem 6.3.1, and seen that the multi-step scheme indeed manages to mitigate the interdependency of regression errors in the approximation of the control process. In the final example of the chapter, we presented numerical solutions on a fairly complex high-dimensional problem for which the standard Backward Deep BSDE method is known to fail in $d \geq 10$ dimensions. We have shown that the hereby proposed methods OSM and MSM manage to overcome this difficulty and provide an order of magnitude better approximations for the Z-process in $d = 10$ dimensions.

7.8 Future Research

Our findings have several straightforward generalizations in which they could be improved or extended. Out of these we would hereby mention a few.

First and foremost, we have seen that the error bounds in Theorem 6.2.1 and Theorem 6.3.1 were only proven under the assumptions that the underlying forward diffusion is an Arithmetic Brownian Motion, and the driver of the BSDE is independent of Z. In order to gain a deeper understanding of OSM and MSM it is of fundamental importance to generalize these results to the case of general, Z-dependent drivers and forward Itô-processes satisfying the minimal requirements of well-posedness. Additionally, relaxations of the smoothness assumptions in Assumption 2.5.1 would also be desirable for the methods to be more widely applicable. We remark that the Malliavin differentiability of BSDEs is well-established under more relaxed assumption than those in Assumption 2.5.1 – see, e.g., [24] or [23] –, and that for the purpose of this work we only restricted our analysis to the stronger conditions to be able to provide rigorous error bounds for the arising discrete schemes. Nevertheless, the thorough investigation of these theoretical questions will be in the central scope of future research.

On the numerical analysis side, we would highlight two significant improvements from which the proposed algorithms could benefit immensely. First, we have seen that due to the nature of sequential backward recursive optimizations, OSM and MSM are approximately twice as much slower than the Backward Deep BSDE method. However, empirical results in the literature suggest – see, e.g., [52] – that by using the transfer learning initialization defined in Equation 5.19, one can significantly reduce the number of optimization steps needed for the pretrained steps in the recursion. Developing an adaptive optimization scheme which exploits this phenomenon could reduce the computational complexity of OSM and MSM to a great extent and close in the gap which – in terms of speed – is currently in favour of the Backward Deep BSDE method. Second, recall that for the numerical experiments in chapter 7 we put $\vartheta_y = \vartheta_z = 1/2$ for both schemes. The detailed investigation of the impact of different theta-discretizations is left for future research both theoretically and numerically.

As we mentioned before the main goal of this work was to obtain more accurate control estimates. One field of application where this is of fundamental importance is in finance, where the BSDE can be interpreted as the simultaneous solution of an option's pricing and hedging problems. Modifications of Deep BSDE solvers have been applied to tackle the american option pricing problem and shown encouraging successes in obtaining accurate delta hedging in dimension up to 100 – see, e.g., [10] or [9]. American options themselves follow a so-called reflected BSDE where the continuation value is driven by a BSDE. It can be shown – see, e.g., [60] – that the control process Z of reflected BSDEs admit to a similar relation with the Malliavin derivative of Y as in Theorem 2.5.1. Therefore, it would be interesting to see how the methods proposed in this thesis compare to other algorithms developed to deal with the delta hedging problem. Additionally, from a financial point of view, our methods could also be extended to take the linear BSDEs driving higher-order Malliavin derivatives of the solution pair into account. Indeed, as we have mentioned it before, it can be shown that the higher-order Malliavin derivatives of the solution pair of the BSDE admit to similar dynamics as in Theorem 2.5.1. Consequently, the encouraging results shown in this work incite hope that similar methods could be used to obtain accurate estimates for higher-order sensitivities (greeks) in high-dimensional option pricing.

Last but not least, it would also be exciting to see how the methods proposed hereby would be able to deal with the most complicated case of 2BSDEs (corresponding to fully non-linear PDEs) where there is additional layer of stochasticity driving the control process. We remark that the Forward Deep BSDE method has already been extended to the case of 2BSDEs in [61] showing promising empirical results. Nevertheless, it needs to be mentioned that the theory of 2BSDEs is fairly underdeveloped and there are many open questions yet to be fully understood for numerical methods to be able to approximate them accurately in high-dimensions. Whether the Malliavin formulation proposed in this work could provide any further insight in this quest is an open question left for future research.

Bibliography

[1] Jean-Michel Bismut. Intégrales convexes et probabilités. *Journal of Mathematical Analysis and Applications*, 42(3):639 – 673, 1973. doi:10.1016/0022-247X(73)90170-4.

[2] Etienne Pardoux and Shige Peng. Backward stochastic differential equations and quasilinear parabolic partial differential equations. In *Stochastic partial differential equations and their applications*, pages 200–217. Springer, 1992. URL: https://link.springer.com/content/pdf/10.1007/BFb0007334.pdf.

[3] Emmanuel Gobet, Jean-Philippe Lemor, and Xavier Warin. A regression-based monte carlo method to solve backward stochastic differential equations. *Ann. Appl. Probab.*, 15(3):2172–2202, 08 2005. doi:10.1214/105051605000000412.

[4] Emmanuel Gobet and Plamen Turkedjiev. Linear regression mdp scheme for discrete backward stochastic differential equations under general conditions. *Mathematics of Computation*, 85(299):1359–1391, 2016. doi:10.1090/mcom/3013.

[5] Jianfeng Zhang. A numerical scheme for bsdes. *Ann. Appl. Probab.*, 14(1):459–488, 02 2004. doi:10.1214/aoap/1075828058.

[6] Bruno Bouchard, Romuald Elie, and Nizar Touzi. Discrete-time approximation of bsdes and probabilistic schemes for fully nonlinear pdes. *Radon Series Comp. Appl. Math., Advanced Financial Modelling*, 8:91–124, 2009. URL: https://hal.archives-ouvertes.fr/hal-00630797.

[7] Jiequn Han, Arnulf Jentzen, and Weinan E. Solving high-dimensional partial differential equations using deep learning. *Proceedings of the National Academy of Sciences*, 115(34):8505–8510, 2018. doi:10.1073/pnas.1718942115.

[8] Haojie Wang, Han Chen, Agus Sudjianto, Richard Liu, and Qi Shen. Deep learning-based bsde solver for libor market model with application to bermudan swaption pricing and hedging, 2018. arXiv:1807.06622.

[9] Côme Huré, Huyên Pham, and Xavier Warin. Some machine learning schemes for high-dimensional nonlinear pdes, 2019. arXiv:1902.01599.

[10] Yangang Chen and Justin W. L. Wan. Deep neural network framework based on backward stochastic differential equations for pricing and hedging american options in high dimensions, 2019. arXiv:1909.11532.

[11] W. Rudin. *Functional Analysis*. International series in pure and applied mathematics. McGraw-Hill, 1991. URL: https://books.google.nl/books?id=Sh_vAAAAMAAJ.

[12] Jianfeng Zhang. *Backward stochastic differential equations*. doi:10.1007/978-1-4939-7256-2.

[13] Bernt Øksendal. *Stochastic differential equations*. Springer, 6 edition, 2003. doi:10.1007/978-3-642-14394-6.

[14] E. Pardoux and S.G. Peng. Adapted solution of a backward stochastic differential equation. *Systems & Control Letters*, 14(1):55 – 61, 1990. doi:10.1016/0167-6911(90)90082-6.

[15] Alexander Steinicke. Backward stochastic differential equations and applications. Vienna Seminar in Mathematical Finance and Probability, 2017. URL: https://www.wu.ac.at/fileadmin/wu/d/i/statmath/Research_Seminar/SS_2017/steinicke_slides.pdf.

[16] N. El Karoui, S. Peng, and M. C. Quenez. Backward stochastic differential equations in finance. *Mathematical Finance*, 7(1):1–71, 1997. doi:10.1111/1467-9965.00022.

[17] David Nualart. *The Malliavin calculus and related topics*, volume 1995. Springer, 2 edition, 2006. doi:10.1007/3-540-28329-3.

[18] Giulia Di Nunno, Bernt Karsten Øksendal, and Frank Proske. *Malliavin calculus for Lévy processes with applications to finance*, volume 2. Springer, 1 edition. doi:10.1007/978-3-540-78572-9.

[19] Eulalia Nualart. Lectures on malliavin calculus and its applications to finance. Lecture Series at the University of Wisconsin (Madison), 2009. URL: http://www.math.wisc.edu/~kurtz/NualartLectureNotes.pdf.

[20] Bernt Øksendal. An introduction to malliavin calculus with applications to economics. Lectures given at the Norwegian School of Economics and Business Administration, Bergen, 1997. URL: https://mat.ug.edu.pl/~mwrzosek/Oksendal.pdf.

[21] Plamen Turkedjiev. Two algorithms for the discrete time approximation of markovian backward stochastic differential equations under local conditions. *Electron. J. Probab.*, 20:49 pp., 2015. doi:10.1214/EJP.v20-3022.

[22] Yaozhong Hu, David Nualart, and Xiaoming Song. Malliavin calculus for backward stochastic differential equations and application to numerical solutions. *Ann. Appl. Probab.*, 21(6):2379–2423, 12 2011. doi:10.1214/11-AAP762.

[23] Thibaut Mastrolia, Dylan Possamaï, and Anthony Réveillac. On the malliavin differentiability of bsdes. *Ann. Inst. H. Poincaré Probab. Statist.*, 53(1):464–492, 02 2017. doi:10.1214/15-AIHP723.

[24] Jin Ma and Jianfeng Zhang. Representation theorems for backward stochastic differential equations. *Ann. Appl. Probab.*, 12(4):1390–1418, 11 2002. doi:10.1214/aoap/1037125868.

[25] Justin Sirignano and Konstantinos Spiliopoulos. Dgm: A deep learning algorithm for solving partial differential equations. *Journal of Computational Physics*, 375:1339–1364, Dec 2018. doi:10.1016/j.jcp.2018.08.029.

[26] M. Raissi, P. Perdikaris, and G.E. Karniadakis. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational Physics*, 378:686 – 707, 2019. doi:10.1016/j.jcp.2018.10.045.

[27] Remco van der Meer, Cornelis Oosterlee, and Anastasia Borovykh. Optimally weighted loss functions for solving pdes with neural networks, 2020. arXiv:2002.06269.

[28] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. http://www.deeplearningbook.org.

[29] Michael A Nielsen. *Neural networks and deep learning*, volume 2018. Determination press San Francisco, CA, 2015. http://neuralnetworksanddeeplearning.com/.

[30] Christopher M Bishop. *Pattern recognition and machine learning*. Springer, 2006. URL: https://www.microsoft.com/en-us/research/uploads/prod/2006/01/Bishop-Pattern-Recognition-and-Machine-Learning-2006.pdf.

[31] George Cybenko. Approximation by superpositions of a sigmoidal function. *Mathematics of control, signals and systems*, 2(4):303–314, 1989. doi:10.1007/BF02551274.

[32] Kurt Hornik. Approximation capabilities of multilayer feedforward networks. *Neural Networks*, 4(2):251 – 257, 1991. doi:10.1016/0893-6080(91)90009-T.

[33] Kurt Hornik, Maxwell Stinchcombe, and Halbert White. Universal approximation of an unknown mapping and its derivatives using multilayer feedforward networks. *Neural Networks*, 3(5):551 – 560, 1990. doi:10.1016/0893-6080(90)90005-6.

[34] Allan Pinkus. Approximation theory of the mlp model in neural networks. *Acta Numerica*, 8:143–195, 1999. doi:10.1017/S0962492900002919.

[35] Patrick Kidger and Terry Lyons. Universal approximation with deep narrow networks, 2019. arXiv:1905.08539.

[36] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2014. arXiv:1412.6980.

[37] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift, 2015. arXiv:1502.03167.

[38] Atılım Günes Baydin, Barak A. Pearlmutter, Alexey Andreyevich Radul, and Jeffrey Mark Siskind. Automatic differentiation in machine learning: A survey. *J. Mach. Learn. Res.*, 18(1):5595–5637, January 2017. URL: http://jmlr.org/papers/v18/17-468.html.

[39] Quentin Chan-Wai-Nam, Joseph Mikael, and Xavier Warin. Machine learning for semi linear pdes. *Journal of Scientific Computing*, 79(3):1667–1712, Jun 2019. doi:10.1007/s10915-019-00908-3.

[40] Weinan E, Jiequn Han, and Arnulf Jentzen. Algorithms for solving high dimensional pdes: From nonlinear monte carlo to machine learning, 2020. arXiv:2008.13333.

[41] Peter E Kloeden and Eckhard Platen. *Numerical Solution of Stochastic Differential Equations*. Springer, 1992. doi:10.1007/978-3-662-12616-5.

[42] Weidong Zhao, Jinlei Wang, and Shige Peng. Error estimates of the theta-scheme for backward stochastic differential equations. *Discrete & Continuous Dynamical Systems - B*, 12(1531-3492):905, 2009. doi:10.3934/dcdsb.2009.12.905.

[43] Weidong Zhao, Yang Li, and Guannan Zhang. A generalized θ-scheme for solving backward stochastic differential equations. *Discrete & Continuous Dynamical Systems - B*, 17(1531-3492):1585, 2012. doi:10.3934/dcdsb.2012.17.1585.

[44] M. J. Ruijter and C. W. Oosterlee. A fourier cosine method for an efficient computation of solutions to bsdes. *SIAM Journal on Scientific Computing*, 37(2):A859–A889, 2015. doi:10.1137/130913183.

[45] Francis A. Longstaff and Eduardo S. Schwartz. Valuing American Options by Simulation: A Simple Least-Squares Approach. *The Review of Financial Studies*, 14(1):113–147, 06 2015. doi:10.1093/rfs/14.1.113.

[46] Christian Bender and Jessica Steiner. Least-squares monte carlo for backward sdes. In René A. Carmona, Pierre Del Moral, Peng Hu, and Nadia Oudjane, editors, *Numerical Methods in Finance*, pages 257–289, Berlin, Heidelberg, 2012. Springer Berlin Heidelberg. doi:10.1007/978-3-642-25746-9_8.

[47] Kristoffer Andersson and Cornelis Oosterlee. A deep learning approach for computations of exposure profiles for high-dimensional bermudan options, 2020. arXiv:2003.01977.

[48] Weinan E, Jiequn Han, and Arnulf Jentzen. Deep learning-based numerical methods for high-dimensional parabolic partial differential equations and backward stochastic differential equations. *Communications in Mathematics and Statistics*, 5(4):349–380, Dec 2017. doi:10.1007/s40304-017-0117-6.

[49] Jiequn Han and Jihao Long. Convergence of the deep bsde method for coupled fbsdes. *Probability, Uncertainty and Quantitative Risk*, 5(1):1–33, 2020. doi:10.1186/s41546-020-00047-w.

[50] Maximilien Germain, Huyen Pham, and Xavier Warin. Deep backward multistep schemes for nonlinear pdes and approximation error analysis, 2020. arXiv:2006.01496.

[51] Achref Bachouch, Côme Huré, Nicolas Langrené, and Huyen Pham. Deep neural networks algorithms for stochastic control problems on finite horizon: numerical applications, 2018. arXiv:1812.05916.

[52] Bernard Lapeyre and Jérôme Lelong. Neural network regression for bermudan option pricing, 2019. arXiv:1907.06474.

[53] Lucio Fernandez-Arjona. A neural network model for solvency calculations in life insurance, 2020. arXiv:2005.02318.

[54] Vikranth Lokeshwar, Vikram Bhardawaj, and Shashi Jain. Neural network for pricing and universal static hedging of contingent claims, 2019. arXiv:1911.11362.

[55] Emmanuel Gobet and Plamen Turkedjiev. Approximation of backward stochastic differential equations using malliavin weights and least-squares regression. *Bernoulli*, 22(1):530–562, 02 2016. doi:10.3150/14-BEJ667.

[56] Philippe Briand and Céline Labart. Simulation of bsdes by wiener chaos expansion. *Ann. Appl. Probab.*, 24(3):1129–1171, 06 2014. doi:10.1214/13-AAP943.

[57] Yuki Izumi. Higher order differentiability of solutions to backward stochastic differential equations. *Stochastics*, 90(1):102–150, 2018. doi:10.1080/17442508.2017.1315119.

[58] Peter Imkeller and Gonçalo Dos Reis. Path regularity and explicit convergence rate for bsde with truncated quadratic growth. *Stochastic Processes and their Applications*, 120(3):348 – 379, 2010. doi:10.1016/j.spa.2009.11.004.

[59] Maziar Raissi. Forward-backward stochastic neural networks: Deep learning of high-dimensional partial differential equations, 2018. arXiv:1804.07010.

[60] Bruno Bouchard and Jean-François Chassagneux. Discrete-time approximation for continuously and discretely reflected bsdes. *Stochastic Processes and their Applications*, 118(12):2269 – 2293, 2008. doi:https://doi.org/10.1016/j.spa.2007.12.007.

[61] Christian Beck, Weinan E, and Arnulf Jentzen. Machine learning approximation algorithms for high-dimensional fully nonlinear partial differential equations and second-order backward stochastic differential equations. *Journal of Nonlinear Science*, 29(4):1563–1619, Aug 2019. doi:10.1007/s00332-018-9525-3.