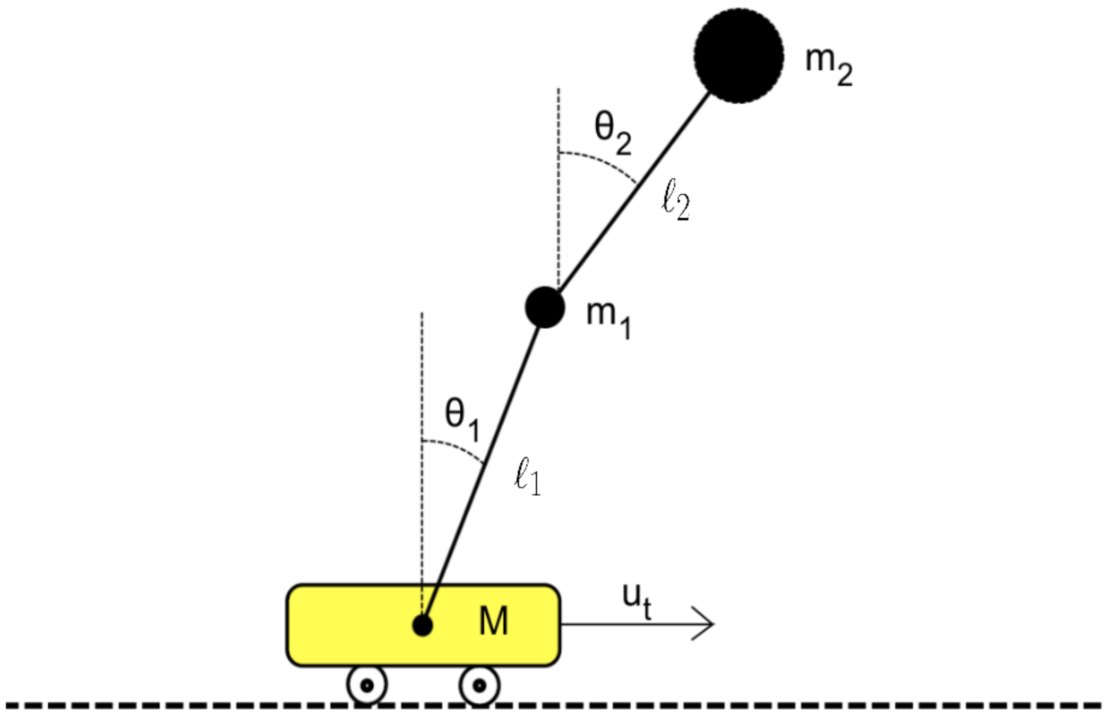
CHALMERS
UNIVERSITY OF TECHNOLOGY



Approximate stochastic control based on deep learning and forward backward stochastic differential equations

Master’s thesis in engineering mathematics and computational science

KRISTOFFER ANDERSSON

Department of mathematical sciences
CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden 2018

Master's thesis 2018:NN

Approximate stochastic control based on deep learning and forward backward stochastic differential equation

KRISTOFFER ANDERSSON

Department of Mathematical Sciences
Chalmers University of Technology
Gothenburg, Sweden 2018

Approximate stochastic control based on deep learning and forward backward stochastic differential equation

KRISTOFFER ANDERSSON

Supervisors: Dr. Adam Andersson, Syntronic and Assoc. prof. Mihály Kovács, Chalmers University of Technology and University of Gothenburg, Department of Mathematical Sciences
Examiner: Assoc. prof. Annika Lang, Chalmers University of Technology, Department of Mathematical Sciences

Master's Thesis 2018:NN
Department of Mathematical Sciences

Chalmers University of Technology
SE-412 96 Gothenburg
Telephone +46 31 772 1000

Cover: A double inverted pendulum on a cart.

Typeset in LaTeX
Gothenburg, Sweden 2018

Approximate stochastic control based on deep learning and forward backward stochastic differential equation

KRISTOFFER ANDERSSON
Department of Mathematical Sciences
Chalmers University of Technology

Abstract

Abstract: In this thesis numerical methods for stochastic optimal control are investigated. More precisely a nonlinear Gaussian diffusion type state equation with control in the drift and a quadratic cost functional with finite time horizon is considered. The proposed algorithm relies on recent work on deep learning based solvers for backward stochastic differential equations. The stochastic optimal control problem is reformulated as a forward backward stochastic differential equation and the algorithm is modified to apply to the problem of this thesis. The algorithm is tested on the benchmark problems of controlling single and double inverted pendulums on a cart. It is shown by numerical experiments that the algorithm performs well on both examples.

Keywords: Stochastic optimal control, neural networks, HJB, BSDE, FBSDE, inverted pendulum.

Acknowledgements

I would like to thank my two supervisors Adam Andersson and Mihàly Kovács. Thank you Adam for being extremely encouraging and helpful. Thank you for always pointing me in the right direction and never getting frustrated. Thank you for being available also in the evenings and weekends. Thank you Mihàly for making me interested in PDEs through your enthusiastic way of teaching the project course in PDEs. Thank you for introducing me to this project and for constantly following up my work and answering my questions. You have both been better supervisors than any student can expect. I would also like to thank Annika Lang and Arnulf Jentzen for helpful comments on the report.

Finally, I would like to thank Syntronic for making this project possible. I would also like to thank everyone at the office for your encouragement and especially Gustaf for your support in my progress.

Kristoffer Andersson, Gothenburg, October 2018

Contents

List of Figures

List of Tables

1 Introduction

Partial differential equations (PDEs) are among the most important mathematical tools for modelling the world around us. PDEs can be used to describe physical phenomena such as heat transfer, propagation of electromagnetic waves and the dynamics of fluids to name a few subjects. While some PDEs have known solution formulas, the vast majority has to be approximated numerically. Since the entry of computers the field of numerical methods for PDEs has been revolutionized. The most widely used methods are the finite element method (FEM), the finite difference method (FDM), the finite volume method (FVM) and spectral methods. Major progress have been made in adapting these methods to specific problems. Although the above methods are successful, they cannot be used to approximate PDEs in high dimensions. The difficulties lies in the "*curse of dimensionality*" [1], which in this context means that the computational cost grows exponentially with the dimensionality of the problem. In [2] the authors propose a deep learning-based algorithm for approximating a large class of high-dimensional second order semilinear parabolic PDEs and demonstrate the performance of the algorithm on various PDEs with 100 space dimensions. The algorithm uses the reformulation of parabolic PDEs as backward stochastic differential equations (BSDEs), which are then approximated by a deep neural network. The algorithm is called the deep BSDE solver and the main features are, roughly, the following:

(1) Choose one fixed spatial point x_0.
(2) Train the neural network to output an Euler–Maruyama type approximation to a certain BSDE, related to the PDE problem, being parametrized by x_0.
(3) After training of the neural network the initial value of the BSDE is an approximation of the solution f to the PDE at space point x_0 at time 0, *i.e.*, one training of the network approximates $f(0, x_0)$.

Experiments are performed on equations with known solutions, which are then compared to the approximate solutions. In this cases the algorithm performs well, both regarding to accuracy and computational cost. In [4], which is a follow-up work on [2] and [3], the authors propose an extended algorithm to solve fully nonlinear parabolic PDEs by solving the corresponding second order BSDE (2BSDE). In [2], [3] and [4] a temporal discretization in $N + 1$ time points is performed. The algorithms then rely on N subnetworks, which means that the number of parameters to optimize increases with N. This may be prohibitive when the temporal domain is large or when a fine time grid is required for the Euler–Maruyama scheme to be sufficiently accurate. In [5] a neural network based algorithm, similar to the BSDE solver, for solving high dimensional semilinear PDEs is proposed. The algorithm uses a reformulation of the PDE as a forward backward stochastic differential equation (FBSDE), but contrary to the BSDE solver, the number of parameters is independent of, N, the number of discrete time points. The authors also point out a straight forward extension to the algorithm, so that also fully nonlinear parabolic PDEs can be solved. To demonstrate the strength of deep learning algorithms, we include a reference to an article where an algorithm for solving stationary PDEs on complex domains is proposed. While different versions of the FEM, FDM and FVM are limited by the difficulties in meshing a complex spatial domain, which can be as difficult as the numerical solution of the PDE itself, the authors show that the algorithm proposed in [6] is able to solve low dimensional PDEs on complex spatial domains with high accuracy.

In stochastic optimal control theory, the optimal (feedback) control is given by the gradient of the solution to a so-called Hamilton–Jacobi–Bellman (HJB) equation, which is a semi-linear parabolic PDE. By construction, the BSDE solver not only computes the solution, but also the gradient of the solution to the HJB equation, which makes it tractable for applications in stochastic optimal control. One problem is that the BSDE solver only provides the optimal control at one point in space-time, while in applications, we are typically interested in a control strategy over a time interval and a spatial domain. To generalize the BSDE solver to be able to approximate a PDE over a spatial domain is feasible, which the authors point out in [3]. The procedure for doing this is to replacing (1)–(3) by:

(1') Choose a $\mathcal{D} \subset \mathbb{R}^n$.
(2') Train the network as in (2) but with the initial value to the BSDE not fixed but being a function (neural network) of $x_0 \in \mathcal{D}$.
(3') Obtain as in (3), an approximation $f(0, x_0)$ but now for any $x_0 \in \mathcal{D}$.

This is demonstrated in Chapter 6 for a 1 dimensional problem. The generalization to obtain the solution of the PDE at later time points is challenging for different reasons. Since the solution to the PDE is not of central interest in stochastic control but rather the gradient of the solution we leave this discussion out. The interested reader is referred to [5], where this is discussed.

In this thesis we propose an algorithm that approximates FBSDEs. As the name suggests a FBSDE is a coupled system of one forward and one backward stochastic differential equation (SDE). The BSDE part, again consists of a stochastic process with two components that correspond to the solution and the gradient of the solution to a PDE. The forward SDE plays the role of the state equation in the system that we want to control. The forward SDE and BSDE interact in the way that the BSDE provides the instantaneous optimal control signal at a space-time point given by the forward SDE, which is then instantaneously affected by the control. The algorithm is an obvious generalization of (1')–(3') with the BSDE replaced by a FBSDE.

In this thesis we apply the algorithm to a number of test problems, including control of single and double inverted pendulums on a moving cart. The control of pendulums on a cart to an upright position are classical benchmark problems in optimal control due to the nonlinear dynamics, see, *e.g.*, [22].

The algorithms rely on neural networks, which in the context of this thesis is a method for approximating functions. It should be pointed out that, although neural networks have been proven to be successful in many applications the theoretical foundation is in many respects underdeveloped.

The thesis is structured as follows: Chapter 2 deals with some basic theory from stochastic analysis. We start by introducing SDEs and BSDEs separately. The two concepts are then linked together to construct a FBSDE. We further give some existence and uniqueness results for SDEs, BSDEs and FBSDEs but it should be stressed that these results do not apply to the particular examples used to test the algorithms. In Chapter 3 we introduce the stochastic optimal control problem and give a sketch of a derivation of the HJB equation. Existence and uniqueness results for the HJB equation as well as a verification theorem that guarantees that the HJB equation actually provides the optimal control, are given. In the end of the chapter the connection between the stochastic optimal control problem and FBSDEs is explained. Chapter 4 contains a brief introduction to deep neural networks and in the final section all algorithms used in this thesis are introduced. Chapter 5 first deals with the dynamics of the single and double inverted pendulum on a cart. We then discuss numerical methods for approximating the dynamics and provide error plots of these approximations. In Chapter 6 the numerical results are presented. We start with two one-dimensional toy problems where one of them is a problem where an analytic optimal control can be found. The analytic control is compared to the approximate counterpart. We then move on to numerical experiments of control of the single and double inverted pendulum. We further suggest some strategies to improve the robustness of the control and show numerical examples of such experiments. Finally, in Chapter 7 we discuss our results and suggest possible future work.

2
Forward backward stochastic differential equations

As described in the introduction, FBSDEs play a central role in this thesis. The field of BSDEs and FBSDEs (which, again, is a coupled system of one SDE and one BSDE) is quite new. The concept of BSDEs was first introduced 1978 as a dual problem to a stochastic optimal control problem in [7] but it was not until 1990 that a proof of the existence of an adapted solution to a class of nonlinear BSDEs was presented in [8]. In 1992 the same authors discovered that the adapted solution of a BSDE could be viewed as a probabilistic interpretation of the solutions to some parabolic PDEs. Another important article [10] about applications of BSDEs and FBSDEs in mathematical finance was published in 1999. Since then a number of books [11], [12] and [13], to name a few, on the subject have been published.

Although the field of BSDE and FBSDE has been extensively explored since 1990 there are still important gaps in the theory, in particular when it concerns solutions to fully coupled FBSDE. In the upcoming sections we refer to theoretical results regarding existence and uniqueness of solutions when there are such results known but we also note that the equations used in later sections do not always satisfy the rather strict assumptions of the established theory. In Section 2.2 we give a brief introduction to SDEs and in Section 2.3 we introduce BSDEs with some more detailed exposition including results regarding existence and uniqueness of solutions to these equations. Proofs are included since they reveal, for the interested reader, what BSDEs actually are and how to understand them. Forward and backward SDEs are then linked together and a general form of FBSDEs are presented in Section 2.4. Finally, the two kinds of FBSDEs used in this thesis, uncoupled and coupled FBDSEs, are introduced. A solution to a FBSDE is considered to be uncoupled if the solution of the BSDE does not enter the dynamics of the SDE and coupled otherwise.

2.1 General setting

From now on we assume that $W = \{W_t\}_{t\in[0,\,T]}$ is a d-dimensional standard Brownian motion on a filtered probability space $(\Omega,\, \mathcal{F},\, \mathbf{F},\, \mathbb{P})$, where $\mathbf{F} = \{\mathcal{F}_t\}_{t\in[0,\,T]}$ is the filtration generated by W and completed by the $\mathbb{P}$-null sets of Ω. Throughout this thesis we assume that $[0,\, T]$ refers to a finite time interval, *i.e.*, $T < \infty$. We further assume that all stochastic processes in this thesis are adapted to $\mathbf{F}$ unless something else is specified.

The norm denoted by $|\cdot|$ should, from now on be interpreted as the Frobenius norm defined by $|A| = \sqrt{\operatorname{Trace}(A^T A)}$ for $A \in \mathbb{R}^{m\times n}$ for positive integers m and n. The trace of a matrix $B \in \mathbb{R}^{n\times n}$ is defined to be the sum of the elements on the diagonal, *i.e.*, $\operatorname{Trace}(B) = \sum_{i=1}^{n} b_{ii}$. Note that the Frobenius norm reduces to the Euclidean norm for vectors and to the absolute value for scalars. When referring to a mean-square process we mean a process $X = \{X_t\}_{t\in[0,\,T]}$ such that $\sup_{t\in[0,\,T]} \mathbb{E}\,[\,|X_t|^2] < \infty$ and for $t \in [0,\,T]$ it holds that $\lim_{s\to t} \mathbb{E}\,[\,|X_s - X_t|^2] = 0$. In this thesis we use the following spaces:

- $\mathcal{L}^p(\mathbb{R}^n)$ denotes the space of measurable functions $f\colon [0,\,T] \to \mathbb{R}^n$ satisfying $\int_0^T |f(t)|^p \mathrm{d}t < \infty$,
- $\mathbb{L}^2_T(\mathbb{R})$ denotes the space of all $\mathcal{F}_T$-measurable random variables $\xi\colon \Omega \to \mathbb{R}$ satisfying $\|\xi\|^2_{\mathbb{L}^2_T(\mathbb{R})} = \mathbb{E}\,[\,|\xi|^2] < \infty$,
- $\mathbb{H}^1_T(\mathbb{R}^d)$ denotes the space of all mean-square continuous, predictable processes $S\colon [0,\,T]\times\Omega \to \mathbb{R}^d$ satisfying $\|S\|_{\mathbb{H}^1_T(\mathbb{R}^d)} = \mathbb{E}\left[\sqrt{\int_0^T |S_t|^2\,\mathrm{d}t}\right] < \infty$,
- $\mathbb{H}^2_T(\mathbb{R}^d)$ denotes the space of all mean-square continuous, predictable processes $Z\colon [0,\,T]\times\Omega \to \mathbb{R}^d$ satisfying $\|Z\|^2_{\mathbb{H}^2_T(\mathbb{R}^d)} = \mathbb{E}\left[\int_0^T |Z_t|^2\,\mathrm{d}t\right] < \infty$,
- $\mathbb{S}^2_T(\mathbb{R})$ denotes the space of all mean-square continuous, predictable processes $Y\colon [0,\,T]\times\Omega \to \mathbb{R}$ satisfying $\|Y\|^2_{\mathbb{S}^2_T(\mathbb{R})} = \sup_{t\in[0,\,T]} \mathbb{E}\,[\,Y_t^2] < \infty$,
- $\mathbb{V}^2_T(\mathbb{R}\times\mathbb{R}^d)$ denotes the product normed vector space $\mathbb{S}^2_T(\mathbb{R}) \oplus \mathbb{H}^2_T(\mathbb{R}^d)$ of all pairs $(Y,\,Z) \in \mathbb{S}^2_T(\mathbb{R}) \oplus \mathbb{H}^2_T(\mathbb{R}^d)$ with norm $\|(Y,\,Z)\|^2_{\mathbb{V}^2_T(\mathbb{R}\times\mathbb{R}^d)} = \sup_{t\in[0,\,T]} \mathbb{E}\,[\,Y_t^2] + \mathbb{E}\left[\int_0^T |Z_t|^2\right]\,\mathrm{d}t$.

We remark that the bold face spaces all consist of equivalence classes of random variables or processes.

2.2 Forward stochastic differential equations

A forward stochastic differential equations (forward SDE or simply SDE) is an equation formally written

$$\mathrm{d}X_t = \mu(t,\,X_t)\,\mathrm{d}t + \sigma(t,\,X_t)\,\mathrm{d}W_t,\ t \in (0,\,T]; \quad X_0 = x_0, \tag{2.1}$$

where $\mu\colon [0,\,T]\times\mathbb{R}^n \to \mathbb{R}^n$ and $\sigma\colon [0,\,T]\times\mathbb{R}^n \to \mathbb{R}^{n\times d}$ are measurable functions referred to as the drift coefficient and the diffusion coefficient, respectively. Equation (2.1) is interpreted as the stochastic integral equation

$$X_t = x_0 + \int_0^t \mu(s,\,X_s)\,\mathrm{d}s + \int_0^t \sigma(s,\,X_s)\,\mathrm{d}W_s. \tag{2.2}$$

A natural question to ask is whether a solution to (2.1) exists and if the solution is unique. It should be mentioned that there exist different kinds of solution concepts to SDEs but for the purpose of this thesis we are only interested in so-called strong solutions. For a further discussion we refer to *e.g.* [15]. To give the most basic existence and uniqueness result for SDEs we need to impose some conditions on the drift and diffusion coefficients. The coefficients are said to be ***uniformly Lipschitz continuous*** in the second variable if there exists a constant $L_{\mu,\sigma} > 0$ such that for $t \in [0, T]$ and $x, y \in \mathbb{R}^n$ it holds

$$|\mu(t, x) - \mu(t, y)| + |\sigma(t, x) - \sigma(t, y)| \leq L_{\mu,\sigma}|x - y|.$$

The coefficients are said to satisfy a ***linear growth condition*** in the second variable if there exists a constant $C_{\mu,\sigma} > 0$ such that for $t \in [0, T]$ and $x, \in \mathbb{R}^n$ it holds

$$|\mu(t, x)| + |\sigma(t, x)| \leq C_{\mu,\sigma}(1 + |x|).$$

If the coefficients of (2.1) are uniformly Lipschitz continuous, satisfy a linear growth condition and the initial condition x_0 has bounded second order moment, *i.e.*

$$\mathbb{E}\left[\,|x_0|^2\right] < \infty,$$

then there exists a unique stochastic process X, with continuous sample paths, satisfying (2.1) for all $t \in [0, T]$ almost surely. Furthermore,

$$\mathbb{E}\left[\int_0^T |X_t|^2 \,\mathrm{d}t\right] < \infty.$$

Such a process is called a strong solution of (2.1). The assumptions on the coefficients above are rather restrictive and there are several results for existence and uniqueness in more general settings, for example in [16] and [17] the authors present results were the assumption on uniform Lipschitz continuity is relaxed.

2.3 Backward stochastic differential equations

For an ordinary differential equation (ODE) one can define both an initial value problem and a terminal value problem and under certain regularity assumptions they are both well-posed. In fact the terminal value problem is equivalent to the initial value problem for ODEs under the transformation $t \mapsto T - t$, where t is the time variable and T is the terminal time. However the task is much more delicate when it comes to SDEs since the time transformation would break the adaptivity of the solution of the SDE. In this section we strive for an intuitive motivation of how to formulate a BSDE with an adapted solution. The BSDEs considered in this thesis are of the form

$$-\,\mathrm{d}Y_t = F(t, Y_t, Z_t)\,\mathrm{d}t - Z_t\,\mathrm{d}W_t,\ t \in [0, T); \quad Y_T = \xi, \tag{2.3}$$

where $F\colon \Omega \times [0, T] \times \mathbb{R} \times \mathbb{R}^d \to \mathbb{R}$ is measurable. The integral form of (2.3) reads

$$Y_t = \xi + \int_t^T F(s, Y_s, Z_s)\,\mathrm{d}s - \int_t^T Z_s\,\mathrm{d}W_s,\ t \in [0, T]. \tag{2.4}$$

The BSDE (2.3) is characterized by the ***generator*** F and the terminal condition ξ. Therefore, the pair (F, ξ) is referred to as the parameters of the BSDE. The parameters are said to be ***standard parameters*** if:

- The terminal value $\xi \in \mathbb{L}^2_T(\mathbb{R})$,
- The stochastic process $F(\,\cdot\,, 0, 0) \in \mathbb{H}^2_T(\mathbb{R})$,
- The function F is uniformly Lipschitz continuous in the second and third variables *i.e.*, there exists a constant $L_F > 0$ such that for $t \in [0, T]$ and for $y_1, y_2 \in \mathbb{R}$ and $z_1, z_2 \in \mathbb{R}^d$, it holds almost surely that

$$|F(t, y_1, z_1) - F(t, y_2, z_2)| < L_F \left(|y_1 - y_2| + |z_1 - z_2|\right). \tag{2.5}$$

2.3.1 BSDEs with zero generator

As an illustrative example consider the equation

$$\mathrm{d}Y_t = 0, \; t \in [0, T); \quad Y_T = \xi, \tag{2.6}$$

where $\xi \in \mathbb{L}^2_T(\mathbb{R})$. It is clear that the only solution to (2.6) is $Y = \{Y_t\}_{t\in[0,T]}$ defined by $Y_t = \xi$ for all $t \in [0, T]$ which is not measurable with respect to the σ-algebra $\mathbf{F}$ for $t \geq 0$ unless ξ is $\mathcal{F}_0$-measurable, *i.e.*, in the case of an ODE with random initial value. We see that Y satisfies (2.6) but is not adapted to the filtration $\mathcal{F}_t$, for $t \in [0, T]$, in general. To fix this we instead consider Y defined by $Y_t = \mathbb{E}\left[\xi \,|\, \mathcal{F}_t\right]$, for $t \in [0, T]$. This process is $\mathbf{F}$-adapted and $Y_T = \xi$ but does not satisfy $\mathrm{d}Y_t = 0$. To find an Itô equation for Y with zero drift we first need to recall the martingale representation theorem.

Theorem 1 (The martingale representation theorem) *Suppose that* $\{M_t\}_{t\in[0,T]}$ *is a square integrable martingale with respect to the filtration* $\mathbf{F}$*. Then there exists a unique, predictable, square integrable, d-dimensional stochastic process* $Z = \{Z_t\}_{t\in[0,T]} \in \mathbb{H}^2_T(\mathbb{R}^d)$ *such that for all* $t \in [0, T]$ *it holds almost surely that*

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

We note that $Y_t = \mathbb{E}[\xi \,|\, \mathcal{F}_t]$, for $t \in [0, T]$ is a square integrable martingale with respect to the filtration $\mathbf{F}$ and can therefore conclude that there exists a unique process Z with the properties specified in Theorem 1 and this implies that Y satisfies

$$Y_t = Y_0 + \int_0^t Z_s \,\mathrm{d}W_s, \quad t \in [0, T]. \tag{2.7}$$

From the definition of Y we have that $Y_T = \xi$ and therefore

$$Y_0 = \xi - \int_0^T Z_s \,\mathrm{d}W_s. \tag{2.8}$$

Combining (2.7) and (2.8) gives

$$Y_t = \xi - \int_t^T Z_s \,\mathrm{d}W_s. \tag{2.9}$$

Equation (2.9) in differential form then reads

$$\mathrm{d}Y_t = Z_t \,\mathrm{d}W_t, \; t \in [0, T); \quad Y_T = \xi. \tag{2.10}$$

This equation is a BSDE (with zero drift), while (2.6) is not. Equation (2.10) is a more important and appropriate equation than (2.6). The solution to (2.10) is the pair $(Y,\, Z)$ of $\mathbf{F}$-adapted processes satisfying $Y_t = \mathbb{E}[\xi \,|\, \mathcal{F}_t]$ and (2.7). The only difference between (2.6) and (2.10) is the term $Z_t \,\mathrm{d}W_t$. This extra term makes the solution to (2.10) adapted to the filtration $\mathbf{F}$ while the solution to (2.6) is not.

2.3.2 BSDEs with non zero generator

To obtain a BSDE of a more general form we consider the martingale $M = \{M_t\}_{t\in[0,\,T]}$ defined by

$$M_t = \mathbb{E}\left[\xi + \int_0^T F(s,\, y_s,\, z_s)\,\mathrm{d}s \,\middle|\, \mathcal{F}_t\right], \tag{2.11}$$

for $t \in [0,\, T]$ where $F\colon \Omega\times[0,\, T]\times\mathbb{R}\times\mathbb{R}^d \to \mathbb{R}$ and $y = \{y_t\}_{t\in[0,\,T]}$ and $z = \{z_t\}_{t\in[0,\,T]}$ form a pair $(y,\, z) \in \mathbb{V}_T^2(\mathbb{R}\times\mathbb{R}^d)$. We further assume that $(F,\, \xi)$ are standard parameters. The reason for introducing the processes y, and z will become clear later in this section. To be able to use Theorem 1 we need to show that M is square integrable, which is equivalent to showing that $\xi + \int_0^T F(s,\, y_s,\, z_s)\,\mathrm{d}s \in \mathbb{L}_T^2(\mathbb{R})$. Since $\mathbb{L}_T^2(\mathbb{R})$ is a vector space which is closed under addition and $\xi \in \mathbb{L}_T^2(\mathbb{R})$ by definition it is enough to show that $\int_0^T F(s,\, y_s,\, z_s)\,\mathrm{d}s \in \mathbb{L}_T^2(\mathbb{R})$. To show this we use Hölder's inequality in the first step and in the second step we add and subtract $F(\cdot,\, 0,\, 0)$ and use the inequality $(a+b)^2 \le 2(a^2+b^2)$ to obtain

$$\begin{aligned}\mathbb{E}\left[\left(\int_0^T F(s,\, y_s,\, z_s)\,\mathrm{d}s\right)^2\right] &\le T\,\mathbb{E}\left[\int_0^T F(s,\, y_s,\, z_s)^2\,\mathrm{d}s\right]\\ &\le 2T\,\mathbb{E}\left[\int_0^T (F(s,\, y_s,\, z_s) - F(s,\, 0,\, 0))^2\,\mathrm{d}s\right] + 2T\,\mathbb{E}\left[\int_0^T F(s,\, 0,\, 0)^2\,\mathrm{d}s\right].\end{aligned} \tag{2.12}$$

The first term is bounded since we have assumed Lipschitz continuity of F in the second and third variable and since $y \in \mathbb{H}_T^2(\mathbb{R}^d)$ and $z \in \mathbb{S}_T^2(\mathbb{R})$. The second term is bounded since $F(\,\cdot\,,\, 0,\, 0) \in \mathbb{H}_T^2(\mathbb{R}^d)$. This implies that the process M is a square integrable martingale with respect to the filtration $\mathbf{F}$. We can then use Theorem 1 to guarantee the existence of a unique, predictable, square integrable stochastic process $Z = \{Z_t\}_{t\in[0,\,T]}$ such that M_t can be written as

$$M_t = M_0 + \int_0^t Z_s\,\mathrm{d}W_s,\ t \in [0,\, T]. \tag{2.13}$$

Since Z is predictable it is also $\mathbf{F}$-adapted and we can conclude that $Z \in \mathbb{H}_T^2(\mathbb{R}^d)$. To obtain the BSDE define $Y = \{Y_t\}_{t\in[0,\,T]}$ as the stochastic process

$$Y_t = \mathbb{E}\left[\xi + \int_t^T F(s,\, y_s,\, z_s)\,\mathrm{d}s \,\middle|\, \mathcal{F}_t\right] = M_t - \int_0^t F(s,\, y_s,\, z_s)\,\mathrm{d}s,\ t \in [0,\, T]. \tag{2.14}$$

By inserting (2.13) in the right hand side of (2.14) and noticing that $Y_0 = M_0$ and $Y_T = \xi$ we obtain

$$Y_t = Y_0 - \int_0^t F(s,\, y_s,\, z_s)\,\mathrm{d}s + \int_0^t Z_s\,\mathrm{d}W_s,\ t \in [0,\, T], \tag{2.15}$$

or, equivalently,

$$Y_t = \xi + \int_t^T F(s,\, y_s,\, z_s)\,\mathrm{d}s - \int_t^T Z_s\,\mathrm{d}W_s,\; t \in [0,\, T]. \tag{2.16}$$

We denote the mapping $\mathcal{G}\colon (y,\, z) \to (Y,\, Z)$, defined implicitly by (2.13) and (2.16). The following theorem guarantees that, under some conditions there exists a fixed point of the mapping $\mathcal{G}$, *i.e.*, a unique pair $(Y^*,\, Z^*) \in V_T^2(\mathbb{R} \times \mathbb{R}^d)$ satisfying $\mathcal{G}(Y^*,\, Z^*) = (Y^*,\, Z^*)$. The pair $(Y^*,\, Z^*)$ is then the unique solution to (2.3). As mentioned in the beginning of this chapter, the existence of unique adapted solutions to a class of nonlinear BSDEs was first proven in [8]. In the proof the authors use a Picard iteration to define an approximating sequence of the solution.

Theorem 2 *Given standard parameters, there exists a unique pair* $(Y,\, Z) \in \mathbb{V}_T^2(\mathbb{R} \times \mathbb{R}^d)$ *satisfying* (2.3).

The proof of Theorem 2 uses a fixed point argument that is based on Banach's fixed point theorem. In addition we use Itô's lemma, the Burkholder–Davis–Gundy inequality, the Lebesgue theorem of differentiation and a Grönwall type inequality. The first four theorems are well known standard theorems and we state them without proofs. For proofs we refer to *e.g.* [34] for Banach's fixed point theorem, [15] for Itô's lemma, [29] for the Lebesgue theorem of differentiation and [28] for the Burkholder–Davis–Gundy inequality. The Grönwall type inequality is a modification of a special case of an inequality from [30], which is a collection of a wide range of different Grönwall type inequalities. We further use the Cauchy–Schwarz inequality, Hölder's inequality and the inequality $(a+b)^2 \le 2(a^2+b^2)$ for $a,\, b \in \mathbb{R}$. These theorems are considered sufficiently well-known to refer to without stating them explicitly.

Theorem 3 (Banach's fixed point theorem) *Let* $(\mathbb{X},\, \|\cdot\|_{\mathbb{X}})$ *be a Banach space and let the mapping* $\mathcal{G}\colon \mathbb{X} \to \mathbb{X}$ *be a contraction, i.e. there exists a constant* $K \in (0,\, 1)$ *such that for all* $x_1,\, x_2 \in \mathbb{X}$ *it holds that*

$$\|\mathcal{G}x_1 - \mathcal{G}x_2\|_{\mathbb{X}} \le K\|x_1 - x_2\|_{\mathbb{X}}. \tag{2.17}$$

Then, there exists a unique fixed point $x \in \mathbb{X}$ *of the mapping* $\mathcal{G}$*, i.e. a unique* $x \in \mathbb{X}$ *such that* $\mathcal{G}x = x$.

Theorem 4 (Itô's lemma) *Let the process* $X = \{X_t\}_{t\in[0,\,T]}$ *defined by*

$$\mathrm{d}X_t = \mu(t,\, X_t)\mathrm{d}t + \sigma(t,\, X_t)\mathrm{d}W_t$$

be an n*-dimensional Itô process. Let the mapping* $g\colon [0,\, T] \times \mathbb{R}^n \to \mathbb{R}$ *be one time continuously differentiable in the first variable and two times continuously differentiable in the second variable. Then the process* $Y = \{Y_t\}_{t\in[0,\,\infty)}$ *defined by*

$$Y_t = g(t,\, X_t)$$

is again an itô process. Further, for $t \in [0,\, \infty)$*, it holds that*

$$\begin{aligned}\mathrm{d}Y_t = &\left[\frac{\partial g}{\partial t}(t,\, X_t) + \frac{1}{2}\left\{\sigma\sigma^T(t,\, X_t)\mathrm{Hess}_x g(t,\, X_t)\right\} + \nabla_x g(t,\, X_t)\mu(t,\, X_t)\right]\mathrm{d}t\\ &+ \nabla_x g(t,\, X_t)\sigma(t,\, X_t)\mathrm{d}W_t.\end{aligned}$$

Theorem 5 (The Burkholder–Davis–Gundy inequality) *For $m \in \left\{\frac{1}{2}, 1\right\}$ let $\phi \in \mathbb{H}_T^{2m}(\mathbb{R}^d)$. Then there exist universal constants k_m, $K_m < \infty$, such that for $T \geq 0$ it holds that*

$$k_m \mathbb{E}\left[\int_0^T |\phi_t|^2 \, dt\right]^m \leq \mathbb{E}\left|\sup_{t\in[0,\,T]} \int_0^t \phi_s \, dW_s\right|^{2m} \leq K_m \mathbb{E}\left[\int_0^T |\phi_t|^2 \, dt\right]^m .$$

Further, if $\phi \in \mathbb{H}_T^1(\mathbb{R}^d)$, then $\mathbb{E}\left[\int_0^T \phi_t dW_t\right] = 0$.

Theorem 6 (The Lebesgue theorem of differentiation) *Let $f\colon [a,\, b] \to \mathbb{R}$ be measurable such that f satisfies $\int_a^b |f(x)|dx < \infty$ and let $F(x) = \int_a^x f(s)ds$ for $x \in (a,\, b]$. Then for almost all $x \in (a,\, b)$ it holds that*

$$\frac{d}{dx}F(x) = \lim_{\epsilon\to 0} \frac{1}{2\epsilon} \int_{t-\epsilon}^{t+\epsilon} f(y)\, dy = f(x). \tag{2.18}$$

Lemma 1 *Let u, α, $\beta\colon [0,\, T] \to \mathbb{R}^+$ where u, $\alpha \in \mathcal{L}^1(\mathbb{R})$ and $\beta \in \mathcal{L}^2(\mathbb{R})$. Further assume that for $t \in [0,\, T]$ the inequality*

$$u_t + \alpha_t \leq 2C \int_t^T \sqrt{u_s}\beta_s \, ds \tag{2.19}$$

holds. Then for $t \in [0,\, T]$ it holds that

$$u_t \leq C^2 \left(\int_t^T \beta_s ds\right)^2 . \tag{2.20}$$

Proof of Lemma 1:
If $\int_0^T \sqrt{u_t}\beta_t \mathrm{d}t = 0$ we must have that $u_t = 0$ for all $t \in [0,\, T]$ and (2.20) holds. Assume that $\int_0^T \sqrt{u_t}\beta_t \mathrm{d}t \neq 0$. Note that since $\alpha_t \geq 0$ for all $t \in [0,\, T]$, it holds that $\sqrt{u_t} \leq \sqrt{2C\int_t^T \sqrt{u_s}\beta_s \, \mathrm{d}s}$. Then it follows from the Lebesgue theorem of differentiation that $\beta_t = \lim_{\epsilon\to 0} \frac{1}{2\epsilon}\int_{t-\epsilon}^{t+\epsilon} \beta_y \, \mathrm{d}y$ for almost every $t \in (0,\, T)$. As a consequence we have for almost every $t \in (0,\, T)$ that

$$\frac{\mathrm{d}}{\mathrm{d}t}\left(\int_t^T \sqrt{u_s}\beta_s \, \mathrm{d}s\right) = -\sqrt{u_t}\beta_t \geq -\beta_t\sqrt{2C\int_t^T \sqrt{u_s}\beta_s \, \mathrm{d}s}.$$

One can use the mean value theorem to show that the Lebesgue differentiation theorem implies that the chain rule holds. Therefore, by using the chain rule we obtain

$$\frac{\mathrm{d}}{\mathrm{d}t}\left(\sqrt{\int_t^T \sqrt{u_s}\beta_s \, \mathrm{d}s}\right) = \frac{\frac{\mathrm{d}}{\mathrm{d}t}\left(\int_t^T \sqrt{u_s}\beta_s \, \mathrm{d}s\right)}{2\sqrt{\int_t^T \sqrt{u_s}\beta_s \, \mathrm{d}s}} \geq -\sqrt{\frac{C}{2}}\beta_t.$$

Integrating over $[t,\, T]$ and squaring both sides implies that

$$\int_t^T \sqrt{u_s}\beta_s \, \mathrm{d}s \leq \frac{C}{2}\left(\int_t^T \beta_s \, \mathrm{d}s\right)^2$$

which is then inserted to (2.19) to obtain

$$u_t + \alpha_t \leq C^2 \left(\int_t^T \beta_s \, \mathrm{d}s \right)^2 .$$

□

We are now ready to prove Theorem 2.

Proof of Theorem 2:
Consider Y, Z, y, z as defined in the beginning of this section. We note that $Y \in \mathbb{S}^2_T(\mathbb{R})$ which can be shown by similar arguments as in (2.12) and that $Z \in \mathbb{H}^2_T(\mathbb{R}^d)$. Therefore $\mathcal{G}(y, z) = (Y, Z)$ is a mapping from the Banach space $\mathbb{V}^2_T(\mathbb{R} \times \mathbb{R}^d)$ to itself, *i.e.* $\mathcal{G}\colon \mathbb{V}^2_T(\mathbb{R} \times \mathbb{R}^d) \to \mathbb{V}^2_T(\mathbb{R} \times \mathbb{R}^d)$. Let $(y, z), (\bar{y}, \bar{z}) \in \mathbb{V}^2_T(\mathbb{R} \times \mathbb{R}^d)$ and let $\bar{Y}$ and $\bar{Z}$ be defined by $(\bar{Y}, \bar{Z}) = \mathcal{G}(\bar{y}, \bar{z})$. If we can show that $\mathcal{G}$ is a contraction, then Banach's fixed point theorem guarantees a unique fixed point for the mapping $\mathcal{G}\colon \mathbb{V}^2_T(\mathbb{R} \times \mathbb{R}^d) \to \mathbb{V}^2_T(\mathbb{R} \times \mathbb{R}^d)$ which would be the solution to the BSDE (2.3). We start by showing that there exists a unique solution to (2.3) on the time interval $[t, T]$, for some $t \in [0, T)$. To do this we introduce the Banach space $\left(\mathbb{V}^2_{t,T}(\mathbb{R} \times \mathbb{R}^d), \|\cdot\|_{\mathbb{V}^2_{t,T}(\mathbb{R}\times\mathbb{R}^d)}\right)$ where the norm is defined by $\|(Y, Z)\|^2_{\mathbb{V}^2_{t,T}(\mathbb{R}\times\mathbb{R}^d)} = \|Y\|^2_{\mathbb{S}^2_{t,T}(\mathbb{R})} + \|Z\|^2_{\mathbb{H}^2_{t,T}(\mathbb{R}^d)} = \sup_{\tau\in[t,T]} \mathbb{E}\left[Y_\tau^2\right] + \mathbb{E}\left[\int_t^T |Z_s|^2 \, \mathrm{d}s\right]$. Note that $\mathcal{G}$ maps $\mathbb{V}^2_{t,T}(\mathbb{R}\times\mathbb{R}^d)$ to itself. The next step is to show that the mapping $\mathcal{G}\colon \mathbb{V}^2_{t,T}(\mathbb{R} \times \mathbb{R}^d) \to \mathbb{V}^2_{t,T}(\mathbb{R} \times \mathbb{R}^d)$ is a contraction for some appropriate t. By Itô's lemma we have that

$$\begin{aligned}\mathrm{d}(Y_t - \bar{Y}_t)^2 =& 2(Y_t - \bar{Y}_t)(F(t, y_t, z_t) - F(t, \bar{y}_t, \bar{z}_t))\mathrm{d}t \\ &- 2(Y_t - \bar{Y}_t)(Z_t - \bar{Z}_t)\mathrm{d}W_t + |Z_t - \bar{Z}_t|^2\mathrm{d}t.\end{aligned}$$

Integrating over $[t, T]$ and using the fact that $Y_T = \bar{Y}_T$ we obtain

$$\begin{aligned}(Y_t - \bar{Y}_t)^2 =& -2\int_t^T (Y_s - \bar{Y}_s)(F(s, y_s, z_s) - F(s, \bar{y}_s, \bar{z}_s))\mathrm{d}s \\ &+ 2\int_t^T (Y_s - \bar{Y}_s)(Z_s - \bar{Z}_s)\mathrm{d}W_s - \int_t^T |Z_s - \bar{Z}_s|^2\mathrm{d}s.\end{aligned}$$

Taking the expectation of both sides and moving the last term to the right hand side gives

$$\begin{aligned}\mathbb{E}\left[(Y_t - \bar{Y}_t)^2\right] + \mathbb{E}\left[\int_t^T |Z_s - \bar{Z}_s|^2\mathrm{d}s\right] =& -2\mathbb{E}\left[\int_t^T (Y_s - \bar{Y}_s)(F(s, y_s, z_s) - F(s, \bar{y}_s, \bar{z}_s))\mathrm{d}s\right] \\ &+ 2\mathbb{E}\left[\int_t^T (Y_s - \bar{Y}_s)(Z_s - \bar{Z}_s)\,\mathrm{d}W_s\right].\end{aligned} \tag{2.21}$$

We now want to show that the expectation of the stochastic integral is zero. To do this we use the last statement in the Burkholder–Davis–Gundy inequality guaranteeing that $\mathbb{E}\left[\int_t^T (Y_s - \bar{Y}_s)(Z_s - \bar{Z}_s)\,\mathrm{d}W_s\right] = 0$ if $(Y - \bar{Y})(Z - \bar{Z}) \in \mathbb{H}^1_T(\mathbb{R}^d)$, or equivalently if $\mathbb{E}\left[\sqrt{\int_0^T |(Y-\bar{Y})(Z-\bar{Z})|^2\,\mathrm{d}t}\right] < \infty$. By Hölder's inequality we have that

$$\int_t^T (Y_s-\bar{Y}_s)^2|Z_s-\bar{Z}_s|^2\,\mathrm{d}s \leq \sup_{\tau\in[t,\,T]}(Y_\tau-\bar{Y}_\tau)^2\int_t^T |Z_s-\bar{Z}_s|^2\,\mathrm{d}s.$$

Raising both sides to $1/2$ and taking the expectation yields

$$\begin{aligned}\mathbb{E}\left[\left(\int_t^T (Y_s-\bar{Y}_s)^2|Z_s-\bar{Z}_s|^2\,\mathrm{d}s\right)^{1/2}\right] &\leq \mathbb{E}\left[\left(\sup_{\tau\in[t,\,T]}(Y_\tau-\bar{Y}_\tau)^2\int_t^T |Z_s-\bar{Z}_s|^2\,\mathrm{d}s\right)^{1/2}\right]\\ &\leq \left(\mathbb{E}\left[\sup_{\tau\in[t,\,T]}(Y_\tau-\bar{Y}_\tau)^2\right]\right)^{1/2}\left(\mathbb{E}\left[\int_t^T |Z_s-\bar{Z}_s|^2\,\mathrm{d}s\right]\right)^{1/2},\end{aligned}$$

where the last inequality is obtained by using the Cauchy–Schwarz Inequality. Since $Z-\bar{Z}\in\mathbb{H}_T^2(\mathbb{R}^d)$ it holds that $\mathbb{E}\left[\int_t^T |Z_s-\bar{Z}_s|^2\,\mathrm{d}s\right] \leq \|Z-\bar{Z}\|^2_{\mathbb{H}_T^2(\mathbb{R}^d)} < \infty$. Recall the representation of Y from (2.16), which in combination with the inequality $(a+b)^2 \leq 2(a^2+b^2)$ yields

$$\begin{aligned}\mathbb{E}\left[\sup_{\tau\in[t,\,T]}(Y_\tau-\bar{Y}_\tau)^2\right] \leq& 2\mathbb{E}\left[\left(\int_t^T |F(s,\,y_s,\,z_s)-F(s,\,\bar{y}_s,\,\bar{z}_s)|\,\mathrm{d}s\right)^2\right]\\ &+2\mathbb{E}\left[\sup_{\tau\in[t,\,T]}\left|\int_\tau^T (Z_s-\bar{Z}_s)\,\mathrm{d}W_s\right|^2\right].\end{aligned}$$

The first term on the right is finite by the Lipschitz continuity (2.5) of F. For the second term the inequality $(a+b)^2 \leq 2(a^2+b^2)$ is used again to obtain

$$\begin{aligned}&\mathbb{E}\left[\sup_{\tau\in[t,\,T]}\left|\int_\tau^T (Z_s-\bar{Z}_s)\,\mathrm{d}W_s\right|^2\right]\\ &\qquad\leq 2\mathbb{E}\left[\left|\int_t^T (Z_s-\bar{Z}_s)\,\mathrm{d}W_s\right|^2\right]+2\mathbb{E}\left[\sup_{\tau\in[t,\,T]}\left|\int_t^\tau (Z_s-\bar{Z}_s)\,\mathrm{d}W_s\right|^2\right]\\ &\qquad\leq 4\mathbb{E}\left[\sup_{\tau\in[t,\,T]}\left|\int_t^\tau (Z_s-\bar{Z}_s)\,\mathrm{d}W_s\right|^2\right].\end{aligned}$$

Since $Z-\bar{Z}\in\mathbb{H}_T^2(\mathbb{R}^d)$ we can again use the Burkholder–Davis–Gundy inequality with $m=1$ to conclude that there exists a constant $K_1<\infty$ such that

$$\mathbb{E}\left[\sup_{\tau\in[t,\,T]}\left|\int_\tau^T (Z_s-\bar{Z}_s)\,\mathrm{d}W_s\right|^2\right] \leq K_1\,\mathbb{E}\left[\int_t^T |Z_s-\bar{Z}_s|^2\,\mathrm{d}s\right] \leq \infty.$$

We therefore conclude that $(Y-\bar{Y})(Z-\bar{Z})\in\mathbb{H}_T^1(\mathbb{R}^d)$ which implies that (2.21) becomes

$$\begin{aligned}&\mathbb{E}\Big[(Y_t-\bar{Y}_t)^2\Big]+\mathbb{E}\Big[\int_t^T |Z_s-\bar{Z}_s|^2\mathrm{d}s\Big]\\ &\qquad\qquad= -2\mathbb{E}\left[\int_t^T (Y_s-\bar{Y}_s)(F(s,\,y_s,\,z_s)-F(s,\,\bar{y}_s,\,\bar{z}_s))\mathrm{d}s\right].\end{aligned}$$

To be able to move the expectation inside the integral, we take the absolute value of the integrand on the right hand side and use Tonelli's theorem and obtain

$$\mathbb{E}\Big[(Y_t-\bar{Y}_t)^2\Big]+\mathbb{E}\Big[\int_t^T |Z_s-\bar{Z}_s|^2\mathrm{d}s\Big] \\ \leq 2\int_t^T \mathbb{E}\left[\Big|(Y_s-\bar{Y}_s)(F(s,\,y_s,\,z_s)-F(s,\,\bar{y}_s,\,\bar{z}_s))\Big|\,\mathrm{d}s\right].$$

By Hölder's inequality and the Lipschitz continuity (2.5) of F it holds that

$$\mathbb{E}\left[(Y_t-\bar{Y}_t)^2\right]+\mathbb{E}\left[\int_t^T |Z_s-\bar{Z}_s|^2\mathrm{d}s\right] \\ \leq 2L_F\int_t^T \left(\mathbb{E}\left[(Y_s-\bar{Y}_s)^2\right]\right)^{1/2}\left\{\left(\mathbb{E}\left[(y_s-\bar{y}_s)^2\right]\right)^{1/2}+\left(\mathbb{E}\left[|z_s-\bar{z}_s|^2\right]\right)^{1/2}\right\}\mathrm{d}s. \tag{2.22}$$

Our aim is to apply Lemma 1 and in this way obtain a bound where the right hand side does not depend on Y and $\bar{Y}$. To do this we need to take some care with the function $s\mapsto(\mathbb{E}\left[|z_s-\bar{z}_s|^2\right])^{1/2}$. Since $z,\bar{z}\in\mathbb{H}_T^2(\mathbb{R}^d)$ it holds that $z,\bar{z}$ are not pointwise defined (changing z and $\bar{z}$ on a $\mathbb{P}\times[0,T]$ nullset gives the same vectors z, $\bar{z}\in\mathbb{H}_T^2(\mathbb{R}^d)$, *i.e.*, they belong to the same equivalence class). For this reason $s\mapsto(\mathbb{E}\left[|z_s-\bar{z}_s|^2\right])^{1/2}$ is not pointwise defined. On the other hand we can pick a version (an element from the equivalence class) ζ of $s\mapsto(\mathbb{E}\left[|z_s-\bar{z}_s|^2\right])^{1/2}$ which is pointwise defined. Let $u_t=\mathbb{E}\left[(Y_t-\bar{Y}_t)^2\right]$, $\alpha_t=\mathbb{E}\left[\int_t^T|Z_s-\bar{Z}_s|\,\mathrm{d}s\right]$ and $\beta_t=(\mathbb{E}\left[(y_t-\bar{y}_t)^2\right])^{1/2}+\zeta$. By Lemma 1 it now holds that

$$\mathbb{E}\left[(Y_t-\bar{Y}_t)^2\right]+\mathbb{E}\left[\int_t^T |Z_s-\bar{Z}_s|^2\,\mathrm{d}s\right] \\ \leq L_F^2\left(\int_t^T\left\{\left(\mathbb{E}\left[(y_s-\bar{y}_s)^2\right]\right)^{1/2}+\left(\mathbb{E}\left[|z_s-\bar{z}_s|^2\right]\right)^{1/2}\right\}\mathrm{d}s\right)^2. \tag{2.23}$$

By again using the inequality $(a+b)^2\leq 2(a^2+b^2)$ on the right hand side of (2.23) we obtain

$$\mathbb{E}\Big[(Y_t-\bar{Y}_t)^2\Big]+\mathbb{E}\left[\int_t^T |Z_s-\bar{Z}_s|^2\,\mathrm{d}s\right] \\ \leq 2L_F^2\left(\left(\int_t^T\left(\mathbb{E}\left[(y_s-\bar{y}_s)^2\right]\right)^{1/2}\mathrm{d}s\right)^2+\left(\int_t^T\left(\mathbb{E}\left[|z_s-\bar{z}_s|^2\right]\right)^{1/2}\mathrm{d}s\right)^2\right).$$

By the Cauchy–Schwartz inequality the above can be written as

$$\mathbb{E}\left[(Y_t-\bar{Y}_t)^2\right]+\mathbb{E}\left[\int_t^T |Z_s-\bar{Z}_s|^2\,\mathrm{d}s\right] \\ \leq 2L_F^2\left(\int_t^T\mathbb{E}\left[(y_s-\bar{y}_s)^2\right]\mathrm{d}s+\int_t^T\mathbb{E}\left[|z_s-\bar{z}_s|^2\right]\mathrm{d}s\right).$$

To obtain the appropriate norm on the right hand side note that

$$\int_t^T \mathbb{E}\left[(y_s - \bar{y}_s)^2\right] \mathrm{d}s \leq (T-t) \sup_{\tau \in [t,\, T]} \mathbb{E}\left[(y_\tau - \bar{y}_\tau)^2\right],$$

and

$$\int_t^T \mathbb{E}\left[|z_s - \bar{z}_s|^2\right] \mathrm{d}s = \mathbb{E}\left[\int_t^T |z_s - \bar{z}_s|^2 \,\mathrm{d}s\right],$$

which gives

$$\begin{aligned}\mathbb{E}\left[(Y_t - \bar{Y}_t)^2\right] + \mathbb{E}\bigg[\int_t^T |Z_s - \bar{Z}_s|^2 \,\mathrm{d}s\bigg] \\ \leq 2L_F^2 \max\left\{T-t,\, (T-t)^2\right\} \|(y,\, z) - (\bar{y},\, \bar{z})\|^2_{\mathbb{V}^2_{t,\,T}(\mathbb{R}\times\mathbb{R}^{\mathrm{d}})}.\end{aligned}$$

For the left hand side note that

$$\begin{aligned}&\sup_{\tau\in[t,\,T]} \left(\mathbb{E}\left[(Y_\tau - \bar{Y}_\tau)^2\right] + \mathbb{E}\left[\int_\tau^T |Z_s - \bar{Z}_s|^2 \,\mathrm{d}s\right]\right) \\ &\qquad \geq \frac{1}{2}\left(\sup_{\tau\in[t,\,T]} \mathbb{E}\left[(Y_\tau - \bar{Y}_\tau)^2\right] + \mathbb{E}\left[\int_t^T |Z_s - \bar{Z}_s|^2, \,\mathrm{d}s\right]\right).\end{aligned}$$

We therefore conclude that

$$\|(Y,\, Z) - (\bar{Y},\, \bar{Z})\|_{\mathbb{V}^2_{t,\,T}(\mathbb{R}\times\mathbb{R}^{\mathrm{d}})} \leq K \|(y,\, z) - (\bar{y},\, \bar{z})\|_{\mathbb{V}^2_{t,\,T}(\mathbb{R}\times\mathbb{R}^{\mathrm{d}})}, \tag{2.24}$$

where $K = 4L_F\sqrt{\max\{T-t,\, (T-t)^2\}}$. Reading from the expression of K, choosing $h^* = T - t^*$ small enough we can make $K < 1$ to obtain a contraction. Without loss of generality we can choose $h^* = \frac{1}{\sqrt{2L_F + \epsilon}}$, for any fixed $\epsilon > 0$. In terms of t^* this can be expressed as $t^* = T - \frac{1}{\sqrt{2L_F+\epsilon}}$. By this choice of t^* we let $K_{h^*} = 2L_F(h^*)^2$ which satisfy $K_{h^*} \in (0,\, 1)$ and in turn that (2.24) is a contraction. By Banach's fixed point theorem the existence of a unique solution to (2.3) on the time interval $[t^*,\, T]$ is guaranteed. To extend the result to the interval $[0,\, T]$ we note that by the same arguments that led to (2.16) it holds

$$Y_t = Y_{T-h^*} + \int_t^{T-h^*} F(s,\, y_s,\, z_s)\mathrm{d}s + \int_t^{T-h^*} Z_s \mathrm{d}W_s,\ t \in [0,\, T-h^*]. \tag{2.25}$$

We can then do the same calculations as above to conclude that there exists a unique solution to (2.3) on the interval $[t^* - 2h^*,\, t^* - h^*]$. Repeating this procedure concludes the proof. □

By Theorem 2 we have concluded that the BSDE (2.3) is well-posed. With this in mind we recall that $Y_0 = \mathbb{E}\left[\xi + \int_0^T F(s,\, Y_s,\, Z_s)\,\mathrm{d}s\right]$ which makes it possible to write Y as the forward SDE

$$Y_t = \mathbb{E}\left[\xi + \int_0^T F(s,\, Y_s,\, Z_s)\,\mathrm{d}s\right] - \int_0^t F(s,\, Y_s,\, Z_s)\,\mathrm{d}s + \int_0^t Z_s\,\mathrm{d}W_s,\ t\in[0,\, T], \tag{2.26}$$

or in differential form

$$\mathrm{d}Y_t = -F(t,\, Y_t,\, Z_t)\,\mathrm{d}t + Z_t\,\mathrm{d}W_t,\ t \in (0,\, T]; \quad Y_0 = \mathbb{E}\left[\xi + \int_0^T F(s,\, Y_s,\, Z_s)\,\mathrm{d}s\right].$$

Note that Y actually has a deterministic initial value. For this reason the BSDE (2.3) can be viewed as a two point boundary value problem.

2.4 Forward backward stochastic differential equations

Later in this thesis a forward SDE is used to describe the dynamics of a physical system, *e.g.*, an inverted pendulum on a cart. The drift term of the SDE comes from the Newtonian laws of motion and a control process $u = \{u_t\}_{t\in[0,\,T]}$. The control process will be a deterministic function of time and the solution of a BSDE. A coupled system of one forward SDE and one BSDE is then obtained. This system is called a FBSDE and in this section we formulate a general form of FBSDEs and define what we mean by a "solution" to a FBSDE. The FBSDEs considered in this thesis are of the form

$$\begin{cases} \mathrm{d}X_t = \mu\Big(t,\, X_t,\, Y_t,\, Z_t\Big)\,\mathrm{d}t + \sigma(t,\, X_t,\, Y_t,\, Z_t)\,\mathrm{d}W_t,\ t\in(0,\,T]; \quad X_0 = x_0 \\ -\mathrm{d}Y_t = F\Big(t,\, X_t,\, Y_t,\, Z_t\Big)\,\mathrm{d}t - Z_t\,\mathrm{d}W_t,\ \in[0,\,T); \quad Y_T = g(X_T), \end{cases} \tag{2.27}$$

where $\mu\colon [0,\,T]\times\mathbb{R}^n\times\mathbb{R}\times\mathbb{R}^d\to\mathbb{R}^n$, $\sigma\colon [0,\,T]\times\mathbb{R}^n\times\mathbb{R}\times\mathbb{R}^d\to\mathbb{R}^{n\times d}$ and $F\colon [0,\,T]\times\mathbb{R}^n\times\mathbb{R}\times\mathbb{R}^d\to\mathbb{R}$ are measurable functions. The processes $X=\{X_t\}_{t\in[0,\,T]}$, $Y=\{Y_t\}_{t\in[0,\,T]}$ and $Z=\{Z_t\}_{t\in[0,\,T]}$ together form the triple $(X,\,Y,\,Z)\in\mathbb{S}^2_T(\mathbb{R}^n)\times\mathbb{S}^2_T(\mathbb{R})\times\mathbb{H}^2_T(\mathbb{R}^d)$ which is called an adapted solution of the FBSDE (2.27) if it satisfies (2.27) almost surely. The FBSDE (2.27) in integral form reads as

$$\begin{aligned} X_t =& x_0 + \int_0^t \mu(s,\, X_s,\, Y_s,\, Z_s)\,\mathrm{d}s + \int_0^t \sigma(s,\, X_s,\, Y_s,\, Z_s)\,\mathrm{d}W_s,\ t\in[0,\,T] \\ Y_t =& g(X_T) + \int_t^T F(s,\, X_s,\, Y_s,\, Z_s)\,\mathrm{d}s - \int_t^T Z_s\,\mathrm{d}W_s,\ t\in[0,\,T]. \end{aligned} \tag{2.28}$$

In the following theorem from [11] the author proves existence and uniqueness of a solution to (2.27) under some rather strict assumptions.

Theorem 7 *Let the following assumptions hold:*

(a) Uniform Lipschitz continuity in $(x,\,y,\,z)$ for $\mu,\,\sigma,\,F$ and g, i.e. for $\rho=\mu,\,\sigma,\,F$ there exists a constant $L_\rho>0$ such that for $t\in[0,\,T]$ and for $x_1,\,x_2\in\mathbb{R}^n$, $y_1,\,y_2\in\mathbb{R}$ and $z_1,\,z_2\in\mathbb{R}^d$ it holds that

$$|\rho(t,\,x_1,\,y_1,\,z_1)-\rho(t,\,x_2,\,y_2,\,z_2)|\le L_\rho\left(|x_1-x_2|+|y_1-y_2|+|z_1-z_2|\right),$$
$$|g(x_1)-g(x_2)|\le L_g|x_1-x_2|,$$

(b) The functions $g(0)\in\mathbb{L}^2_T(\mathbb{R})$, $F(\,\cdot\,,\,0,\,0,\,0)\in\mathbb{H}^2_T(\mathbb{R})$, $\mu(\,\cdot\,,\,0,\,0,\,0)\in\mathbb{H}^2_T(\mathbb{R}^n)$ and $\sigma(\,\cdot\,,\,0,\,0,\,0)\in\mathbb{L}^2_T(\mathbb{R}^{n\times d})$,

(c) The Lipschitz constants for σ and g satisfies $L_\sigma L_g<1$.

Then there exists $\delta_0>0$, only denpending on the Lipschitz constants such that for $T\le\delta_0$ there exists a unique solution $(X,\,Y,\,Z)$ to the FBSDE (2.27).

2.4.1 Decoupled FBSDEs

In this section we are concerned with FBSDEs that are decoupled (or uncoupled) in the sense that the solution of the BSDE does not enter the dynamics of the SDE.

The backward equation in a FBSDE of this type can be viewed as a BSDE where the randomness of the parameters $(F,\, \xi)$ comes from a forward SDE of the form (2.1). The FBSDEs with uncoupled forward SDEs considered in this thesis are of the form

$$\begin{cases} \mathrm{d}X_t = \mu\Big(t,\, X_t\Big)\,\mathrm{d}t + \sigma(t,\, X_t)\,\mathrm{d}W_t,\ t \in (0,\, T]; \quad X_0 = x_0, \\ -\mathrm{d}Y_t = F\Big(t,\, X_t,\, Z_t\Big)\,\mathrm{d}t - Z_t\,\mathrm{d}W_t,\ t \in [0,\, T); \quad Y_T = g(X_T), \end{cases} \tag{2.29}$$

where $\mu\colon [0,\, T] \times \mathbb{R}^n \to \mathbb{R}^n$, $\sigma\colon [0,\, T] \times \mathbb{R}^{\mathrm{n}} \to \mathbb{R}^{n\times d}$ and $F\colon [0,\, T] \times \mathbb{R}^n \times \mathbb{R}^d \to \mathbb{R}$ are measurable functions. The integral form of (2.29) reads as

$$X_t = x_0 + \int_0^t \mu(s,\, X_s)\,\mathrm{d}s + \int_0^t \sigma(s,\, X_s)\,\mathrm{d}W_s,\ t \in [0,\, T], \tag{2.30}$$

$$Y_t = g(X_T) + \int_t^T F(s,\, X_s,\, Z_s)\,\mathrm{d}s + \int_t^T Z_s\,\mathrm{d}W_s,\ t \in [0,\, T]. \tag{2.31}$$

The forward SDE is an equation of the form treated in Section 2.1. Existence and uniqueness of a process X satisfying (2.30) is therefore guaranteed if μ, σ and x fullfills the conditions stated in Section 2.1. The BSDE is more complicated since it depends on the solution of the SDE. Given initial condition of the SDE $X_0 = x_0$ (2.31) can be viewed as a parametrized BSDE, where $(0,\, x_0)$ is data. With this in mind it is not surprising that the conditions for existence and uniqueness of a solution to (2.31) are similar to the conditions for existence of a unique solution to a BSDE stated in Theorem 2. The only thing we need to have in mind is that the generator also depends on X and that the terminal condition of the BSDE depends on X_T. The following theorem is stated and proved in for example [14].

Theorem 8 *Suppose that F and g have polynomial growth in x, i.e., for $p \geq 1/2$, there exists a constant $C > 0$ such that for $t \in [0,\, T]$ and $(y,\, z) \in \mathbb{R} \times \mathbb{R}^d$ it holds that*

$$|F(t,\, x,\, y,\, z)| + |g(x)| \leq C(1 + |x|^p).$$

In addition we assume that $X = \{X_t\}_{t\in[0,T]}$ satisfies (2.1), μ and σ satisfies the conditions for existence of a unique solution of (2.1) stated in Section 2.1 and F is uniformly Lipschitz continuous in the second and third variables. Then there exists a unique solution $(X,\, Y,\, Z)$ to (2.29).

3
Stochastic optimal control

Optimal control theory deals with the problem of optimizing actions to attain some predetermined goal. An easy real life example is the mixing of hot and cold water (under constant stirring) for achieving the desired temperature for a bath. A natural way to describe the state of the system is by the temperature as a function of time. Often in literature, this function is referred to as the state equation of the system. If we, for simplicity, assume a constant inflow of water, the feedback control could be the proportion of hot (or cold) water added to the system as a function of the current sensed temperature. Note that the state equation depends both on the physical laws of heat transfer and the change in temperature obtained by the control signal. In reality, many systems are disturbed by random noise. If noise is taken account for in the state equation we enter the field of ***stochastic optimal control***.

In Section 3.1 we introduce necessary notation and state some general assumptions. Section 3.2 aims to explain the two main strategies for finding the feedback control used in this thesis. Section 3.3 and 3.4 deals with the so-called Hamilton–Jacobi–Bellman equation. In Section 3.5 and 3.6 the stochastic optimal control problem is reformulated as a FBSDE and the chapter ends with an illustrative example were the abstract methodology becomes concrete.

This chapter relies heavily on [19], to which we constantly refer. Another good reference for stochastic control is [20].

3.1 General setting

In this thesis we want to control stochastic processes described by a forward SDE introduced in Chapter 2. To be able to do this we add dependency of a control process u in the drift coefficient of the SDE describing the process that we want to control. It is also possible to consider dependency of the control process in the diffusion coefficient but we consider a diffusion coefficient without dependency of the control:

$$\mathrm{d}X_t = \mu(t,\, X_t,\, u_t)\,\mathrm{d}t + \sigma(t,\, X_t)\,\mathrm{d}W_t,\ t \in (0,\, T];\quad X_0 = x_0. \tag{3.1}$$

For now, we only assume that the drift and diffusion coefficients $\mu\colon [0,\, T] \times \mathbb{R}^n \times \mathbb{U} \to \mathbb{R}^n$ and $\sigma\colon [0,\, T] \times \mathbb{R}^n \to \mathbb{R}^{n\times d}$ are continuous. The space $\mathbb{U}$ is a closed subspace of $\mathbb{R}^d$ and is called the control space. In the context of this thesis (3.1) describes the physical states of a dynamical system and is therefore referred to as the state equation. The process $u\colon [0,T] \times \Omega \to \mathbb{U}$ is referred to as the control process. Throughout this thesis we assume that u_t is a deterministic function π depending only on t and X_t, *i.e.*, $u_t = \pi(t,\, X_t)$ for all $t \in [0,\, T]$. If this is the case $\pi\colon [0,T] \times \mathbb{R}^n \to \mathbb{U}$ is said to be a ***Markov control policy***. In many applications, including those considered in this thesis a Markov control policy can be expressed as an explicit mapping. If $X = \{X_t\}_{t\in[0,\,T]}$ is a solution to (3.1) and satisfies the ***Markov property***, *i.e.*, for $A \in \mathbb{R}^n$ and $0 \le s < t \le T$

$$\mathbb{P}\left(X_t \in A \,|\, \mathcal{F}_s\right) = \mathbb{P}\left(X_t \in A \,|\, X_s\right), \tag{3.2}$$

we then say that X is a ***controlled Markov diffusion process***. Throughout this thesis we assume that the forward SDE in the stochastic control problem is a controlled Markov diffusion process. To be able to give a meaning to 'optimal control' we first need to introduce a way to measure the performance of the control. We therefore define the so called ***cost functional***

$$J(t,\, x\,;\, u) = \mathbb{E}\left[\int_t^T L(s,\, X_s,\, u_s)\,\mathrm{d}s + g(X_T) \;\Big|\; X_t = x\right], \tag{3.3}$$

where $L\colon [0,\, T] \times \mathbb{R}^n \times \mathbb{U} \to \mathbb{R}^+$ is called the ***running cost*** and $g\colon \mathbb{R}^n \to \mathbb{R}^+$ is called the ***terminal cost***. As the name suggests the cost functional should be viewed as the cost for a control process u at time t starting in state x and the lower the value of (3.3), the better the control. Note that the cost functional always takes on non-negative values. Throughout this thesis we make the following assumptions regarding (3.1) and (3.3).

Assumption 1 (State equation)

- *The coefficients $\mu(\,\cdot\,,\,\cdot\,,\, v)$ and $\sigma(\,\cdot\,,\,\cdot\,)$ are continuously differentiable in t and x for all $v \in \mathbb{U}$.*
- *There exists a constant $L_{\mu,\sigma} < \infty$ such that*

$$|\mu(s,\, y,\, v) - \mu(t,\, x,\, v)| + |\sigma(s,\, y) - \sigma(t,\, x)| \le L_{\mu,\sigma}\Big(|s-t| + |y-x|\Big),$$

  for all $v \in \mathbb{U}$ and $(s,\, y),\, (t,\, x) \in [0,\, T] \times \mathbb{R}^n$.
- *The diffusion coefficient $\sigma(t,\, x,\, v)$ has full rank for all $(t,\, x,\, v) \in [0,\, T] \times \mathbb{R}^n \times \mathbb{U}$.*

Assumption 2 (Cost functional)

- *The terminal cost g is convex and continuously differentiable in x.*
- *There exists a constant $C_{L,\,g} < \infty$ such that for all $u \in \mathbb{U}$ and $(t,\, x) \in [0,\, T] \times \mathbb{R}^n$ it holds that*

$$|L(t,\, x,\, u)| + |g(x)| \le C_{L,\,g}(1 + |u| + |x|^2).$$

Note that μ and σ are vector and matrix valued functions, respectively, and recall that $|\cdot|$ is the Frobenius norm defined in Section 2.1. We say that a control process

u is ***progressively measurable*** if, for every $s \in [t,\, T]$ the mapping $\Omega \times [t,\, T] \to \mathbb{U}$ defined by $(\omega,\, s) \to u_s(\omega)$ is $\mathcal{B}([t,\, s]) \otimes \mathcal{F}_t$-measurable, where $\mathcal{B}([t,\, s])$ is the Borel σ-algebra on $[t,\, s]$. If a progressively measurable control in addition satisfies

$$\mathbb{E}\left[\int_t^T |u_s|^m \,\mathrm{d}s\right] < \infty,$$

for all $m \in \mathbb{N}^+$ we call it an ***admissible control***. Note that an admissible control is $\mathbf{F}$-adapted, which makes intuitive sense. We denote the class of admissible controls by $\mathcal{A}_t$. For simplicity we assume from now on that $\mathbb{U}$ is a compact set. This guarantees that a progressively measurable control is admissible. Under the assumptions above there exists a unique strong solution $X = \{X_t\}_{t\in[0,\,T]}$ of (3.1) (see *e.g.* [19]) which is progressively measurable and has continuous sample paths. In addition, for $m \in \mathbb{R}^+$, $t \in [0,\, T]$ and $u \in \mathcal{A}_0$ it holds that

$$\mathbb{E}\left[|X_t|^m\right] < \infty.$$

Our ***stochastic optimal control problem*** is then: *Find an admissible control u such that the cost functional* (3.3) *is minimized.* We finally define the so-called ***value function*** by

$$V(t,\, x) = \inf_{u\in\mathcal{A}_t} J(t,\, x\,;u), \;\; (t,\, x) \in [0,\, T] \times \mathbb{R}^n. \tag{3.4}$$

The value function is the cost when the process is optimally controlled from the state $(t,\, x)$. It has a central importance in optimal control.

3.2 Conceptual overview of the strategies for finding the optimal Markov control policy

Before presenting details we give a brief overview of the methodology to solve the control problem. The purpose of this section is to give a hint of the aim of the (often long) derivations in the sections below and to show why they are important. In Section 3.3 we show that the value function is the solution to a semi-linear parabolic PDE. This surprising connection between a stochastic optimal control problem and a PDE is fundamental in this thesis. The PDE is referred to as the ***Hamilton–Jacobi–Bellman*** (HJB) equation. We further show that the optimal Markov control policy, denoted π^*, is a mapping of the form

$$\pi^*(t,\, x) = \nu(t,\, x,\, \nabla_x V(t,\, x)),$$

where $\nabla_x V(t,\, x)$ is the vector valued gradient of the value function V with respect to x. This means that, given the state $(t,\, x)$, it is enough to find the gradient of the solution to the HJB equation, evaluated at $(t,\, x)$, to obtain the optimal Markov control policy. The first strategy to find a Markov control policy used in this thesis is given by:

Control with dynamic programming equation (DPE):

(i) Find the expression for ν algebraically,
(ii) Compute $\nabla_x V$ analytically or numerically,
(iii) Immediate consequence: π^* is obtained.

The second strategy for finding a Markov control policy used in this thesis is based on the connection between parabolic PDEs and stochastic diffusion processes [26]. In Section 3.5 we derive a FBSDE of the type introduced in Section 2.4 with solution given by the triple $(X,\, Y,\, Z)$ satisfying the following:

$$\begin{aligned}&X \text{ is the state equation, given by equation (3.1)},\\ &Y = \{Y_t\}_{t\in[0,\,T]} \text{ is given by } Y_t = V(t,\, X_t),\\ &Z = \{Z_t\}_{t\in[0,\,T)} \text{ is given by } Z_t = \nabla_x V(t,\, X_t)\sigma(t,\, X_t).\end{aligned}$$

We further show that the optimal Markov control policy π^* is given by a mapping of the form

$$\pi^*(t,\, X_t) = \kappa(t,\, X_t,\, Z_t).$$

The mapping κ is determined by the structure of the specific problem and becomes clear in Section (3.5).

Control with FBSDE:
(i) Find the expression for κ algebraically,
(ii) Compute Z analytically or numerically (function of X),
(iii) Immediate consequence: π^* is obtained.

3.3 A sketch of derivation of the Hamilton–Jacobi–Bellman equation

In this section a sketch of a derivation of the HJB equation is presented. The intention of the section is to give an intuitive explanation of the connection between the stochastic optimal control problem and HJB equation. The reader is encouraged to look at this section in the same way as one looks at a deduction of a physicist where an equation is derived from some physical principles. For a rigorous derivation we refer to [21]. This section, on the other hand, relies heavily on the formal derivation of the HJB equation in [19].

The value function is central in the part of optimal control known as dynamic programming. Recall the value function (3.4), which by using the definition of the cost functional in (3.3), reads as

$$V(t,\, x) = \inf_{u\in\mathcal{A}_t} \mathbb{E}\left[\int_t^T L(s,\, X_s,\, u_s)\,\mathrm{d}s + g(X_T) \;\middle|\; X_t = x\right],\ (t,\, x) \in [0,\, T] \times \mathbb{R}^n. \tag{3.5}$$

Since X is a Markov diffusion process and u is determined by a Markov control policy it is reasonable to believe that each subinterval of an optimally controlled process is itself optimally controlled on each subinterval. That is to say that it is equivalent to minimize the cost function over the interval $[t,\, T]$ as it is to first minimize it over the interval $[t,\, t+h]$ and then over $[t+h,\, T]$. This is called ***Bellman's dynamic programming principle***, which reads as

$$V(t,\, x) = \inf_{u\in\mathcal{A}_t} \mathbb{E}\left[\int_t^{t+h} L(s,\, X_s,\, u_s)\,\mathrm{d}s + V(t+h,\, X_{t+h}) \,\middle|\, X_t = x\right]. \tag{3.6}$$

We sketch a derivation of the HJB equation under the assumption that Bellman's dynamic programming principle holds. Let u^* be the optimal control and u be the control given by $u_s = v$ for $s \in [t, t+h]$ and $u_s = u_s^*$ for $s \in (t+h, T]$. Since the control u is sub-optimal it holds that

$$V(t,\, x) \le \mathbb{E}\left[\int_t^{t+h} L(s,\, X_s,\, v)\,\mathrm{d}s \,\middle|\, X_t = x\right] + \mathbb{E}\left[V(t+h,\, X_{t+h}) \,\middle|\, X_t = x\right]. \tag{3.7}$$

By subtracting $V(t,\, x)$ from both sides, dividing by h and letting $h \to 0^+$ we obtain

$$\begin{aligned} 0 \le \lim_{h\to 0^+} \frac{1}{h}\, \mathbb{E}\left[\int_t^{t+h} L(s,\, X_s,\, v)\,\mathrm{d}s \,\middle|\, X_t = x\right] \\ + \lim_{h\to 0^+} \frac{\mathbb{E}\left[V(t+h,\, X_{t+h}) \,|\, X_t = x\right] - V(t,\, x)}{h}. \end{aligned} \tag{3.8}$$

Boldly assuming that we can use Tonelli's theorem and Lebesgue's theorem of differentiation, we have

$$\begin{aligned} \lim_{h\to 0^+} \frac{1}{h}\mathbb{E}\left[\int_t^{t+h} L(s,\, X_s,\, v)\,\mathrm{d}s \,\middle|\, X_t = x\right] &= \lim_{h\to 0^+} \frac{1}{h} \int_t^{t+h} \mathbb{E}\left[L(s,\, X_s,\, v) \,|\, X_t = x\right]\, \mathrm{d}s \\ &= L(t,\, X_t,\, v). \end{aligned} \tag{3.9}$$

To rewrite the second term on the right we use Itô's lemma to obtain

$$\begin{aligned} &V(t+h,\, X_{t+h}) - V(t,\, x) \\ &= \int_t^{t+h} \left[\frac{\partial V}{\partial t}(s,\, x) + \frac{1}{2}\mathrm{Trace}\left\{\sigma\sigma^T(s,\, x)\mathrm{Hess}_x V(s,\, x)\right\} + \nabla_x V(s,\, x)^T \mu(s,\, x,\, u_s)\right]\mathrm{d}s \\ &\qquad + \int_t^{t+h} \nabla_x V(s,\, X_s)\sigma(s,\, X_s)\,\mathrm{d}W_s. \end{aligned} \tag{3.10}$$

As before, Trace denotes the trace of a matrix and Hess_x and ∇_x denote the Hessian and the vector valued gradient with respect to x. By introducing the ***backward operator*** denoted $\mathcal{A}^v$ defined by

$$\begin{aligned} \mathcal{A}^v\Phi(t,\, x) =& \frac{\partial \Phi}{\partial t}(t,\, x) + \frac{1}{2}\mathrm{Trace}\left\{\sigma\sigma^T(t,\, x)\mathrm{Hess}_x\Phi(t,\, x)\right\} \\ &+ \nabla_x\Phi(t,\, x)^T\mu(t,\, x,\, v), \end{aligned} \tag{3.11}$$

we can write (3.10) as

$$V(t+h,\, X_{t+h}) - V(t,\, x) = \int_t^{t+h} \mathcal{A}^v V(s,\, X_s)\,\mathrm{d}s + \int_t^{t+h} \nabla_x V(s,\, X_s)\sigma(s,\, X_s)\,\mathrm{d}W_s. \tag{3.12}$$

The notation $\mathcal{A}^v$ aims to emphasize the dependency on v of the backward operator. By taking the conditional expectation of both sides of equation (3.12) we obtain

$$\begin{aligned}\mathbb{E}\left[V(t+h,\, X_{t+h})\,|\, X_t = x\right] - V(t,\, x) &= \mathbb{E}\left[\int_t^{t+h} \mathcal{A}^v V(s,\, X_s,\, v)\,\mathrm{d}s \,\middle|\, X_t = x\right] \\ &+ \mathbb{E}\left[\int_t^{t+h} \nabla_x V(s,\, X_s)\sigma(s,\, X_s)\,\mathrm{d}W_s \,\middle|\, X_t = x\right].\end{aligned} \tag{3.13}$$

We assume that the conditional expectation of the stochastic integral above is zero and that the $\mathcal{A}^v V$ is continuous for all $v \in \mathbb{U}$. Dividing equation (3.13) by h, letting $h \to 0^+$ and again using Tonelli's theorem to move the expectation inside the integral and Lebesgue's theorem of differentiation yields

$$\begin{aligned}&\lim_{h\to 0^+} \frac{\mathbb{E}\left[V(t+h,\, X_{t+h})\,|\, X_t = x\right] - V(t,\, x)}{h} \\ &= \lim_{h\to 0^+} \frac{1}{h}\int_t^{t+h} \mathbb{E}\left[\mathcal{A}^v V(s,\, X_s)\,|\, X_t = x\right]\mathrm{d}s = \mathcal{A}^v V(t,\, x).\end{aligned} \tag{3.14}$$

By inserting (3.9) and (3.14) to (3.8) it holds, for any $v \in \mathbb{U}$ that

$$0 \le \mathcal{A}^v V(t,\, x) + L(t,\, x,\, v). \tag{3.15}$$

On the other hand, assume that π^* is an optimal Markov control policy and $X^* = \{X_t^*\}_{t\in[0,\,T]}$ is the optimal dynamics controlled by π^*. Then from Bellman's dynamic programming principle we have

$$0 = \mathbb{E}\left[\int_t^{t+h} L(s,\, X_s^*,\, \pi^*(s,\, X_s^*))\,\mathrm{d}s \,\middle|\, X_t^* = x\right] + \mathbb{E}\left[V(t+h,\, X_{t+h}^*)\,\middle|\, X_t^* = x\right]. \tag{3.16}$$

By similar calculations as before and assuming that π^* is regular enough we obtain

$$0 = \mathcal{A}^{\pi^*} V(t,\, X_t^*) + L(t,\, X_t^*,\, \pi^*(t,\, X_t^*)). \tag{3.17}$$

By combining (3.15) and (3.17) we obtain the ***dynamic programming equation*** which reads

$$0 = \inf_{u\in\mathbb{U}} \left[\mathcal{A}^u V(t,\, x) + L(t,\, x,\, u)\right], \tag{3.18}$$

with the terminal condition $V(T, x) = g(x)$. A reasonable choice for the optimal control is then given by

$$\pi^*(t,\, x) = \arg\min_{u\in\mathbb{U}} \left[\mathcal{A}^u V(t,\, x) + L(t,\, x,\, u)\right], \tag{3.19}$$

$(t,\, x) \in [0,\, T) \times \mathbb{R}^n$. Here we assume that the argmin is nonempty and unique but the uniqueness is not necessary. In the next section we provide a theorem that, under some conditions on V guarantees that (3.19) holds. The theorem is therefore called a verification theorem. In this section, and for the sake of this formal derivation we assume that (3.19) holds. By combining (3.18) and (3.19) we obtain the HJB equation, which in our setting is a second order PDE given by

$$\frac{\partial \Psi}{\partial t}(t,\, x) + \frac{1}{2}\text{Trace}\left\{\sigma\sigma^T(t,\, x)\text{Hess}_x\Psi(t,\, x)\right\}$$
$$+\nabla_x\Psi(t,\, x)^T\mu(t,\, x,\, \pi^*(t,\, x)) + L(t,\, x,\, \pi^*(t,\, x)) = 0,\ (t,\, x) \in [0,\, T) \times \mathbb{R}^n. \tag{3.20}$$

Equation (3.20) should also satisfy a terminal condition extracted from the cost function given by

$$\Psi(T,\, x) = g(x), \quad x \in \mathbb{R}^n. \tag{3.21}$$

For $a = \sigma\sigma^T$, we say that the HJB equation is ***uniformly parabolic*** if there exists a constant $C < \infty$ such that it holds that

$$\sum_{i,\, j=1}^{n} a_{ij}\xi_i\xi_j \geq C|\xi|^2 \text{ for all } (t,\, x,\, u) \in [0,\, T] \times \mathbb{R}^n \times \mathbb{U}, \text{ and } \xi \in \mathbb{R}^n. \tag{3.22}$$

When (3.22) holds we can use standard PDE theory to state a theorem for a classical solution to (3.20)-(3.21). Otherwise the HJB equation is said to be degenerate and it is often necessary to consider so called viscosity solutions. In this thesis we restrict ourselves to the uniformly parabolic case. From the perspective of the state equation (3.22) holds when the state equation is perturbed in each spatial dimension for all states x and time t. This is fulfilled when $\sigma \in \mathbb{R}^{n\times d}$ has full rank and $d \geq n$.

3.4 Existence and uniqueness theorem for the HJB equation and a verification theorem

In Section 3.3 a formal derivation of the HJB equation was outlined under the assumption that the dynamic programming principle holds. In this section we state an existence and uniqueness theorem for the HJB equation. We also state a corollary to a so-called verification theorem that guarantees that the dynamic programming principle holds if the HJB equation has a classical solution. Both the existence and uniqueness theorem and the verification theorem are proved in, *e.g.* [19]. The corollary follows almost immediately from the verification theorem.

We start by introducing notation for continuous functions. We let $C^{\ell,k}([0,\, T] \times \mathbb{R}^n\,;\, \mathbb{V})$ be the space of all functions $[0, T] \times \mathbb{R}^n \to \mathbb{V}$, where $\mathbb{V}$ is a finite dimensional vector space, that are ℓ times continuously differentiable functions in t and $\frac{\partial^k\phi}{\partial^{k_1}\partial^{k_2}\ldots\partial^{k_n}}$ exist and are continuous for all $k_1, \ldots, k_n \in \{0, 1, \ldots, k\}$ satisfying $k_1 + k_2 + \cdots + k_n \in \{0, 1, \ldots, k\}$. We further let $C_b^{\ell,k}([0,\, T] \times \mathbb{R}^n\,;\, \mathbb{V})$ be the space of all functions $\phi \in C^{l,k}([0,\, T] \times \mathbb{R}^n\,;\, \mathbb{V})$ such that all the partial derivatives of ϕ are bounded. Similarly, we denote by $C^k(\mathbb{R}^n\,;\, \mathbb{V})$ and $C_b^k(\mathbb{R}^n\,;\, \mathbb{V})$, the corresponding spaces that do not depend on t.

Theorem 9 *Assume that the following assumptions holds:*

(a) *The matrix* $a = \sigma\sigma^T$ *satisfies* (3.22).
(b) *The set* $\mathbb{U}$ *is compact.*
(c) *It holds that* $\mu \in C_b^{1,2}([0,\, T] \times \mathbb{R}^n\,;\, \mathbb{R}^n)$, $\sigma \in C_b^{1,2}([0,\, T] \times \mathbb{R}^n\,;\, \mathbb{R}^{n\times d})$ *and* $a \in C_b^{1,2}([0,\, T] \times \mathbb{R}^n\,;\, \mathbb{R}^{n\times n})$.
(d) *The function* $g \in C_b^3(\mathbb{R}^n\,;\, \mathbb{R})$.

Then there exists a classical solution to (3.20)–(3.21), *i.e., there exists a unique* $\Psi \in C_b^{1,2}([0,\, T] \times \mathbb{R}^n)$ *satisfying* (3.20)–(3.21).

Theorem 10 (Verification theorem) *Let Assumption 1 and Assumption 2 hold and let* $\Psi \in C^{1,2}([0,\, T]\times\mathbb{R}^n)$ *be a solution to* (3.20)-(3.21) *and* $\pi^*\colon [0,\, T]\times\mathbb{R}^n \to \mathbb{U}$ *be a Markov control policy satisfying:*
(a) *The optimal Markov control policy* $\pi^*(t,\, x) \in \arg\min\left[\mathcal{A}^v\Psi(t,\, x) + L(t,\, x,\, v)\right]$.
(b) *The SDE* (3.1) *controlled with* π^* *has a unique strong solution.*
(c) *Dynkin's formula holds, i.e.*

$$\mathbb{E}\left[\Psi(T,\, X_T)\right] - \Psi(t,\, x) = \mathbb{E}\left[\int_t^T \mathcal{A}^{u_s}\Psi(s,\, X_s)\,ds\right].$$

Then the following holds:
(i) $\Psi(t,\, x) \le J(t,\, x\,;\, \pi)$, *for every Markov control policy,*
(ii) $\Psi(t,\, x) = J(t,\, x\,;\, \pi^*)$,
(iii) $\Psi(t,\, x) = V(t,\, x)$,
(iv) *The dynamic programming principle holds.*

Proof:
The proof of (*i*)–(*iii*) follows from Theorem 8.1 in [19] by fixing the probability space to $(\Omega,\, \mathcal{F},\, \mathbf{F},\, \mathbb{P})$ and observing that $\pi^*(t,\, x) \in \arg\min\left[\mathcal{A}^v\Psi(t,\, x) + L(t,\, x,\, v)\right]$. Statement (*iii*) holds by Theorem 7.1 in [19]. □

Note that in Section 3.3 we derived the HJB equation under the assumption that the dynamic programming principle holds. In Theorem 10 we see that the dynamic programming principle holds under the assumption that the HJB equation has a classical solution. One could therefore argue that the derivation in Section 3.3 is made in the "wrong" direction. The reason for the chosen order is to make it more pedagogical and easier to follow for the reader.

3.5 Reformulation of a stochastic optimal control problem to a FBSDE

In this section we assume a stochastic optimal control problem defined by the state equation (3.1) and the cost functional (3.3). We further assume that the statements in Theorems 9 and 10 hold. With these assumptions we have established a connection between the stochastic optimal control problem and the HJB equation (3.20)–(3.21). We further know that if we can find a Markov control policy π^* satisfying

$$\pi^*(t,\, x) = \underset{u\in\mathbb{U}}{\arg\min}\left[\mathcal{A}^u V(t,\, x) + L(t,\, x,\, u)\right], \tag{3.23}$$

for $(t, x) \in [0, T] \times \mathbb{R}^n$ then it holds that $V(t, x) = J(t, x, \pi^*(t, x))$, where V is the value function defined in (3.4). For simplicity we assume existence and uniqueness of such π^* in this section. The HJB equation in compact form reads as

$$\mathcal{A}^{\pi^*}V(t, x) + L(t, x, \pi^*(t, x)) = 0, \ t \in [0, T); \quad V(T, x) = g(x). \tag{3.24}$$

The aim of this section is to, given (3.23) and (3.24), derive an associated FBSDE.

By inspection of (3.23) and the definition of the backward operator (3.11) we conclude that π^* is a function of t, x and $\nabla_x V(t, x)$, *i.e.* π^* can be written as

$$\pi^*(t, x) = \nu(t, x, \nabla V_x(t, x)). \tag{3.25}$$

Recall that this mapping was the basis for the idea of controlling with DPE, which was introduced in the beginning of this chapter. To derive the optimal control from the solution component Z of a FBSDE, we note that by the full rank property of σ, it is possible to perform a change of variable from $\nabla_x V(t, x)$ to $\nabla_x V(t, x)\sigma(t, x)$. Therefore we can define $\tilde{L}$ equivalent to L by

$$\tilde{L}\Big(t, x, \nabla_x V(t, x)\sigma(t, x)\Big) = L(t, x, \pi^*(t, x)) \tag{3.26}$$

for all $(t, x) \in [0, T] \times \mathbb{R}^n$. By Theorem 9 it holds that $V \in C_b^{1,2}([0, T] \times \mathbb{R}^n)$ and therefore we can use Itô's lemma and (3.20) to obtain for $t \in [0, T]$ that

$$\begin{aligned} \mathrm{d}V(t, X_t) &= \mathcal{A}^{u^*}V(t, X_t)\mathrm{d}t + \nabla_x V(t, X_t)\sigma(t, X_t)\mathrm{d}W_t \\ &= -\tilde{L}\left(t, X_t, \nabla_x V(t, X_t)\sigma(t, X_t)\right)\mathrm{d}t + \nabla_x V(t, X_t)\sigma(t, X_t)\mathrm{d}W_t. \end{aligned} \tag{3.27}$$

The terminal condition $V(T, X_T) = g(X_T)$ is a direct consequence of (3.21). To make it easier to see that (3.27) actually is a BSDE we define the stochastic processes $Y = \{Y_t\}_{t\in[0,T]}$ and $Z = \{Z_t\}_{t\in[0,T]}$ given by

$$Y_t = V(t, X_t), \quad Z_t = \nabla_x V(t, X_t)\sigma(t, X_t).$$

Equation (3.27) written in terms of Y and Z then becomes

$$\mathrm{d}Y_t = -\tilde{L}\left(t, X_t, Z_t\right)\mathrm{d}t + Z_t\,\mathrm{d}W_t, \ t \in [0, T); \quad Y_T = g(X_T). \tag{3.28}$$

Equation (3.28) is the BSDE part of the FBSDE (which is the final form in the reformulation of the stochastic optimal control problem). The forward equation in the FBSDE is the state equation (3.1) expressed in terms of Z instead of u^*. This is done by introducing $\tilde{\mu}$, equivalent to the drift coefficient in the state equation, μ and for $t \in [0, T]$ defined by

$$\tilde{\mu}(t, X_t, Z_t) = \mu(t, X_t, \pi^*(t, X_t)).$$

The reformulation of μ is justified by the same arguments as the reformulation in (3.26). The state equation then reads as

$$\mathrm{d}X_t = \tilde{\mu}(t, X_t, Z_t)\mathrm{d}t + \sigma(t, X_t)\mathrm{d}W_t, \ t \in (0, T]; \quad X_0 = x_0. \tag{3.29}$$

Finally, by combining (3.28) and (3.29) we obtain the FBSDE

$$\begin{cases} \mathrm{d}X_t = \tilde{\mu}(t,\, X_t,\, Z_t)\mathrm{d}t + \sigma(t,\, X_t)\mathrm{d}W_t,\ t \in (0,\, T]; \quad X_0 = x_0, \\ \mathrm{d}Y_t = -\tilde{L}\,(t,\, X_t,\, Z_t)\,\mathrm{d}t + Z_t\mathrm{d}W_t,\ t \in [0,\, T); \quad Y_T = g(X_T). \end{cases} \tag{3.30}$$

The system above is a coupled FBSDE of the type introduced in Section 2.4.

To reconnect to Section 3.2 and the second strategy for finding the Markov control policy, we recall that

$$u_t^* = \pi^*(t,\, X_t) = \nu(t,\, X_t,\, \nabla_x V(t,\, X_t)). \tag{3.31}$$

Again, by the full rank property of σ it is possible to perform a change of variable from $\nabla_x V$ to $\nabla_x V\sigma$. We can therefore introduce κ, equivalent to ν, defined by

$$\kappa(t,\, X_t,\, Z_t) = \nu(t,\, X_t,\, \nabla_x V(t,\, X_t)). \tag{3.32}$$

By combining (3.31) and (3.32) we obtain an optimal Markov control policy given by

$$\pi^*(t,\, X_t) = \kappa(t,\, X_t,\, Z_t)$$

which is on the claimed form.

3.6 Change of drift coefficients

In Chapter 2 we introduced coupled and decoupled FBSDEs. By Theorems 7 and 8 one notices that the conditions for existence and uniqueness of a solution is much more restrictive for coupled than for decoupled FBSDEs. Since the theoretical results may or may not be sharp no conclusions can be drawn from this but it can be seen as an indication of the difficulties in solving a coupled FBSDE. In this section we state a proposition that makes it possible to change the drift coefficient in the forward SDE part of a FBSDE. As a consequence a "correction" term is added to the drift coefficient in the BSDE. In particular this allows us to reformulate a coupled FBSDE as a decoupled FBSDE. It is worth pointing out that "reformulate" in this context refers to the possibility of solving the same stochastic optimal control problem with a reformulated FBSDE, *i.e.*, the optimal Markov control policy, π^* does not change.

Proposition 1 *Let π^* be defined as in Theorem 10. Assume that $V \in C_b^{1,2}([0,\, T] \times \mathbb{R}^n)$ is the unique solution to the HJB equation*

$$\begin{aligned} \frac{\partial V}{\partial t}(t,\, x) + \operatorname{Trace}\left\{\sigma\sigma^T(t,\, x)\mathrm{Hess}_x V(t,\, x)\right\} + \nabla_x V(t,\, x)\mu\,(t,\, x,\, \nabla_x V(t,\, x)\sigma(t,\, x)) \\ + L\,(t,\, x,\, \nabla_x V(t,\, x)\sigma(t,\, x)) = 0,\ (t,\, x) \in [0,\, T) \times \mathbb{R}^n; \quad V(T,\, x) = g(x). \end{aligned} \tag{3.33}$$

Assume further that $\psi\colon [0,\, T] \times \mathbb{R}^n \times \mathbb{R}^d \to \mathbb{R}^n$ is uniformly Lipschitz continuous in the second and third variable, i.e., there exists a constant $L_\psi > 0$ such that for $t \in [0,\, T]$ and for $x_1,\, x_2 \in \mathbb{R}^n$ and $z_1,\, z_2 \in \mathbb{R}^d$ it holds that

$$|\psi(t,\, x_1,\, z_1) - \psi(t,\, x_2,\, z_2)| \leq L_\psi\left(|x_1 - x_2| + |z_1 - z_2|\right).$$

A family of associated FBSDEs is then given by

$$\begin{cases} \mathrm{d}X_t = \eta\left(t,\, X_t,\, \nabla_x V(t,\, X_t)\sigma(t,\, X_t)\right)\mathrm{d}t + \sigma(t,\, X_t)\mathrm{d}W_t,\; t \in (0,\, T]; \quad X_0 = x_0, \\ \mathrm{d}V(t,\, X_t) = -F\left(t,\, X_t,\, \nabla_x V(t,\, X_t)\sigma(t,\, X_t)\right)\mathrm{d}t + \nabla_x V(t,\, X_t)\sigma(t,\, X_t)\mathrm{d}W_t, \\ t \in [0,\, T); \quad V(T,\, X_T) = g(X_T), \end{cases} \tag{3.34}$$

where

$$\begin{aligned} &\eta\left(t,\, X_t,\, \nabla_x V(t,\, X_t)\sigma(t,\, X_t)\right) = \\ &\qquad \mu(t,\, X_t,\, \nabla_x V(t,\, X_t)\sigma(t,\, X_t)) + \psi\left(t,\, X_t,\, \nabla_x V(t,\, X_t)\sigma(t,\, X_t)\right), \\ &F\left(t,\, X_t,\, \nabla_x V(t,\, X_t)\sigma(t,\, X_t)\right) = \\ &\qquad L(t,\, X_t,\, \nabla_x V(t,\, X_t)\sigma(t,\, X_t)) - \psi\left(t,\, X_t,\, \nabla_x V(t,\, X_t)\sigma(t,\, X_t)\right). \end{aligned}$$

Proof:
Let

$$\begin{aligned} \mathrm{d}X_t = \Big[&\mu(t,\, X_t,\, \nabla_x V(t,\, X_t)\sigma(t,\, X_t)) - \psi\left(t,\, X_t,\, \nabla_x V(t,\, X_t)\sigma(t,\, X_t)\right)\Big]\mathrm{d}t \\ &+ \sigma(t,\, X_t)\mathrm{d}W_t,\; t \in (0,\, T]; \quad X_0 = x_0. \end{aligned}$$

Then by Itô's lemma, it holds for $t \in [0,\, T]$ that

$$\begin{aligned} \mathrm{d}V(t, X_t) = -\Big[&L\left(t,\, X_t,\, \nabla_x V(t,\, X_t)\sigma(t,\, X_t)\right) \\ &- \nabla_x V(t,\, X_t)\psi\left(t,\, X_t,\, \nabla_x V(t,\, X_t)\sigma(t,\, X_t)\right)\Big]\mathrm{d}t + \nabla_x V(t,\, X_t)\sigma(t,\, X_t)\mathrm{d}W_t. \end{aligned}$$

The terminal condition $V(T,\, X_t) = g(X_T)$ is a direct consequence of V being the solution to (3.33). ☐

3.6.1 Affine state equation

Often in optimal control, so-called affine state equations arise. An ***affine state equation*** is on the form

$$\mathrm{d}X_t = \left[\mu_1(t,\, X_t) + \mu_2(t,\, X_t)u_t^*\right]\mathrm{d}t + \sigma(t,\, X_t)\mathrm{d}W_t,\; t \in (0,\, T]; \quad X_0 = x_0. \tag{3.35}$$

With a state equation as (3.35) we can obtain a decoupled FBSDE by letting $\psi(t,\, X_t,\, \nabla_x V(t,\, X_t)\sigma(t,\, X_t)) = \mu_2(t,\, X_t)u_t^*$ in Proposition 1. The obtained BSDE reads as

$$\mathrm{d}V(t,\, X_t) = -\left[\tilde{L}\left(t,\, X_t,\, \nabla_x V(t,\, X_t)\sigma(t,\, X_t)\right) - \mu_2(t,\, X_t)u_t^*\right]\mathrm{d}t \tag{3.36}$$

$$+\nabla_x V(t, X_t)\sigma(t,\, X_t)\mathrm{d}W_t,\; t \in [0,\, T); \quad V(T,\, X_T) = g(X_T). \tag{3.37}$$

Again, letting Y_t and Z_t be defined by

$$Y_t = V(t,\, X_t), \quad Z_t = \nabla_x V(t,\, X_t)\sigma(t,\, X_t),$$

and letting $f(t,\, X_t,\, Z_t) = \tilde{L}\left(t,\, X_t,\, Z_t\right) - \mu_2(t,\, X_t)u_t^*$ we obtain the decoupled FBSDE

$$\begin{cases} \mathrm{d}X_t = \mu_1\Big(t,\, X_t\Big)\mathrm{d}t + \sigma(t,\, X_t)\,\mathrm{d}W_t,\; t \in (0,\, T]; \quad X_0 = x_0, \\ -\mathrm{d}Y_t = f\Big(t,\, X_t,\, Z_t\Big)\mathrm{d}t - Z_t\,\mathrm{d}W_t, t \in [0,\, T); \quad Y_T = g(X_T). \end{cases} \tag{3.38}$$

The system above is a decoupled FBSDE of the type introduced in Section 2.4.1.

3.6.2 The linear quadratic regulator

As a simple example, we consider the scalar valued linear SDE with quadratic cost function. This is called a linear quadratic regulator (LRQ). Later in this section we present an analytic solution to the LQR problem which will be used as a benchmark solution to test the performance of the numerical methods for this problem.

3.6.2.1 Reformulation of the control problem to the FBSDE

Consider the scalar valued linear SDE

$$\mathrm{d}\bar{X}_t = a(c - \bar{X}_t)\,\mathrm{d}t + \sigma\,\mathrm{d}W_t,\ t \in (0,\,T]; \quad \bar{X}_0 = x_0, \tag{3.39}$$

for $c \in \mathbb{R}$ and $a,\,\sigma \in \mathbb{R}^+$. Our aim is to control the process around 0, which is referred to as the optimal state for X, by adding a control function in the drift term of (3.39). For $b \in \mathbb{R}$ and $u_t \in \mathbb{R}$ for $t \in [0,\,T)$ the SDE in the LQR problem has the form

$$\mathrm{d}X_t = (a(c - X_t) + bu_t)\,\mathrm{d}t + \sigma\,\mathrm{d}W_t,\ t \in (0,\,T]; \quad X_0 = x_0. \tag{3.40}$$

The quadratic cost function is set to

$$J(t,\,x\,;u) = \mathbb{E}\left[\int_t^T \left(\gamma^2 X_s^2 + r^2 u_s^2\right)\,\mathrm{d}s + \lambda^2 X_T^2 \,\middle|\, X_t = x\right], \tag{3.41}$$

for $\gamma,\,r,\,\lambda \in \mathbb{R}^+$. With the notation from previous sections we have

$$\mu(t,\,X_t,\,u_t) = a(c - X_t) + bu_t, \tag{3.42}$$

$$\sigma(t,\,X_t) = \sigma, \tag{3.43}$$

$$L\,(t,\,X_t,\,u_t) = \gamma^2 X_t^2 + r^2 u_t^2, \tag{3.44}$$

$$g(X_T) = \lambda^2 X_T^2. \tag{3.45}$$

The intuitive explanation of the control problem is the following: The uncontrolled Ornstein-Uhlenbeck process (3.39) has a drift that strives to push the process towards c with mean reversion proportional to a and a diffusion term which disturbs the process with Gaussian additive noise generated by a standard Wiener process. In (3.40) a control process, which pushes the process with strength b, is added. The running cost (3.44) penalizes the squared distance from the optimal state with intensity γ^2 and the cost of the control is r^2. The terminal cost (3.45) penalizes the squared distance from the optimal state by λ^2. The optimal control at time $t \in [0,\,T)$ is then chosen so that (3.41) is minimized.

The backward operator (3.11), in this case is given by

$$\mathcal{A}^u\Phi(t,\,x) = \frac{\partial\Phi}{\partial t}(t,\,x) + \frac{\sigma^2}{2}\frac{\partial^2\Phi}{\partial t^2}(t,\,x) + (a(c-x) + bu_t)\frac{\partial\Phi}{\partial t}(t,\,x).$$

By equation (3.23) the optimal control u^* for this problem is given by

$$\begin{aligned} u_t^* =& \pi^*(t, X_t) = \underset{u\in\mathbb{R}}{\arg\min}[\mathcal{A}^u V(t, x) + L(t, x, u)]\Big|_{x=X_t} \\ =& \underset{u\in\mathbb{R}}{\arg\min}\left[\frac{\partial V}{\partial t}(t, x) + \frac{\sigma^2}{2}\frac{\partial^2 V}{\partial x^2}(t, x) + (a(c-x)+bu)\frac{\partial V}{\partial x}(t, x)\right. \\ &\left. + \gamma^2 x^2 + r^2 u^2\right]\Bigg|_{x=X_t} = -\frac{b}{2r^2}\frac{\partial V}{\partial x}(t, X_t). \end{aligned} \tag{3.46}$$

By reconnecting to the DPE control strategy from Section 3.2 we obtain a mapping ν of the desired form

$$\nu\left(t, X_t, \frac{\partial V}{\partial x}(t, X_t)\right) = -\frac{b}{2r^2}\frac{\partial V}{\partial x}(t, X_t).$$

The HJB equation is then the nonlinear PDE

$$\begin{aligned} \frac{\partial V}{\partial t}(t, x) &+ \frac{\sigma^2}{2}\frac{\partial^2 V}{\partial x^2}(t, x) + a(c-x)\frac{\partial V}{\partial x}(t, x) + \gamma^2 x^2 \\ &- \frac{b^2}{4r^2}\left(\frac{\partial V}{\partial x}(t, x)\right)^2 = 0, \ (x, t) \in (0, T) \times \mathbb{R}, \end{aligned} \tag{3.47}$$

$$V(T, x) = \lambda^2 x^2, \ x \in \mathbb{R}. \tag{3.48}$$

By defining $Y_t = V(t, X_t)$ and $Z_t = \sigma\frac{\partial V}{\partial x}(t, X_t)$ we obtain the FBSDE

$$\begin{cases} \mathrm{d}X_t = \left[a(c - X_t) - \frac{b^2}{2\sigma^2 r^2}Z_t^2\right]\mathrm{d}t + \sigma\mathrm{d}W_t, \ t \in (0, T]; \quad X_0 = x_0, \\ \mathrm{d}Y_t = -\left[\gamma^2 X_t^2 + \frac{b^2}{4\sigma^2 r^2}Z_t^2\right]\mathrm{d}t + Z_t\mathrm{d}W_t, \ t \in [0, T); \quad Y_T = \lambda^2 X_T^2. \end{cases} \tag{3.49}$$

Note that (3.49) is a FBSDE with a coupled forward SDE since it depends on Z. To obtain a decoupled FBSDE we use Proposition 1 and let $\psi(t, X_t, Z_t) = -\frac{b^2}{2\sigma^2 r^2}Z_t^2$ which gives

$$\begin{cases} \mathrm{d}\bar{X}_t = a(c - \bar{X}_t)\mathrm{d}t + \sigma\mathrm{d}W_t, \ t \in (0, T]; \quad \bar{X}_0 = x_0, \\ \mathrm{d}\bar{Y}_t = -\left[\gamma^2 \bar{X}_t^2 - \frac{b^2}{4\sigma^2 r^2}\bar{Z}_t^2\right]\mathrm{d}t + \bar{Z}_t\mathrm{d}W_t, \ t \in [0, T); \quad \bar{Y}_T = \lambda^2 \bar{X}_T^2. \end{cases} \tag{3.50}$$

The mapping κ from the FBSDE control strategy in Section 3.2 is given by

$$\kappa(t, X_t, Z_t) = -\frac{b}{2\sigma^2 r^2}Z_t.$$

3.6.2.2 Analytic solution via a system of ODEs

In this special case it is possible to find an analytical solution to both the HJB equation (3.47)-(3.48) and the FBSDEs (3.49) and (3.50). To this aim we make the ansatz $V(t, x) = P(t)x^2 + Q(t)x + R(t)$. When inserting this into (3.47) we obtain

$$\begin{aligned} & x^2\left(\dot{P}(t) - 2aP(t) + b^2P(t)^2 + \gamma^2\right) + x\left(\dot{Q}(t) - acP(t) - bQ(t) - b^2P(t)Q(t)\right) \\ & + \dot{R}(t) + \sigma^2 P(t) + ac\,Q(t) - \frac{b^2}{4}Q(t)^2 = 0, \ (t, x) \in [0, T) \times \mathbb{R}, \\ & V(T, x) = \lambda^2 x^2, \ x \in \mathbb{R}. \end{aligned}$$

Since this holds for all $x \in \mathbb{R}$ we obtain the system of ODEs

$$\begin{cases} \dot{P}(t) - 2bP(t) + a^2 P(t)^2 + \gamma^2 = 0, \ t \in [0, T); \quad P(T) = \lambda^2, \\ \dot{Q}(t) - b\mu P(t) - bQ(t) - a^2 P(t)Q(t) = 0, \ t \in [0, T); \quad Q(T) = 0, \\ \dot{R}(t) + \sigma^2 P(t) + b\mu Q(t) - \frac{a^2}{4} Q(t)^2 = 0, \ t \in [0, T); \quad R(T) = 0. \end{cases} \tag{3.51}$$

Assume that there exists a unique solution (P, Q, R), where $P = \{P(t)\}_{t\in[0,T]}$, $Q = \{Q(t)\}_{t\in[0,T]}$ and $R = \{R(t)\}_{t\in[0,T]}$ to (3.51). We then obtain an analytical solution $(x, y, z) = (X, Y, Z)$ to (3.49) or $(x, y, z) = (\bar{X}, \bar{Y}, \bar{Z})$ to (3.50) by

$$y_t = P(t)x_t^2 + Q(t)x_t + R(t), \tag{3.52}$$

$$z_t = \sigma(2P(t)x_t + Q(t)). \tag{3.53}$$

4
The neural network based algorithms

Artificial neural networks or simply neural networks are one of the hottest topics today and are widely used in different arenas. Neural networks are inspired by the architecture and interaction between neurons in a human brain. Although the artificial neurons are highly simplified the fundamental principles are similar. It is common to separate between supervised and unsupervised learning. In ***supervised learning*** the network learns to recognize structures in a set of training data with associated labels and generalize to unseen data. One example, which is explained in Section 4.1.3, is the classification problem *e.g.,* to classify handwritten digits. Another example where supervised learning is useful is regression. In ***unsupervised learning***, on the other hand, the training data is not labeled and the goal is instead to learn some natural structure within the set of training data. One example of unsupervised learning, further described in Section 4.1.3 is approximate equation solving.

This chapter aims to define all the techniques used in this thesis but for detailed motivations we refer to *e.g.,* [36]. In Section 4.1 the main building blocks for a neural network are presented and in Section 4.2 we present the neural network based algorithms used to solve stochastic optimal control problems.

4.1 Deep neural networks - function approximators

This section aims to describe the main building blocks of a neural network. In Section 4.1.1 the network is described as a mapping and in Section 4.1.2 a low dimensional mapping is visualized. The goal of Sections 4.1.3 and 4.1.4 is to describe how the network can be trained in order to approximate some function.

4.1.1 The structure of an artificial neural network

Let $D \in \mathbb{N}^+$ be the number of trainable ***parameters*** in a neural network and $\Theta \subseteq \mathbb{R}^D$ be the set of all feasible parameters. The neural network is a composition of $L \geq 1$ so called ***layers*** with $k_0, k_1, \ldots, k_L \in \mathbb{N}^+$ so called ***nodes*** (or ***neurons***) in each layer. More precisely, it is a family of functions

$$\mathrm{NN}^\theta \colon \mathbb{R}^{k_0} \to \mathbb{R}^{k_L}, \quad \theta \in \Theta,$$

of the form

$$\mathrm{NN}^{\theta} = \mathfrak{L}_L^{\theta_L} \circ \cdots \circ \mathfrak{L}_2^{\theta_2} \circ \mathfrak{L}_1^{\theta_1},$$

parametrized by $\theta = (\theta_1, \theta_2, \ldots, \theta_L) \in \Theta$. Each layer $\mathfrak{L}_n^{\theta_n} \colon \mathbb{R}^{k_{n-1}} \to \mathbb{R}^{k_n}$ for $n = 1, 2, \ldots, L$, is a function of the form

$$\mathfrak{L}_n^{\theta_n}(x) = H_n(W_n x + b_n), \quad \theta_n = (\mathrm{vec}(W_n), b_n), \quad \text{with}$$

$$H_n(x) = \begin{pmatrix} h(x_1) \\ h(x_2) \\ \vdots \\ h(x_{k_{n+1}}) \end{pmatrix}, \quad \text{for } n < L \text{ and } \quad H_L(x) = \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_{k_L} \end{pmatrix},$$

with so called weight matrices $W_n \in \mathbb{R}^{k_n \times k_{n-1}}$ and bias vectors $b_n \in \mathbb{R}^{k_n}$. The function $h\colon \mathbb{R} \to \mathbb{R}$ is called the activation function. Note that by using the identity activation function, *i.e.*, letting $h(y) = y$, for $y \in \mathbb{R}$, the neural network becomes an affine mapping. The purpose of the activation function is to add non-linearity to the network, which in turn admits approximation of non-affine mappings. Commonly used activation functions are the hyperbolic tangent function $\tanh(y) = \frac{e^{2y}-1}{e^{2y}+1}$, the sigmoid function $\sigma(y) = \frac{1}{1+e^{-y}}$ and the ReLU function $\mathrm{ReLU}(y) = \max\{0, y\}$, for $y \in \mathbb{R}$. In this thesis we use the ReLU function, which was suggested in [37] with a biological motivation. In [38], the authors argued that using the ReLU function during the training procedure, the output of many neurons in the hidden layers equal to zero. This makes the network of active neurons (with nonzero output) sparsely connected, which is thought of as a desirable property [38].

If $L > 2$ the neural network is said to be deep and the layers $\mathfrak{L}_n^{\theta_n}$ for $n < L$ are referred to as the hidden layers. A neural network is often visualized by a graph consisting of nodes and edges, see, e.g., the two upper plots in Figure 4.1. The nodes to the very left has the values of the input $y_0 = x = (x_1, \ldots, x_{k_0})$ of the network. The value of the n:th nodes from the left have the values

$$y_n = \mathfrak{L}_n^{\theta_n}(y_{n-1}) = \begin{pmatrix} h(W_n^1 y_{n-1} + b_n^1) \\ h(W_n^2 y_{n-1} + b_n^2) \\ \vdots \\ h(W_n^{k_n} y_{n-1} + b_n^{k_n}) \end{pmatrix},$$

propagating information from the left to the right in the graph until the output layer to the far right. Here W_n^m and b_n^m denote the m:th rows of the weight matrix W_n and bias vector b_n of the n:th layer, respectively. Even more explicit, a node performs the action $y_n^m = h(W_n^{m,1} y_{n-1}^1 + \cdots + W_n^{m,k_{n-1}} y_{n-1}^{k_{n-1}} + b_n^m)$, where $W_n^{m,k}$ is the (m,k):th element of W_n. From this it is clear that each edge in the graph corresponds to the multiplication of a weight. This is the full interpretation of a neural network as a network or graph.

4.1.2 Example of a low dimensional neural network

The purpose of this example is to visualize a neural network and to show how the flexibility of the output depends on the number of layers and the number of nodes in each layer. This is done by constructing neural networks with randomly selected parameters θ. Samples from these neural networks are then generated with different number of layers and nodes in each layer. Although the results are random, we will be able to see structural differences in the output for the different neural networks. We note that to be able to approximate a specific function, the parameter θ cannot be selected at random but must be optimized for this specific function. This optimization is the so-called training of a neural network and is described later in this chapter.

Consider the neural network $\mathrm{NN}^\theta\colon \mathbb{R}^2 \to \mathbb{R}$ on the square $\mathcal{D} = [-1, 1] \times [-1, 1]$. We let all elements in the weight matrices $W_1, W_2, \ldots, W_L$ and the bias vectors $b_1, b_2, \ldots, b_L$ be independent, identically distributed random variables with a uniform distribution on $[-0.5, 0.5]$. in Figures 4.1 and 4.2 we have the following setting: in the plots to the left $L = 2$ and to the right $L = 4$. The upper plots show the structure of the neural network and the lower plots show outcome of one experiment. The number of nodes in each hidden layer is 4 in Figure 4.1, and it is 15 in Figure 4.2.

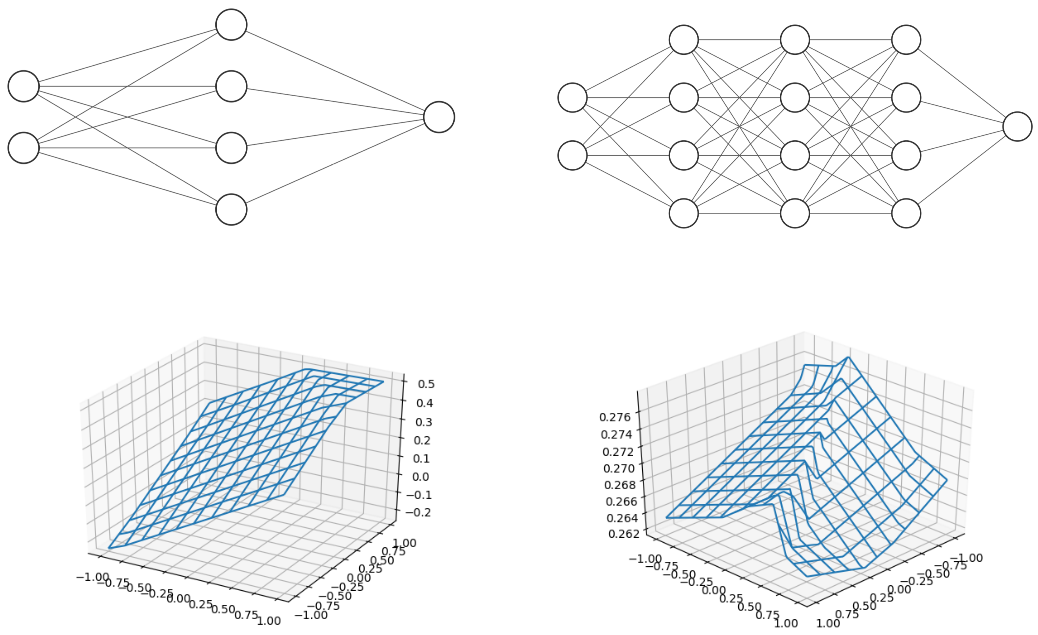


Figure 4.1: A neural network mapping, $\mathrm{NN}^\theta\colon \mathbb{R}^2 \to \mathbb{R}$. The upper plots show the structure of the network and the lower plots show the graphs of NN^θ for randomly sampled θ. Left: $L = 2$ with $k_1 = 4$. Right: $L = 4$ with $k_1 = k_2 = k_3 = 4$.

4.1.3 The loss function

In Section 4.1.1 we described a neural network as a mapping from $\mathbb{R}^{k_0}$ to $\mathbb{R}^{k_L}$. To simplify the presentation we suppress the dimensions and consider the neural

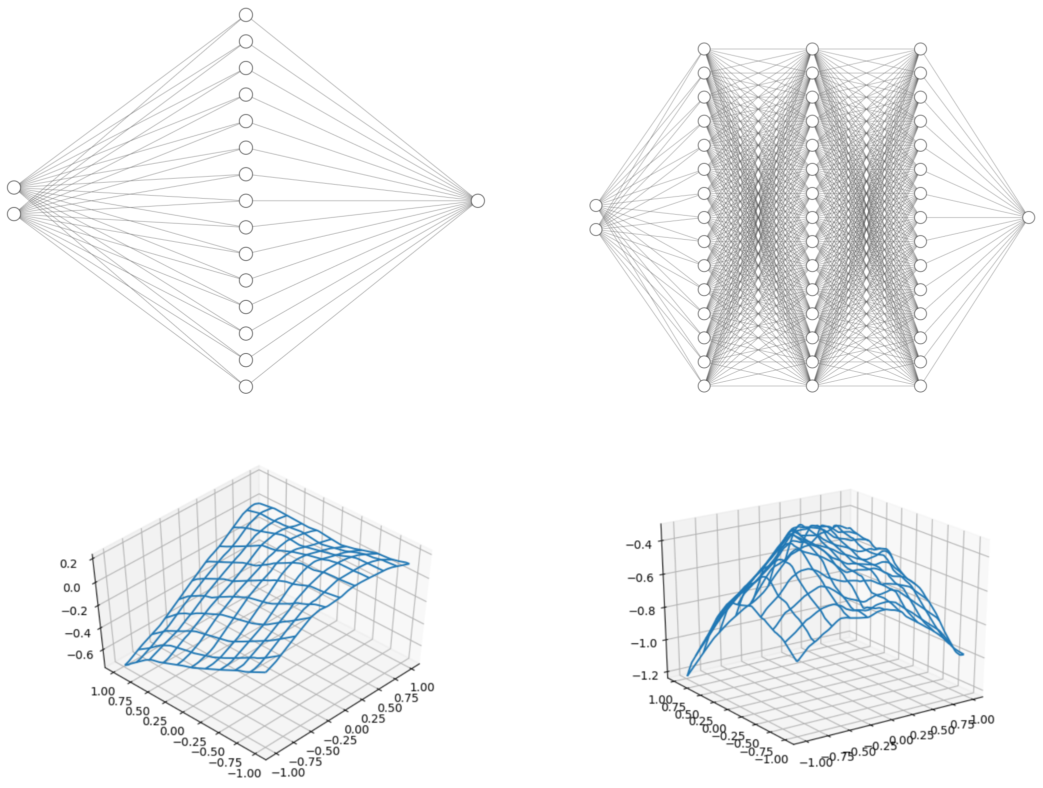


Figure 4.2: A neural network mapping, $\mathrm{NN}^{\theta}\colon \mathbb{R}^2 \to \mathbb{R}$. The upper plots show the structure of the network and the lower plots show the graphs of NN^{θ} for randomly sampled θ. Left: $L = 2$ with $k_1 = 15$. Right: $L = 4$ with $k_1 = k_2 = k_3 = 15$.

network as a mapping

$$\mathrm{NN}^{\theta}\colon \mathcal{X} \to \mathcal{Z},$$

parameterized by $\theta \in \Theta$ for some sets $\mathcal{X}, \mathcal{Z}$. In every application of neural networks the parameters are tuned or trained to satisfy a desired input/output relation of the network. The function that quantifies the goodness of fit is called the ***loss function*** and it is used as objective function in the minimization problem of training the network. The loss function and the training procedure vary depending on the function that is to be approximated, *i.e.,* on the application of the neural network. In this section we give examples of three different types of neural networks and introduce their corresponding loss functions. In all three examples we assume that the neural networks are trained with training data in $\mathcal{X} \times \mathcal{Y}$ sampled from a probability distribution Q on $\mathcal{X} \times \mathcal{Y}$. The space $\mathcal{Y}$, referred to as the ***label space*** contains additional information about the data and is essential in supervised learning. The distribution Q can be a mathematical distribution or it can be a real world distribution. An example of the former is the multivariate normal distribution and an example of the latter is the joint probability distribution between body mass index $x \geq 0$ and diabetes $y \in \{0, 1\}$ within the global population at a given time. No one knows the distribution exactly but one can make random samples from it and this is the important feature here.

Example 1. (Regression) In this case we let $\mathcal{Y} = \mathcal{Z}$ be a normed vector space with norm $\|\cdot\|_{\mathcal{Y}}$. The network is trained to estimate the conditional expectation $\mathbb{E}[Y|X]$ for $(X, Y) \sim Q$. The goal of the loss function is to evaluate how well the network has captured the relationship and is often chosen as

$$\mathrm{Loss}(\theta\,;\,Q) = \mathbb{E}_{(X,Y)\sim Q}\left[\|\mathrm{NN}^{\theta}(X) - Y\|_{\mathcal{Y}}^{2}\right].$$

In practice only finite number of data pairs $(x_k,\, y_k)_{k=1}^{K} \subset \mathcal{X} \times \mathcal{Y}$ are available. Let $Q_{\text{empirical}} = \frac{1}{K}\sum_{n=1}^{K}\delta_{(x_k,\, y_k)}$ be the corresponding empirical distribution. The ***empirical loss function*** is given by

$$\mathrm{Loss}(\theta\,;\,Q_{\text{empirical}}) = \frac{1}{K}\sum_{k=1}^{K}\|\mathrm{NN}^{\theta}(x_k) - y_k\|_{\mathcal{Y}}^{2}.$$

The training procedure, described further in the next section, is to find $\theta^{*} \in \Theta$ such that the empirical loss function is minimized for $\theta = \theta^{*}$.

Example 2. (Classification) In the classification problem we let $\mathcal{Y} = \{1,\, 2,\, \ldots,\, C\}$ and $\mathcal{Z} = [0,\, 1]^{C}$. For $(X, Y) \sim Q$ the interpretation of $Y = c \in \mathcal{Y}$, is that X belongs to class c. For classification a so-called softmax output layers is used. If the output of the second last layer of the network is o then the output of the softmax layer is $(e^{o_1}, \ldots, e^{o_n})/\sum_{i=1}^{n} e^{o_i}$. Clearly the output sums to one. The neural network is trained to generate the conditional probability distribution $\mathbb{P}(Y = c \,|\, X)$ for $c \in \mathcal{Y}$. The loss function for the classification problem is often the so-called cross entropy loss, given by

$$\mathrm{Loss}(\theta\,;\,Q) = -\mathbb{E}_{(X,Y)\sim Q}\left[\sum_{c=1}^{C}\mathbf{1}_{\{Y=c\}}\ln\left(\mathrm{NN}^{\theta}(X)_c\right)\right],$$

where $\mathrm{NN}^\theta(X)_c$ is the c:th element in the output of the network. As in Example 1, an empirical distribution is used in practice.

Example 3. (Approximate equation solving) Consider the problem of finding a function $f\colon \mathcal{X} \to \mathcal{Z}$, in some class $\mathcal{D}(\Phi)$ satisfying

$$\Phi(f)(x) = 0, \quad x \in \mathcal{X}, \tag{4.1}$$

for some operator Φ. The network is trained to approximate f using data $x \in \mathcal{X}$, sampled from some distribution Q over $\mathcal{X}$. In this example $\mathcal{Y} = \emptyset$. This example include problems in many different areas, *e.g.,* discrete numerical schemes of ordinary and partial differential equations, both deterministic and stochastic. In the stochastic case the space $\mathcal{X}$ contains both time and space variables as well as discretizations of the noise. The loss function is abstractly given by

$$\mathrm{Loss}(\theta\,;\,Q) = \mathbb{E}_{X\sim Q}\left[\Psi(\mathrm{NN}^\theta(X))\right],$$

for some $\Psi\colon \mathcal{Z} \to \mathbb{R}^+$. One natural choice is to let $\Psi(z) = \|\Phi(z)\|_{\mathcal{D}}$ for some appropriate norm $\|\cdot\|_{\mathcal{D}}$. This is done in *e.g.,* [41], where (4.1) is a parabolic PDE and the norm is designed to take into account also the boundary and initial conditions. In the FBSDE controller described below, parts of the equation is built into the architecture of the network and Ψ handles the terminal condition of the backward equation. As in the previous examples, an empirical distribution is used in practice.

Example 4. (Approximate optimization) This class of problems is similar to the previous but instead of solving equations, optimization problems are considered. Let $\mathcal{D}(\Phi)$ be a class of functions $\mathcal{X} \to \mathcal{Z}$ and $\Phi\colon \mathcal{D}(\Phi) \to \mathbb{R}$, be a functional that is bounded from below. Consider the problem to find a function $f \in \mathcal{D}(\Phi)$ that solves the problem

$$\text{minimize } \Phi(f), \qquad\qquad \text{subject to } f \in \mathcal{D}(\Phi). \tag{4.2}$$

If $\{NN^\theta;\ \theta \in \Theta\} \subseteq \Phi(f)$ then instead of solving (4.2) we can solve the problem

$$\text{minimize } \Phi(\mathrm{NN}^\theta), \qquad\qquad \text{subject to } \theta \in \Theta, \tag{4.3}$$

and if $\{NN^\theta;\ \theta \in \Theta\}$ is rich enough, maybe the optimimum of (4.3) is a good approximation of that for (4.2). The loss function for this problem is naturally Φ itself. In Sections 4.2.1–4.2.2 below, the so called naive and DPE controllers are introduced and they rely on this principle.

4.1.4 Training of a neural network

As mentioned in the previous section, the ***training*** of a neural network is the process of finding θ^* such that the loss function is minimized for $\theta = \theta^*$, *i.e.,*

$$\theta^* \in \operatorname*{arg\,min}_{\theta\in\Theta} \mathrm{Loss}(\theta\,;\,Q). \tag{4.4}$$

In practice an empirical distribution $Q_{\text{empirical}}$ is used, and from now on we only consider empirical loss functions. The specific loss functions used in this thesis are explained and motivated in Section 4.2. The training goes as follows: Assign values, possibly at random, to θ. An appropriate optimization algorithm is then used to find θ^* such that $\text{Loss}(\theta^*\,;\, Q_{\text{empirical}}) < \text{Loss}(\theta\,;\, Q_{\text{empirical}})$ and let $\theta = \theta^*$. Ideally as in (4.4) but it is often hard to find a global minimum since θ is a high dimensional vector (often in at least 10^6 dimensions) and the loss function is often non convex.

After the training has terminated then hopefully the neural network has learned the relationship between the input and the output sufficiently well to generalize to data it has not yet seen during training. To test how well the network is able to generalize to new data, a test procedure is done. It is based on using another set of samples $(\tilde{x}_k, \tilde{y}_k)_{k=1}^K$, not used for training, and evaluate the prediction performance on that.

4.1.4.1 Optimization algorithms used in the training procedure

This section deals with Step 3 in the training algorithm described in Section 4.1.4. The problem is to find

$$\theta^* \in \underset{\theta\in\Theta}{\arg\min}\, \text{Loss}(\theta\,;\, Q_{\text{empirical}}). \tag{4.5}$$

There are several different numerical methods for finding θ^* and the vast majority of them stems from the gradient descent. The ***gradient descent*** is a simple algorithm that uses the (vector) gradient $\nabla_\theta \text{Loss}(\theta\,;\, Q_{\text{empirical}})$ to decide in which direction θ should be adjusted. Recall that the gradient of a function is a vector pointing in the direction of the greatest rate of increase of that function. Since we want to minimize $\text{Loss}(\theta\,;\, Q_{\text{empirical}})$ the gradient descent updates θ in the opposite direction of the gradient. Given a learning rate $\alpha > 0$ the update rule for the gradient descent reads as:

$$\theta_{t+1} = \theta_t - \alpha \nabla_\theta \text{Loss}(\theta_t\,;\, Q_{\text{empirical}}),\ \ t \in \mathbb{N}^+.$$

The iteration continues until convergence of θ. This method is very inefficient for large training sets since all the data is used every time to compute the gradient. Another problem is that if the algorithm easily reaches a bad local minimum it cannot escape. Due to these drawbacks classical gradient descent is rarely used in practice.

An updated version of the gradient descent is the ***stochastic gradient descent***. Instead of finding the gradient over the entire training data the stochastic gradient descent updates θ for each sample from the training data individually. We denote the empirical loss function, containing only one pair $(x,\, y)$ from the empirical distribution $Q_{\text{empirical}}$ by

$$\text{Loss}(\theta\,;\, x,\, y).$$

Given a learning rate $\alpha > 0$, the update rule for the most basic stochastic gradient descent is given by:

$$\theta_t \leftarrow \theta_t - \alpha \nabla_\theta \text{Loss}(\theta_t\,;\, x,\, y),$$

which is repeated for all pairs $(x,\, y) \in \operatorname{supp}(Q_{\text{empirical}})$. Looping through the data set once and updating the gradient for every data point is called an epoch. The training often consists of many episodes and the subscript t indicates which epoch we are in. When it is possible to sample directly from Q, for instance by some cheap computer simulation, then it is better to continue with new data in one long episode instead of using multiple episodes. Clearly this captures Q better than using the same samples over and over again. In practice it is common to shuffle the training data after each epoch to minimize the risk of providing the training data in a meaningful order, which may bias the optimization algorithm [44]. This method converges almost surly to a global minimum if the loss function is convex and to a local minimum otherwise. The parameters obtained from using stochastic gradient descent are often empirically better than the parameter computed by gradient descent [44]. By combing the gradient descent and the stochastic gradient descent the ***batch gradient descent*** can be obtained. Instead of computing the gradient and perform an update for every pair $(x,\, y)$ individually, this is is done over n mini batches. The update rule for the batch gradient descent is given by:

$$\theta_t \leftarrow \theta_t - \alpha \nabla_\theta \text{Loss}(\theta_t\, ;\, x_{i:i+n},\, y_{i:i+n}),$$

where $\text{Loss}(\theta_t\, ;\, x_{i:i+n},\, y_{i:i+n})$ is the empirical loss function containing n pairs from $Q_{\text{empirical}}$.

One problem with the algorithms above is that all the components of the parameter vector θ are updated with the same learning rate α. This might cause problems, *e.g.*, if the training data is sparse (containing a large proportion of zeros) or if the different components in θ has a very varying magnitude. There are several methods that uses different learning rates for each parameter such as ***Adagrad***, ***Momentum optimizer***, ***RMSProp*** and ***Adam optimizer***. The arguably most used algorithm within the neural network community [44] as well as the one used in this thesis is the Adam optimizer. The algorithm uses adaptive moment estimation to compute an individual learning rate for each parameter [45]. Let $g_t = \nabla_\theta \text{Loss}(\theta\, ;\, x,\, y)$ for some $(x,\, y) \sim Q_{\text{empirical}}$, the following estimates for the first and second moments (the mean and the uncentered variance) of the gradient of the loss function are then used

$$m_t = \beta_1 m_{t-1} + (1-\beta_1)g_t, \quad v_t = \beta_2 v_{t-1} + (1-\beta_2)g_t^2, \quad \beta_1, \beta_2 \in \mathbb{R}^+,\ t \in \mathbb{N}^+.$$

The initialization of m_0 and v_0 is set to be the zero vector. In [45] the authors claim that m and v are biased towards zero, especially during the initial iterations and when the decay rates are small, *i.e.* when β_1 and β_2 are close to 1. They therefore introduced the bias corrected moment estimates

$$\hat{m}_t = \frac{m_t}{1-\beta_1}, \quad \hat{v}_t = \frac{v_t}{1-\beta_2}.$$

Finally, the update rule for θ reads

$$\theta_{t+1} = \theta_t + \alpha \frac{\hat{m}_t}{\sqrt{\hat{v}_t} + \epsilon},$$

where ϵ is a small constant included for numerical stability. The update to the version where the gradient is computed for each element from the training data individually (similar to the stochastic gradient descent) is straight forward.

4.2 Neural network based algorithms for solving stochastic optimal control problems

The notation below is general and applies to all algorithms in this section. Consider a state equation given by

$$\mathrm{d}X_t = \mu(t,\, X_t,\, u_t)\,\mathrm{d}t + \sigma(t,\, X_t)\,\mathrm{d}W_t,\ t \in (0,\, T]; \quad X_0 = x_0, \tag{4.6}$$

and a cost functional, as described in Chapter 3, given by

$$J(t,\, x\,;\, u) = \mathbb{E}\left[\int_t^T L(s,\, X_s,\, u_s)\,\mathrm{d}s + g(X_T)\,\middle|\, X_t = x\right]. \tag{4.7}$$

The time interval $[0,\, T]$ is discretized into the equidistant grid $0 = t_0 < t_1 < \cdots < t_N = T$ with uniform step size $\Delta t = t_k - t_{k-1}$ for $k = 1,\, 2,\, \ldots,\, N$. We further denote $X_k = X_{t_k}$, $u_k = u_{t_k}$ and $W_k = W_{t_k}$ for $k = 0, 1, \ldots,\, N$ and $\Delta W_k = W_k - W_{k-1}$ for $k = 1,\, 2,\, \ldots,\, N$. In Figure 4.3–4.6 ↗ represents a deterministic function and ⇗ represents a neural network. The arrows are pointing from the input towards the output of the function and the neural network respectively. When there are multiple arrows pointing towards an output it should be interpreted as a function with multiple inputs. To approximate (4.6) we use the ***Euler–Maruyama scheme*** which is a finite difference type method. The Euler–Maruyama scheme is the most basic method for approximating SDEs. The algorithm is easy and straight forward to implement but it can sometimes be unstable in the sense that the approximation can blow up even though the solution does not. In Section 5.3 we therefore investigate convergence properties for the Euler–Maruyama scheme for the case when the state equation describes an inverted pendulum on a cart, which is the main state equation considered in this thesis. In the following subsections the main algorithms of this thesis are outlined. Note that the initial state X_0 is an input to the first neural network in each algorithm which in turn effects X_k for $k > 0$. By sampling many different X_0 during training from a spatial domain we train the networks to be able to solve stochastic optimal control problems where the initial state varies.

In the following sections, the main algorithms used to solve the stochastic optimal control problem are presented. All the algorithms below are of the type introduced in Examples 3 and 4. The initial state are given by $X_0 \sim U(\mathcal{D}_{X_0})$ for some spatial domain of interest $\mathcal{D}_{X_0} \subset \mathbb{R}^n$ and the increments of the d-dimensional Wiener process ΔW_k satisfies $(\Delta W_k)_j \sim \mathcal{N}(0,\, h)$ for $j = 1,\, 2\, \ldots,\, d$ for all $k = 1,\, 2,\, \ldots,\, N$. This means that, with notation as in Section 4.1.3 we have $\mathcal{X} = \mathcal{D}_{X_0} \times \mathbb{R}^{d\times N}$ and $\mathcal{Y} = \emptyset$. By assuming that $(\Delta W_k)_j$ and $(\Delta W_l)_m$ are independent for all $(k,\, l) \neq (j,\, m)$ and using $(\cdot)^*$ to denote the transpose of a matrix, we obtain the random vector $((\Delta W_1)^*,\, (\Delta W_2)^*,\, \ldots,\, (\Delta W_N)^*)$ taking on values in $\mathbb{R}^{N\cdot d}$. The random vector is multivariate normal distributed with mean zero and a covariance matrix with Δt on the diagonal and zero elsewhere. For notational convenience we denote $q = ((X_0)^*, (\Delta W_1)^*, (\Delta W_2)^*, \dots, (\Delta W_N)^*)$ and $Q = U(\mathcal{D}_{X_0}) \otimes \mathcal{N}(\mathbf{0}, \operatorname{diag}(\Delta t, \dots, \Delta t))$ to obtain the compact notation $q \sim Q$. Note that in this specific case it is possible to sample from the exact distribution Q and no empirical distribution is needed. The loss function, on the other hand, must be approximated by a Monte-Carlo method.

4.2.1 The naive controller

The naive controller is called ”naive” since it uses only the formulation of the control problem stated in (4.6), (4.7) and relies on no theoretical insights of the problem. In each time step the current state is used as input to a neural network which outputs the control signal. This method was introduced in [46], in which it was shown to perform well in comparison to benchmark problems from finance and energy storage.

<u>The naive controller:</u>

For each $n = 0, 1, \dots, N-1$ do:
(1) Generate the control signal u_n with a neural network $X_n \mapsto u_n$ with parameters θ^n.
(2) Update the state equation by the Euler–Maruyama update rule:

$$X_{n+1} = X_n + \mu(t_n, X_n, u_n)\Delta t + \sigma(t_n, X_n)\Delta W_n.$$

In Figure 4.3 the structure of the naive control algorithm is visualized. The parameters in all the parallel neural networks are denoted by Θ and given by $\boldsymbol{\theta} = (\theta^0, \theta^1, \dots, \theta^{N-1})$. The notation for $\boldsymbol{\theta}$ should not be confused with the notation for the layers of a neural network introduced in Section 4.1. Note that $\boldsymbol{\theta}$ represents the parameters for N parallel neural network where each θ^n, for $n = 0, 1, \dots, N-1$, itself has the structure described in Section 4.1.

The loss function, which is a time discrete version of the cost functional (4.7) is given by

$$\text{Loss}^{\text{naive}}(\boldsymbol{\theta}\,; Q) = \mathbb{E}_{q\sim Q}\left[\sum_{n=0}^{N-1} L(t_n, X_n, u_n)h + g(X_N)\right].$$

The loss function is the approximated as

$$\text{Loss}^{\text{naive}}(\boldsymbol{\theta}\,; Q) \approx \frac{1}{K}\sum_{k=1}^{K}\left(\sum_{n=0}^{N-1} L(t_n, X_n^k, u_n^k)h + g(X_N^k)\right),$$

where X_n^k, u_n^k for $n = 1, 2 \dots, N$ are realizations from the neural network with $\mathbf{X_k}$, sampled from Q, as input for $k = 1, 2 \dots K$. In literature, K is often referred to as the ***batch size***. For the numerical experiments in this thesis we use a single batch, *i.g.*, $K = 1$ while training the network. To compare the performance of the trained networks the loss function is approximated with $K = 5$ with new, unseen test data as input.

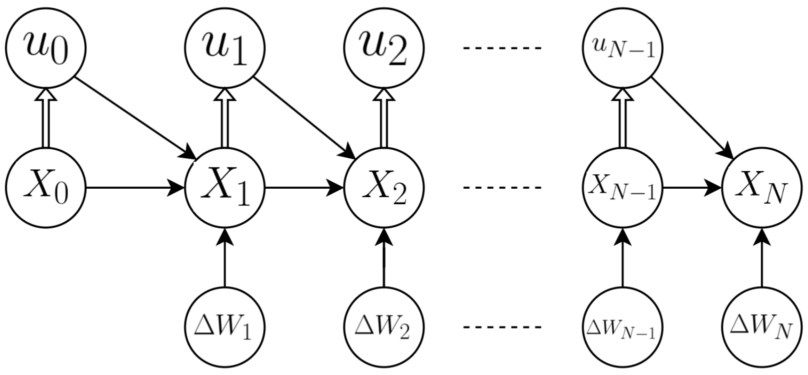


Figure 4.3: A schematic picture of the naive controller.

4.2.2 The DPE controller

In the DPE controller we use the dynamic programming equation stated in Section 3.3 which establishes a connection between the optimal control signal and the gradient of the value function. In many stochastic optimal control problems, including those considered in this thesis the optimal control signal can be written as an explicit function π of t, X_t and $\nabla_x V(t, X_t)$. This relationship is used in such a way that we let the neural network approximate the gradient of the value function in each discrete time step, from which we then obtain the control signal.

The DPE controller:

For each $n = 0, 1, \dots, N-1$ do:

(1) Generate the gradient of the solution to the HJB equation $\nabla V_n = \nabla V(t_n, X_n)$ with a neural network $X_n \mapsto \nabla V_n$ with parameters θ_n.
(2) Compute the control signal $u_n = \pi(t_n, X_n, \nabla V_n)$.
(3) Update the state equation by the Euler–Maruyama update rule:

$$X_{n+1} = X_n + \mu(t_n, X_n, u_n)\Delta t + \sigma(t_n, X_n)\Delta W_n.$$

In Figure 4.4 the structure of the DPE control algorithm is visualized. Like the previous algorithm the parameters in all the parallel neural networks are denoted by $\boldsymbol{\theta}$ and given by $\boldsymbol{\theta} = (\theta^0, \theta^1, \dots, \theta^{N-1})$ and the loss function is, again, given by

$$\text{Loss}^{\text{DPE}}(\boldsymbol{\theta}\,; Q) = \mathbb{E}_{q\sim Q}\left[\sum_{n=0}^{N-1} L(t_n, X_n, u_n)h + g(X_N)\right].$$

Thus, the DPE controller minimizes the same loss function as the naive controller, but it utilizes the structure of the problem better. As for the naive controller, the loss function is approximated with the Monte-Carlo method with K realizations.

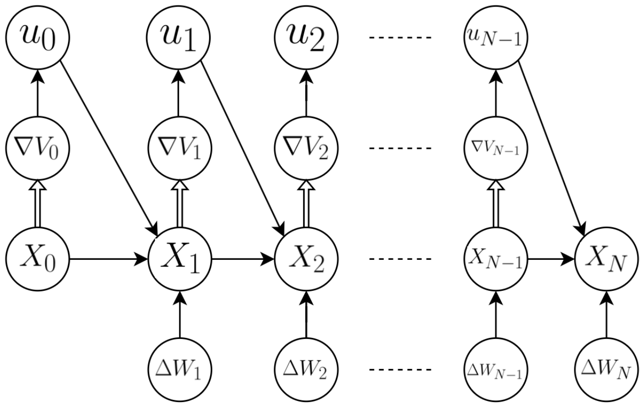


Figure 4.4: A schematic picture of the DPE controller.

4.2.3 The FBSDE controller

The FBSDE controller uses the reformulation of a stochastic optimal control problem to a FBSDE described in Section 3.5, thus being the controller we consider that relies on most theoretical understanding of the problem. The FBSDE is of the form given in (3.30) and we use a neural network to approximate Y_0, *i.e.*, the value of the BSDE part at the initial time $t = 0$. In each time step t_n, a neural network is used to approximate Z_n which in turn gives the control signal $u_n = \pi(t_n, X_n, Z_n)$ by an explicit mapping as described in Section 3.5.

The FBSDE controller:

(0) Generate Y_0 with a neural network $X_0 \mapsto Y_0$ parametrized by θ_0^Y.

For each $n = 0, 1, \ldots, N-1$ do:

(1) Approximate $Z_n = Z_{t_n}$ with a neural network $X_n \mapsto Z_n$ with parameters θ_n^Z.

(2) Update the FBSDE by the Euler–Maruyama update rule:

$$X_{n+1} = X_n + \mu(t_n, X_n, Z_n)\Delta t + \sigma(t_n, X_n)\Delta W_n,$$
$$Y_{n+1} = Y_n - f(t_n, X_n, Z_n)\Delta t + Z_n\Delta W_n.$$

In Figure 4.5 the structure of the FBSDE control algorithm is visualized. The parameters of all the parallel neural networks are denoted by
$\boldsymbol{\theta} = (\theta^{Y_0}, \theta^{Z_0}, \theta^{Z_1}, \ldots, \theta^{Z_{N-1}})$. We use the terminal condition of the BSDE, $Y_N = g(X_N)$ to construct the loss function

$$\text{Loss}^{\text{FBSDE}}(\boldsymbol{\theta}\,;\, Q) = \mathbb{E}_{q\sim Q}\left[(g(X_N) - Y_N)^2\right].$$

Once again, the loss function is approximated with the Monte-Carlo method with K samples.

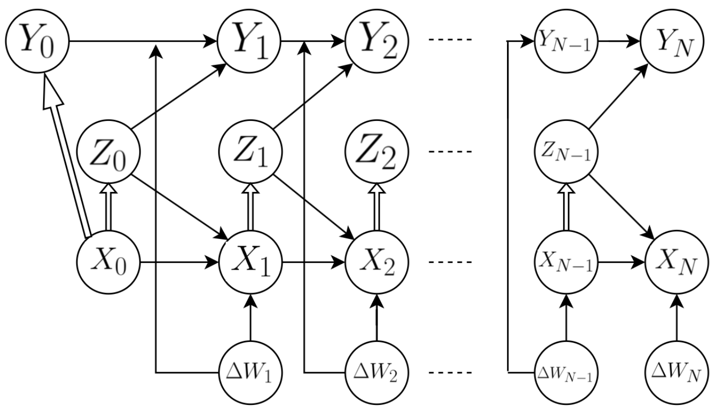


Figure 4.5: A schematic picture of the FBSDE controller.

4.2.4 The Deep BSDE Solver

The Deep BSDE solver is basically the same algorithm as the Deep FBSDE Solver. The difference lies in the reformulation of the stochastic optimal control problem. The Deep FBSDE Solver explained in Section 4.2.3 solves a FBSDE. The Deep BSDE Solver on the other hand consider a decoupled FBSDE which means that the forward SDE can be approximated at all time points in advance and separately from the BSDE. The Deep BSDE Solver then uses a neural network to solve the remaining BSDE. The algorithm was introduced in [3] and the original purpose was to overcome the curse of dimensionality when solving high dimensional PDEs. This was done by solving a decoupled FBSDE of the form (3.38), *i.e.*, finding $(X,\, Y,\, Z)$. By the definition of Y we have that Y_t is the solution of the associated HJB equation evaluated at $(t,\, X_t)$. The solution to the HJB equation at the point $(0,\, X_0)$ is given by Y_0, *i.e.* $V(0,\, X_0) = Y_0$ and the sought solution $V(0,\, x_0)$ is obtained.

When the purpose instead is to solve a stochastic optimal control problem the approach is not as obvious as when we solve PDEs. We first need to reformulate the stochastic optimal control problem to a decoupled FBSDE (3.38). Since the state equation is not controlled its state will drift or diffuse away from the typical state of a controlled dynamics. Training along such paths can lead to neural networks only being able to generate Z_n outside the region of interest for the controlled dynamics. The reason for including the deep BSDE controller is that it is the most direct generalization of the method from [3] which was the starting point of this thesis.

As we can see in Figure 4.6 the algorithm is the same as the Deep FBSDE Solver expect that there are no arrow from Z to X.

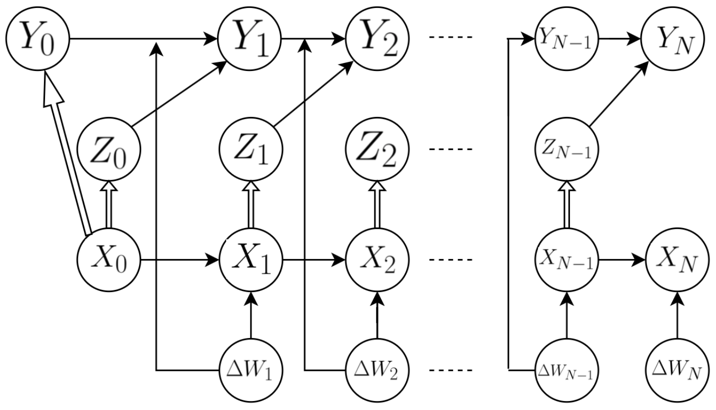


Figure 4.6: A schematic picture of the Deep BSDE solver.

5

Inverted pendulum on a cart

In this section the state equations for the single, and double inverted pendulum on a cart are derived. This is done from the Newtonian equations of motion and leads to a four and six dimensional ODE for the single and double inverted pendulums, respectively. From now on we denote the dimension by n and keep in mind that $n = 4$ for the single inverted pendulum and $n = 6$ for the double inverted pendulum. The dynamics of the pendulums are described by equations of the form

$$\dot{X}_t = f(X_t, u_t), \quad t \in (0, T]; \; X_0 = x, \tag{5.1}$$

where $\dot{X}_t$ is the time derivative of X at time t, the function $f \colon \mathbb{R}^n \times \mathbb{R} \to \mathbb{R}^n$. Let $W = \{W_t\}_{t \in [0,T]}$ be an n-dimensional Wiener process. Additive noise is then added to (5.1) to obtain

$$\mathrm{d}X_t = f(X_t, u_t)\mathrm{d}t + \sigma \mathrm{d}W_t, \; t \in (0, T]; \quad X_0 = x, \tag{5.2}$$

where $\sigma \in \mathbb{R}^{n \times n}$ is a constant, invertible matrix. Note that the equation above is a forward SDE described in Chapter 2. In the final section of this chapter convergence properties of finite difference methods for these equations are investigated.

5.1 Single inverted pendulum on a cart

Consider a single pendulum placed on a cart, see Figure 5.1. The cart is assumed

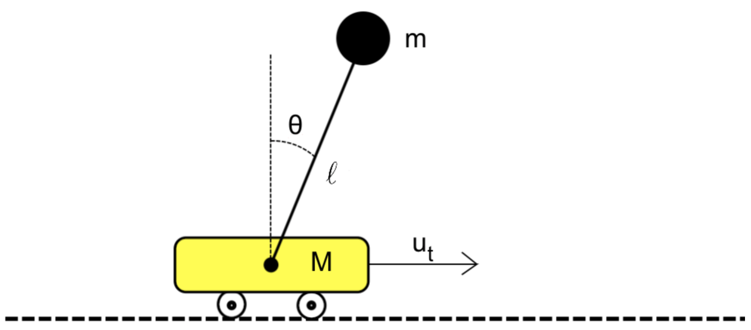


Figure 5.1: Inverted pendulum on a cart.

to move friction less in x-direction and the arm of the pendulum is assumed to be

Notation	
M	Mass of the cart.
m	Mass of the ball.
ℓ	Length of the pendulum.
θ_t	Angle of the pendulum ($\theta = 0$ when the pendulum is pointing up).
x_t	Position of the cart in x-direction.
u_t	Control of the system.

Table 5.1: Notation for the single inverted pendulum on a cart.

massless and rotate without friction. The parameters and constants are presented in Table (5.1). Note that x, u and θ depends on time, t. This will often be omitted from now on for convenience. The total energy in the system E_{tot} is the sum of the potential and the kinetic energy which are given by the equations

$$E_k = \frac{1}{2}M\dot{x}^2 + \frac{m}{2}\left(\left(\dot{x} + \ell\dot{\theta}\cos\theta\right)^2 + \left(\ell\dot{\theta}\sin\theta\right)^2\right), \tag{5.3}$$

$$E_p = mg\ell\cos\theta. \tag{5.4}$$

To obtain the equation of motions for the system we introduce the Lagrange function

$$L = E_k - E_p. \tag{5.5}$$

The Lagrange function for the pendulum on a cart has the properties

$$\frac{\mathrm{d}}{\mathrm{d}t}\left(\frac{\partial L}{\partial \dot{x}}\right) - \frac{\partial L}{\partial x} = u, \tag{5.6}$$

$$\frac{\mathrm{d}}{\mathrm{d}t}\left(\frac{\partial L}{\partial \dot{\theta}}\right) - \frac{\partial L}{\partial \theta} = 0, \tag{5.7}$$

where $\frac{\mathrm{d}}{\mathrm{d}t}$ is the full derivative with respect to t and $\frac{\partial}{\partial x}$ is the partial derivative with respect to x. For a deeper explanation of Lagrangian dynamics we refer to [31]. From the above equations we obtain

$$(M+m)\ddot{x} + m\ell\ddot{\theta}\cos\theta - m\ell\dot{\theta}^2\sin\theta = u, \tag{5.8}$$

$$\ell\ddot{\theta} + \ddot{x}\cos\theta - g\sin\theta = 0. \tag{5.9}$$

We want to rewrite the system above as a system of first order differential equations. Let $X = (x,\, \dot{x},\, \theta,\, \dot{\theta})^T = (x_1,\, x_2,\, x_3,\, x_4)^T$. This yields

$$\dot{X} = f(X,u) = \begin{pmatrix} x_2 \\ \frac{-mg\sin x_3\cos x_3 + m\ell x_4^2\sin x_3 + u}{M + m\sin^2 x_3} \\ x_4 \\ \frac{(M+m)g\sin x_3 - m\ell x_4^2\sin x_3\cos x_3 - u\cos x_3}{\ell(M + m\sin^2 x_3)} \end{pmatrix}, \tag{5.10}$$

which is the so called state equation of the dynamical system.

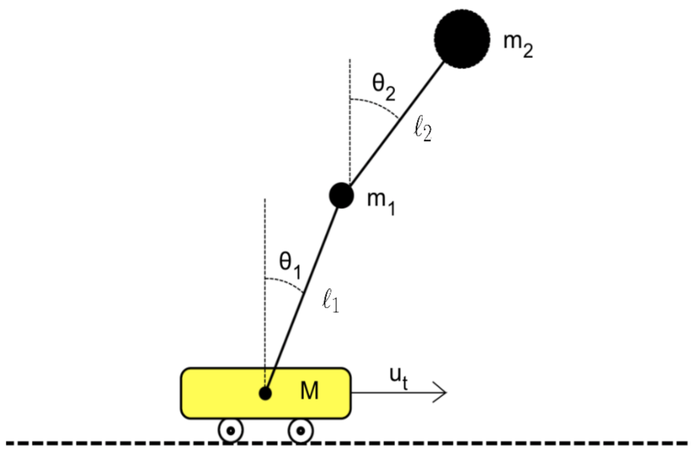


Figure 5.2: Double inverted pendulum on a cart.

5.2 Double inverted pendulum on a cart

The double inverted pendulum is best understood by looking at Figure 5.2. The only difference compared to the single inverted pendulum is that there are two pendulums that both can rotate frictionless around its base. We use the same notation as in the previous section with additional subscripts to distinguish the two pendulums. The equation of motions are again derived from the kinetic and potential energy. To make the derivation easier to follow the kinetic and potential energies are divided into three components corresponding to the cart, the first pendulum and the second pendulum. The total kinetic and potential energies can then be written as

$$\begin{aligned} E_k &= E_k^0 + E_k^1 + E_k^2, \\ E_p &= E_p^0 + E_p^1 + E_p^2, \end{aligned}$$

where

$$\begin{aligned} E_k^0 &= \frac{1}{2}M\dot{x}^2, \\ E_k^1 &= \frac{1}{2}m_1\left((\dot{x}+\ell_1\dot{\theta}_1\cos\theta_1)^2 + (\ell_1\dot{\theta}_1\sin\theta_1)^2\right), \\ E_k^2 &= \frac{1}{2}m_2\left((\dot{x}+\ell_1\dot{\theta}_1\cos\theta_1+\ell_2\dot{\theta}_2\cos\theta_2)^2 + (\ell_1\dot{\theta}_1\sin\theta_1+\ell_2\dot{\theta}_2\sin\theta_2)^2\right), \\ E_p^0 &= 0, \\ E_p^1 &= m_1 g\ell_1\cos\theta_1, \\ E_p^2 &= m_2 g(\ell_1\cos\theta_1+\ell_2\cos\theta_2). \end{aligned}$$

The Lagrange equation is again given by $L = E_k - E_p$ which becomes

$$\begin{aligned} L =& \frac{1}{2}(M+m_1+m_2)\dot{x}^2 + \frac{1}{2}(m_1+m_2)\ell_1^2\dot{\theta}_1^2 + \frac{1}{2}m_2\ell_2^2\dot{\theta}_2^2 + (m_1+m_2)\ell_1\dot{x}\dot{\theta}_1\cos\theta_1 \\ &+ m_2\ell_2\dot{x}\dot{\theta}_2\cos\theta_2 + m_2\ell_1\ell_2\dot{\theta}_1\dot{\theta}_2\cos(\theta_1-\theta_2) - (m_1+m_2)g\ell_1\cos\theta_1 - m_2 g\ell_2\cos\theta_2. \end{aligned}$$

Again, with the properties

$$\begin{aligned}\frac{\mathrm{d}}{\mathrm{d}t}\left(\frac{\partial L}{\partial \dot{x}}\right)-\frac{\partial L}{\partial x}=&u,\\ \frac{\mathrm{d}}{\mathrm{d}t}\left(\frac{\partial L}{\partial \dot{\theta}_1}\right)-\frac{\partial L}{\partial \theta_1}=&0,\\ \frac{\mathrm{d}}{\mathrm{d}t}\left(\frac{\partial L}{\partial \dot{\theta}_2}\right)-\frac{\partial L}{\partial \theta_2}=&0.\end{aligned}$$

After writing the above explicitly we obtain

$$\begin{aligned}u =&(M+m_1+m_2)\ddot{x}+(m_1+m_2)\ell_1(\ddot{\theta}_1\cos\theta_1-\dot{\theta}_1^2\sin\theta_1)\\ &+m_2\ell_2(\ddot{\theta}_2\cos\theta_2-\dot{\theta}_2^2\sin\theta_2),\end{aligned} \tag{5.11}$$

$$\begin{aligned}0 =&(m_1+m_2)\ell_1\ddot{x}\cos\theta_1+(m_1+m_2)\ell_1^2\ddot{\theta}_1+m_2\ell_1\ell_2\ddot{\theta}_2\cos(\theta_1-\theta_2)\\ &+m_2\ell_1\ell_2\dot{\theta}_2^2\sin(\theta_1-\theta_2)-(m_1+m_2)\ell_1 g\sin\theta_1,\end{aligned} \tag{5.12}$$

$$\begin{aligned}0 =&m_2\ell_2\ddot{x}\cos\theta_2+m_2\ell_1\ell_2\ddot{\theta}_1\cos(\theta_1-\theta_2)+m_2\ell_2^2\ddot{\theta}_2\\ &-m_2\ell_1\ell_2\dot{\theta}_1^2\sin(\theta_1-\theta_2)-m_2 g\ell_2\sin\theta_2.\end{aligned} \tag{5.13}$$

The Lagrangian equations can be written explicitly in the same way as (5.8) and (5.9) but the expression is rather messy. The following compact notation is borrowed from [27]. Let $X=(x,\,\theta_1,\,\theta_2,\,\dot{x},\dot{\theta}_1,\,\dot{\theta}_2)^T=(x_1,\,x_2,\,x_3,\,x_4,\,x_5,\,x_6)^T$. Equations (5.11)-(5.13) can then be written as

$$\boldsymbol{D}(X)(\dot{x}_3,\,\dot{x}_4,\,\dot{x}_5)^T+\boldsymbol{C}(X)(x_4,\,x_5,\,x_6)^T+\boldsymbol{G}(X)=\boldsymbol{H}u, \tag{5.14}$$

with

$$\begin{aligned}\boldsymbol{D}(X)&=\begin{pmatrix}d_1 & d_2\cos x_2 & d_3\cos x_3\\ d_2\cos x_2 & d_4 & d_5\cos(x_2-x_3)\\ d_3\cos x_3 & d_5\cos(x_2-x_3) & d_6\end{pmatrix},\\ \boldsymbol{C}(X)&=\begin{pmatrix}0 & -d_2x_5\sin x_2 & -d_3x_6\sin x_3\\ 0 & 0 & d_5x_6\sin(x_2-x_3)\\ 0 & -d_5x_5\sin(x_2-x_3) & 0\end{pmatrix},\\ \boldsymbol{G}(X)&=\begin{pmatrix}0\\ -f_1\sin x_2\\ -f_2\sin x_3\end{pmatrix},\\ \boldsymbol{H}&=\begin{pmatrix}1\\0\\0\end{pmatrix},\end{aligned}$$

where

$$\begin{aligned}
d_1 &= M + m_1 + m_2,\\
d_2 &= (m_1 + m_2)\ell_1,\\
d_3 &= m_2\ell_2,\\
d_4 &= (m_1 + m_2)\ell_1^2,\\
d_5 &= m_2\ell_1\ell_2,\\
d_6 &= m_2\ell_2^2,\\
f_1 &= (m_1 + m_2)\ell_1 g,\\
f_2 &= m_2\ell_2 g.
\end{aligned}$$

Note that D is a non-singular, symmetric matrix. Equation (5.14) can then be written as the six dimensional ODE

$$\dot{X} = f(X,\, u) = \begin{pmatrix} \mathbf{0} & \boldsymbol{I} \\ \mathbf{0} & -\boldsymbol{D}^{-1}\boldsymbol{C} \end{pmatrix} X + \begin{pmatrix} \mathbf{0} \\ -\boldsymbol{D}^{-1}\boldsymbol{G} \end{pmatrix} + \begin{pmatrix} \mathbf{0} \\ -\boldsymbol{D}^{-1}\boldsymbol{H} \end{pmatrix} u, \tag{5.15}$$

were $\mathbf{0}$ and $\boldsymbol{I}$ are the zero matrix and the identity matrix in appropriate dimensions respectively.

5.3 Numerical methods for simulations of inverted pendulum on a cart

In this thesis four different approaches are used to solve a stochastic optimal control problem of the type described in Chapter 3; the naive controller, the DPE controller, the FBSDE controller and the deep BSDE solver (the deep BSDE solver is only used in the LQR problem introduced in Section 3.6.2). Despite the differences all approaches use neural networks with the current state (solution to the state equation at the current time) as input to approximate the optimal action at that time. Therefore it is crucial to be able to approximate (5.2) accurately.

The dynamics of an inverted pendulum on a cart without control is very sensitive to initial conditions, *i.e.* a small change in the initial value can cause a large change in the future. When a sufficiently good control[1] is added to the system it is less chaotic since the control pushes the state equation towards some desired final state. Therefore, we only consider performance of numerical methods for the uncontrolled state equations, *i.e.* equation (5.2) for single and double inverted pendulum with $u_t = 0$ for all $t \in [0,\, T]$. We now present the numerical method for solving SDEs and define in what sense it converges to the exact solution. The goal is to approximate an SDE with the following settings. Given an initial condition $X_0 = x \in \mathbb{R}^n$, a non singular matrix $\sigma \in \mathbb{R}^{n\times n}$, $f\colon [0,\, T] \times \mathbb{R}^n \to \mathbb{R}^n$ and the Wiener process $W = \{W_t\}_{t\in[0,\,T]} \in \mathbb{R}^n$ the stochastic process $X = \{X_t\}_{t\in[0,\,T]}$ is described by

$$\mathrm{d}X_t = f(X_t)\mathrm{d}t + \sigma \mathrm{d}W_t,\ t \in (0,\, T]; \quad X_0 = x. \tag{5.16}$$

[1]In this context a sufficiently good control refers to a control that manage to stabilize the pendulum in a possibly non optimal way.

We then consider an equidistant discretization of the time interval [0, T]

$$0 = t_0 < t_1 < \cdots < t_N = T, \tag{5.17}$$

with step size $\Delta t = \frac{T}{N}$. With this settings and for $k = 0, 1, \ldots, N-1$ the ***Euler–Maruyama*** method for approximating equations on the form (5.16) is given by the iterative scheme

$$X_{k+1} = X_k + f(X_k)\Delta t + \sigma(W_{k+1} - W_k); \quad X_0 = x_0. \tag{5.18}$$

where X_k is the approximation of X_{t_k} and $W_k = W_{t_k}$. Note that $W_{k+1} - W_k$ is normally distributed with zero mean and variance h. A natural question to ask is then when, and in what sense (5.18) converges to the solution of (5.16). One way of measuring the error at time t is with the Euclidean norm $|X(t_n) - X_n|$. If we consider the last time step and if there exists a $C < \infty$ independent of N satisfying

$$\|X_T - X_N\|_{\mathbb{L}^2_T(\mathbb{R}^n)} \leq CN^{-q}, \tag{5.19}$$

we then say that X_N converges strongly to X_T with convergence order q. From standard theory of numerical methods for SDEs, *e.g.*, [32],it holds that if the initial value x_0 has finite second moment and the drift and diffusion coefficients satisfy a global Lipschitz condition, then the Euler–Maruyama approximation (5.18) for SDEs of the form (5.16) has strong convergence order $q = 0.5$. Recall from Section 2.2 that, among other assumption we assumed Lipschitz continuity of the drift coefficient for existence and uniqueness of a SDE. In the case with the inverted pendulums on a cart the SDEs do not have Lipschitz continuous drift coefficients and the theoretical results do not apply. This does not mean that a solution does not exists but only that it is not verified theoretically whether it has or not.. If a solution in fact exists then the Euler–Maruyama approximation may or may not converge. Since the theoretical results do not apply to our specific problem the best we can do is to try to establish an empirical convergence. This is done by replacing X_T by another Euler–Maruyama approximation on a much finer grid, *i.e.*, for $\tilde{N} >> N$ the interval [0, T] is divided into the equidistant grid

$$0 = t_0 < t_1 < \cdots < t_{\tilde{N}} = T. \tag{5.20}$$

We then use the Monte–Carlo approximation of $\|X_T - X_N\|_{\mathbb{L}^2_T(\mathbb{R}^n)}$ defined by

$$\frac{1}{M_{\text{samples}}} \sum_{i=1}^{M_{\text{samples}}} |X^i_{\tilde{N}} - X^i_N| \approx \|X_T - X_N\|_{\mathbb{L}^2_T(\mathbb{R}^n)}, \tag{5.21}$$

where superscript i separates different realizations of (5.18). The empirical convergence rate is then the smallest constant $\tilde{q}$ such that there exists a constant $C < \infty$ such that

$$\frac{1}{M_{\text{samples}}} \sum_{i=1}^{M_{\text{samples}}} |X^i_{\tilde{N}} - X^i_N| \leq CN^{-\tilde{q}}. \tag{5.22}$$

5.4 Numerical experiments for simulations of inverted pendulums on a cart

In this section we simulate the single and double inverted pendulum on a cart (without control signal) with the Euler–Maruyama scheme (5.18). We evaluate some critical aspects of the performance of the method and present the results in plots. Empirical convergence rates for starting points close to stable and unstable stationary points are established both for the single and double inverted pendulum. For the double inverted pendulum, which is sensitive to the length of the time interval T, we also investigate the dependence of T for the simulation error.

5.4.1 Single inverted pendulum

For the single inverted pendulum we consider the parameters $T = 2,\ M = 2,\ m = 1$ and $\ell = 1$. We further let $\sigma = \mathrm{diag}(s, s, s, s)$ for $s = 0.05$ and $s = 5$, where $\mathrm{diag}(s_1, \ldots, s_n)$ is a $n \times n$ diagonal matrix with $(s_1, \ldots, s_n)$ on the diagonal. The reason for testing different levels of noise is that one strategy for making the control more robust is to increase the noise of the state equation. We use initial data in the neighborhood to the two stationary points (stationary in the sense that uncontrolled, deterministic dynamics are stationary). They are $x_0^1 = (x,\, \dot{x},\, \theta,\, \dot{\theta})^T = (0,\, 0,\, 0.1,\, 0)^T$ and $x_0^2 = (0,\, 0,\, \pi - 0.1,\, 0)^T$.

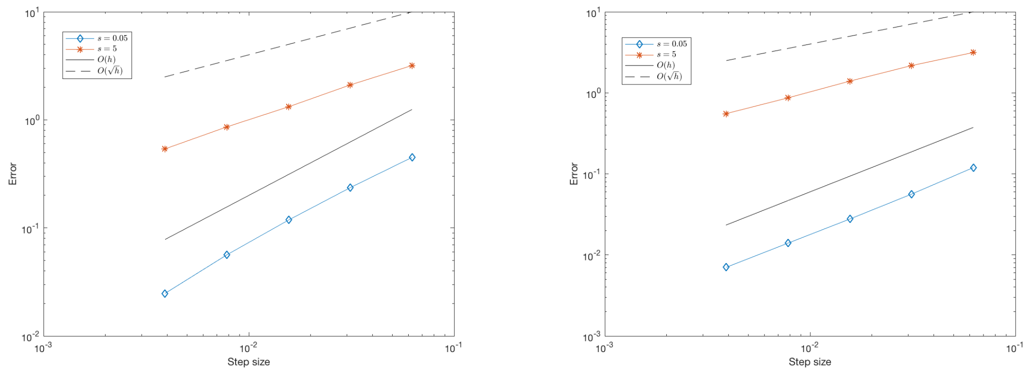


Figure 5.3: Convergence plots with $M_{\text{samples}} = 10$ for the uncontrolled state equation approximated with the Euler–Maruyama method for single pendulum. Left: Initial data x_0^1. Right: Initial data x_0^2.

As we can see in Figure 5.3, for both initial values x_0^1 and x_0^2, we have an empirical convergence order $\tilde{q} = 1$ for $s = 0.05$ and $\tilde{q} = 0.5$ for $s = 5$. Note that even though f is not Lipschitz continuous we obtain the empirical convergence order 0.5 for the case of high noise. Regarding the case with lower noise we first note that for an ODE of the type $\dot{x} = f(t,\, x)$ we can, under similar conditions as for the SDE, prove a theoretical convergence order 1 for the forward Euler scheme (which is the Euler–Maruyama scheme for deterministic equations). With this in mind it is not surprising that the empirical convergence order obtained in the case with $s = 0.05$ is the same as in the deterministic case since 0.05 is relatively close to zero.

5.4.2 Double inverted pendulum

For the double inverted pendulum we consider the parameters $T = 0.4$, $M = 1$, $m_1 = 0.5$, $m_2 = 0.75$, $\ell_1 = 0.5$ and $\ell_2 = 0.75$. We further let $\sigma = \text{diag}(s, s, s, s, s, s)$ for $s = 0.05$ and $s = 0.3$. We use initial data close to two of the four equilibrium points. They are given by $x_0^1 = (x,\, \theta_1,\, \theta_2,\, \dot{x},\, \dot{\theta}_1,\, \dot{\theta}_2)^T = (0,\, 0.1,\, 0.1,\, 0,\, 0,\, 0)^T$ and $x_0^2 = (0,\, \pi - 0.1,\, \pi - 0.1,\, 0,\, 0,\, 0)^T$. These are the equilibrium points where the system is in rest with both pendulums standing in upright and downright position, respectively. The reason for the relatively small $T = 0.4$ is that the simulation of the dynamics (with the Euler–Maruyama scheme) of the double inverted pendulum is very unstable and sensitive to initial conditions. This makes it almost impossible to perform simulations of the state equation for large T with this scheme. Another observation when the noise level increases the stability of the approximation scheme decreases. For this reason we have not considered noise levels above $s = 0.3$ for the double inverted pendulum.

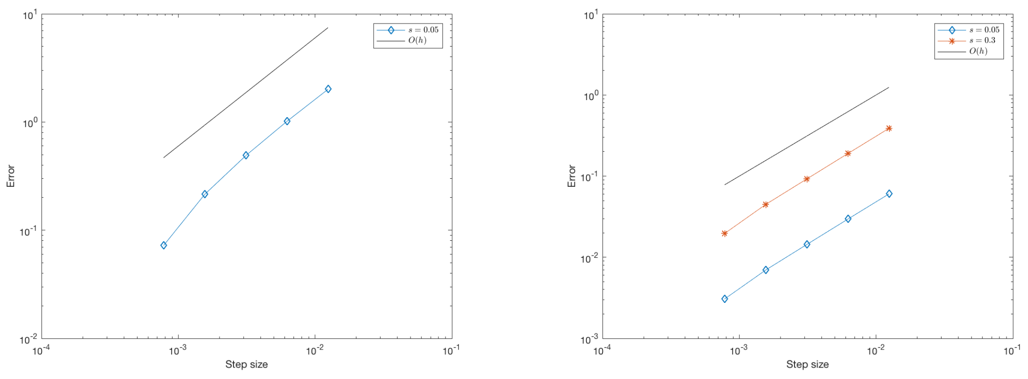


Figure 5.4: Convergence plots with $M_{\text{samples}} = 10$ for the uncontrolled state equation approximated with the Euler–Maruyama method for double pendulum. Left: Initial data x_0^1. Right: Initial data x_0^2.

From Figure 5.4 (to the right) we can conclude that for initial data x_0^2, close to the downright stationary point, we have empirical convergence order $\tilde{q} = 1$ for the both noise level $s = 0.05$ and $s = 0.3$. For the initial condition x_0^1, *i.e.* when both pendulums starts close to a upright position the situations is more complicated. For the case with $s = 0.05$ we again obtain convergence order $\tilde{q} = 1$, which is shown in Figure 5.4 (to the left). By comparing the plots in Figure 5.4 one notices that the error is a factor 10 larger (for the same step size and with noise level $s = 0.05$) in the plot to the left. This makes intuitive sense since the values of the state equation are higher when a double pendulum is released from a position close to upright for both pendulums than from a position close to downright for both pendulums. This problem can be resolved by decreasing the step step size with a longer computational time as a result. Another problem that does not have a simple solution is that the Euler–Maruyama is unstable for the double inverted pendulum for long time intervals, meaning that the approximation grows big even though the solution, or to be precise, the approximation computed on a finer grid, does not. The convergence for the case with initial value x_0^2 and noise level $s = 0.3$ cannot be found for $T = 0.4$ which is why that plot is missing Figure 5.4 (to the right). In Figure 5.5 the error is plotted against the length of the time interval T, for the two initial values x_0^1 and x_0^2, respectively. Note that since the range of different T in the plots are rather small the step size is approximately the same. We see that for the initial value close to the downright position (to the left) the error stays relatively small when T increases but for the initial value close to the upright position (to the right) the error explodes when T exceeds a certain threshold, close to $T = 0.45$.

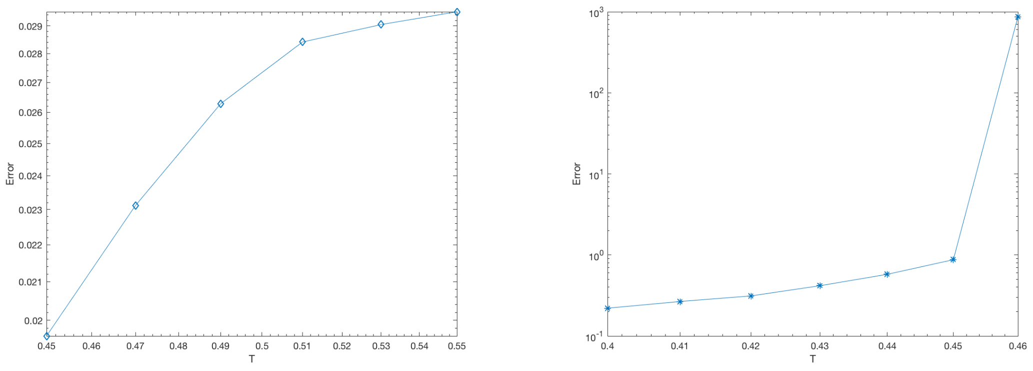


Figure 5.5: The error as a function of the length of the time interval T for noise level $s = 0.05$ with initial data x_0^1 to the left and x_0^2 to the right.

6
Numerical experiments

All numerical experiments in this chapter have been carried out in Python. For the training procedure of the neural networks we have used Google's open source machine learning framework TensorFlow. If nothing else is stated, default parameters are used.

Throughout this chapter we consider cost functions on the form

$$J(t,\, x\,;\, u) = \mathbb{E}\left[\int_t^T \left(|RX_s|^2 + r^2 u_s^2\right) \mathrm{d}s + |GX_T|^2 \,\middle|\, X_t = x\right], \tag{6.1}$$

where $R,\, G \in \mathbb{R}^{n\times n}$, $r \in \mathbb{R}$, $X_t \in \mathbb{R}^n$ and $u_t \in \mathbb{R}$ for $t \in [0,\, T]$. The matrices R and G are diagonal, and they are referred to as the running and terminal cost matrix respectively and the scalar r is referred to as the control cost. We refer repeatedly to the ***empirical confidence interval*** in this chapter. These are represented by shaded areas in figures below. The interpretation of these is that at all time points, 95% of the trajectories lie within the shaded area with 2.5% lie above and below the shaded area respectively. Throughout this Chapter batch size $K = 1$ is used.

6.1 Examples in one dimension

In this section we use the algorithms from Section 4 to control two different SDEs in one space dimension. The first example is the LQR problem, described in Section 3.6.2 and the second example is the control of a SDE with nonlinear drift coefficient. Recall from Section 3.6.2.2 that, given a state $(t,\, x)$, the optimal Markov control policy for the LQR problem is given by

$$\pi^*(t,\, x) = -\frac{b}{2r^2}(2P(t)x + Q(t)), \tag{6.2}$$

where P and Q solve the corresponding ODEs in (3.51). It is reasonable to believe that the network can learn this affine relationship. For the same reason the BSDE controller is a good choice in this case since there is no need for training in the right spatial domain where the approximation will be used. If the affine relationship is learned somewhere in space it has a good chance to extend to the whole space, depending only on the possible nonlinearity of the network. An affine function approximator (which is a neural network with no hidden layer and activation function being the identity) would of course be best but here we want to pretend we do not know this additional structure of the problem. We remark here that the LQR problem is easily solved and implemented using numerical approximations of P and Q without using neural networks. This gives us a benchmark solution for this specific problem.

In the second example we do not have the affine relationship between the current state and the feedback control. Therefore we use a coupled algorithm to ensure that the network is trained in the region in space where the optimally controlled dynamics will be. In one space dimension the DPE controller is the same algorithm as the naive controller.

6.1.1 Linear quadratic regulator

In this section we consider the Ornstein-Uhlenbeck process in one dimension, being the solution X to (3.39). We use the BSDE controller described in Section 4.2.4 to approximate the feedback control. The dynamics of the controlled process are given by equation (3.40). As described in Section 3.6.2.2 LQR problems can be solved analytically[1]. We can therefore compare the analytic and approximate solutions. We let $T = 1$, $N = 100$, $a = 1$, $b = 0.5$, $c = 0.2$, $\sigma = 0.05$, $R = 10$, $G = 1$, $r = 1$ and $x_0 \sim U([0.08, 0.32])$, where $U(I)$ denotes the uniform distribution on $I \subset \mathbb{R}$. We use N parallel neural networks to approximate Z_k for $k = 0, 1, \ldots, N-1$ and one subnetwork to approximate Y_0, which gives a total of $N+1$ subnetworks. Each subnetwork has 2 hidden layers with 5 nodes (neurons) and both the input and the output are 1-dimensional. This means that each subnetwork consists of 25 weights and 11 biases (free parameters) to be optimized. For $N = 100$ we then have 3636 free parameters. The upper plots in Figure 6.1 show two sample paths of uncontrolled processes, BSDE controlled processes and analytically controlled processes, all with the same noise. The corresponding control signals are compared in the lower plots in Figure 6.1. Figure 6.2 shows the average of 500 controlled and uncontrolled processes and the average of their corresponding control signals. The shaded areas represents an empirical 95% confidence intervals. This means that 95% of the training has been performed inside the gray shaded area in Figure 6.2 to the left. We can therefore, with high certainty conclude that the network is able to learn the optimal Markov control policy in this area.

6.1.2 Non-linear control

In this section we consider a state equation with a nonlinear drift given by

$$\mathrm{d}\bar{X}_t = a \sin\left(\frac{\pi \bar{X}_t}{c}\right) \mathrm{d}t + \sigma\, \mathrm{d}W_t,\ t \in (0, T]; \quad \bar{X}_0 = x_0, \tag{6.3}$$

for $a \in \mathbb{R}$, $c \in \mathbb{R} \setminus \{0\}$ and $\sigma \in \mathbb{R}^+$. This process has stable stationary points at odd multiples of c and unstable stationary points at even multiples of c. Our aim is to control the process around 0 by adding a feedback control in the drift term of

[1]Strictly speaking it is an approximation since we have to approximate the Riccati equation (3.51). We use a simple explicit Euler scheme to approximate (3.51) with $10N$ grid points.

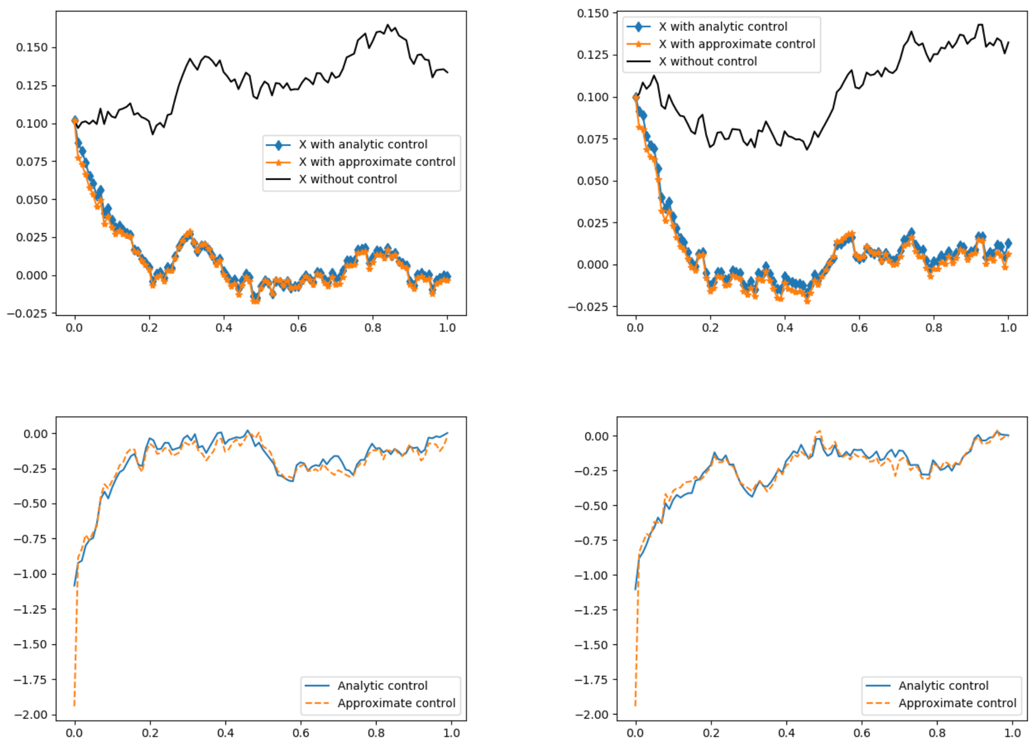


Figure 6.1: Upper: The BSDE control and the analytic control applied to two sample paths of an Ornstein-Uhlenbeck process. Lower: The BSDE controller and the analytic optimal control for two different sample paths of an Ornstein-Uhlenbeck process.

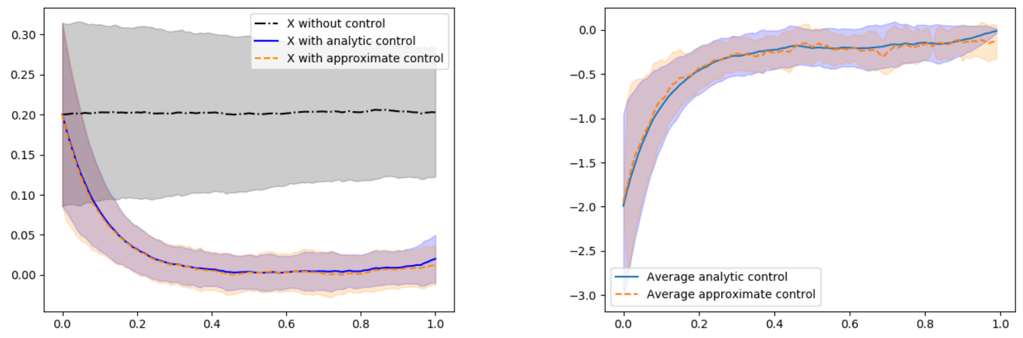


Figure 6.2: Left: The average of 500 test samples of controlled and uncontrolled Ornstein-Uhlenbeck processes with an empirical 95% confidence interval. Right: The average of 500 sample paths of the control signals that corresponds to the controlled processes in the plot to the right.

(6.3). For $b \in \mathbb{R}^+$ and $u_t \in \mathbb{R}$ for $t \in [0, T)$, the controlled state equation is given by

$$\mathrm{d}X_t = \left(a \sin\left(\frac{\pi X_t}{c}\right) + b\, u_t\right) \mathrm{d}t + \sigma\, \mathrm{d}W_t,\ t \in (0, T]; \quad X_0 = x_0. \tag{6.4}$$

We let $T = 0.5$, $N = 50$, $a = 1$, $b = 1$, $c = 0.2$, $\sigma = 0.05$, $R = 10$, $G = 1$, $r = 1$ and $x_0 \sim U([-0.22, 0.22])$. The upper plots in Figure 6.3 shows the average of 500 test samples of controlled and uncontrolled processes, with the naive controller to the left and the FBSDE controller to the right. Note that we have separated the

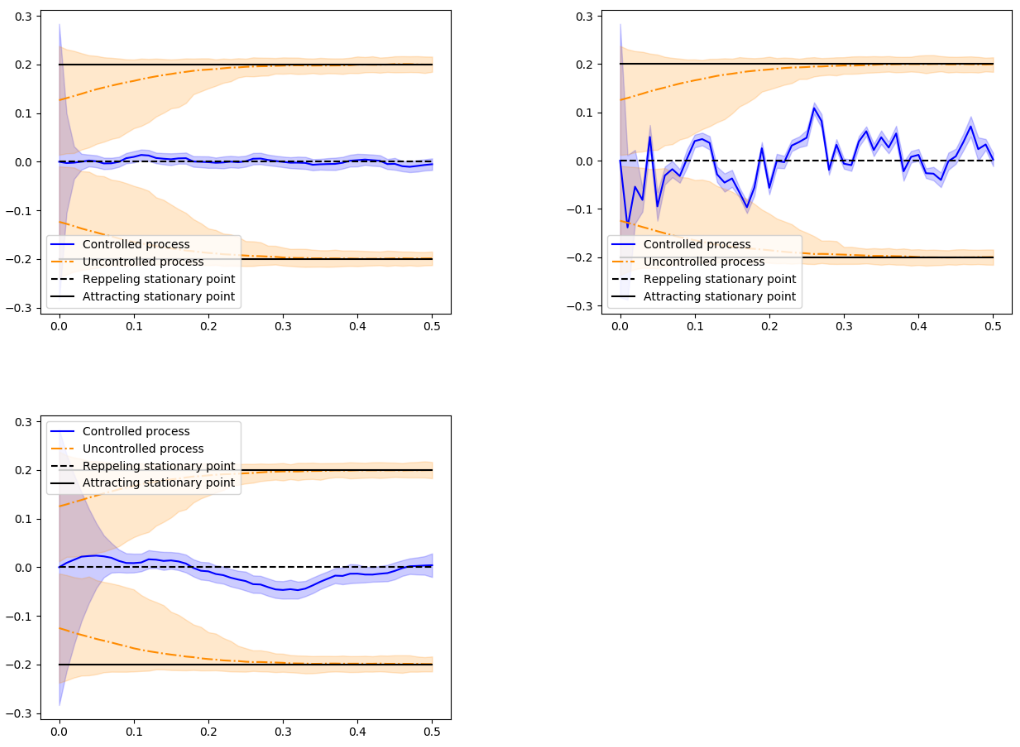


Figure 6.3: Average of 500 controlled and uncontrolled processes with a 95% empirical confidence intervals. Upper left: The naive controller. Upper right: The FBSDE controller. Lower: The FBSDE2 controller.

uncontrolled processes ending up in the upper half plane from those ending up in the lower half plane to make the plots more visual. We note that the process controlled by the FBSDE controller is oscillating. To avoid this we have also included a slightly different version of the FBSDE controller that counteracts oscillation in this section. This is done by adding a regulating term to the single batch loss function which for $\rho \in \mathbb{R}^+$ becomes

$$\overline{\text{loss}}^{\text{FBSDE2}}(\vartheta\,;\, x_0) = (g(X_N) - Y_N)^2 + \rho \sum_{k=0}^{N} (X_{k+1} - X_k)^2.$$

For all experiments in this section we use $\rho = 2.5$. The lower plot in Figure 6.3 shows the average of processes controlled by the FBSDE2 controller, which we for convenience denote the modified version of the FBSDE controller by. In Figure 6.4 sample paths of controlled processes are compared with their uncontrolled counterpart for the naive controller, the FBSDE controller and the FBSDE2 controller. Figure 6.5

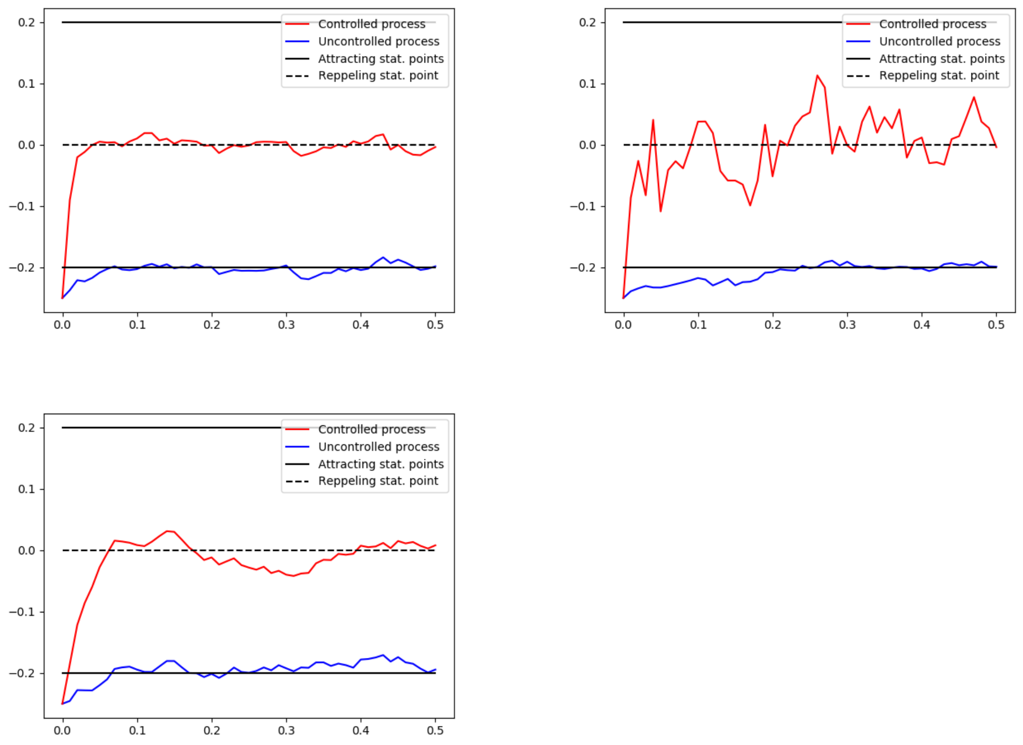


Figure 6.4: One sample of a controlled process and one sample of an uncontrolled process. Upper left: The naive controller. Upper right: The FBSDE controller. Lower: The FBSDE2 controller.

shows the average control signal for the three algorithms. For this problem, it is clear that the naive controller performs better than the FBSDE controllers. We further see that the oscillation decreases when the FBSDE2 controller is used but the performance is still not as good as with the naive controller.

6.2 Control of a single inverted pendulum on a cart

In this section we use the algorithms described in Section 4.2 to control a single inverted pendulum on a cart with the dynamics from Section 5.4.1. We compare the performance of the naive, DPE and FBSDE controllers by means of the cost during training. We further demonstrate that the FBSDE controller is able to swing up the pendulum from a position close to its downright position to the unstable upright position. This is done by showing single trajectories from controlled processes as well as the average of 500 controlled processes. In Section 6.2.1 we demonstrate that by changing the running and terminal cost matrices we can easily change the control strategy. Section 6.2.2 deals with an attempt to increase the robustness of

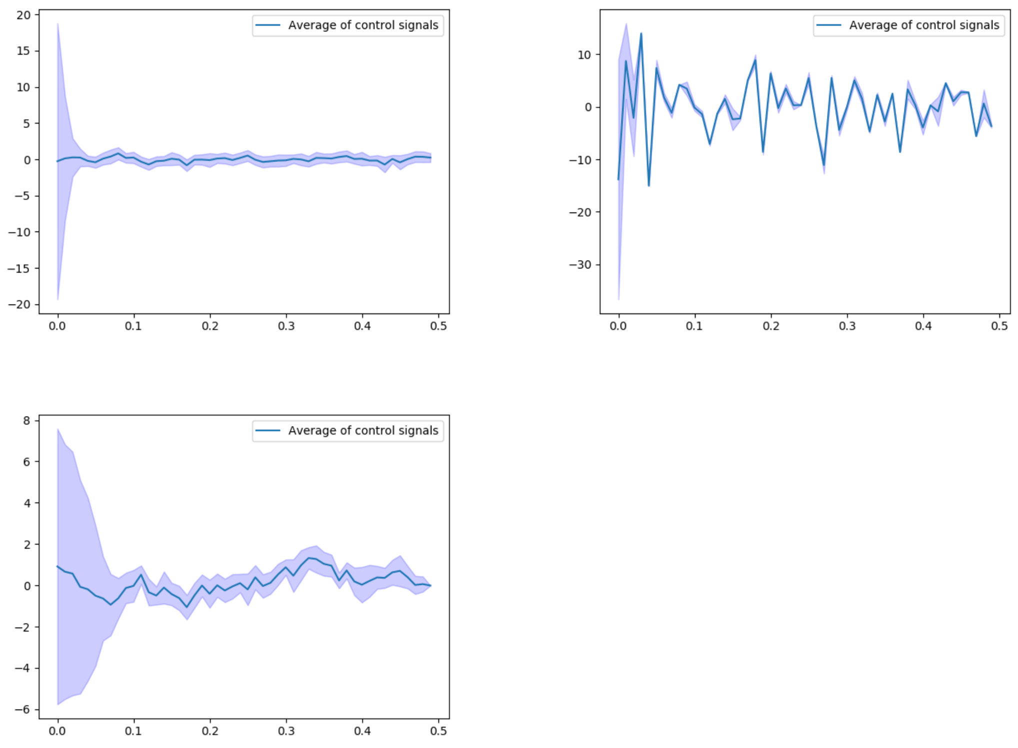


Figure 6.5: The average control signal with an empirical 95% confidence interval. Upper left: The naive controller. Upper right: The FBSDE controller. Lower: The FBSDE2 controller.

Single pendulum		
	Case I	Case II
N	100	100
M_{samples}	10000	20000
T	4	4
M	2	2
m	1	1
L	2	2
s	0.05	0.05
r	0.1	0.1
R_{11}	0	1
R_{22}	0	5
R_{33}	10	50
R_{44}	5	25
G_{11}	10	0
G_{22}	10	0
G_{33}	50	0
G_{44}	25	0
x_0	$U([0,\, 0])$	$U([0,\, 0])$
$\dot{x}_0$	$U([-0.5,\, 0.5])$	$U([-0.5,\, 0.5])$
θ_0	$U([\pi-0.5,\, \pi])$	$U([\pi-0.1,\, \pi])$
$\dot{\theta}_0$	$U([-0.5,\, 0.5])$	$U([-0.5,\, 0.5])$

Table 6.1: Parameters and constants for the single inverted pendulum on a cart used in the numerical experiments.

the control by increasing the noise level. The somewhat sloppy term 'robust control' means that the control strategy does not break down if the system is exposed to some perturbation, *e.g.*, if the pendulum is perturbed, causing a rapid change in the state. We also show that the FBSDE controller performs well over a large domain of initial values, *i.e.* for different initial velocities, angles and angular velocities.

6.2.1 Two different control strategies

Consider the two cases with values of the parameters as given in Table 6.1. We use N subnetworks to approximate Z_k for $k = 0, 1, \dots, N-1$ and one subnetwork to approximate Y_0, which gives a total of $N+1$ subnetworks. Each subnetwork has 2 hidden layers with 14 nodes and both the input and the output are 4-dimensional. This means that each subnetwork consists of 308 weights and 32 biases to be optimized. For $N = 100$ we then have 34340 free parameters. In both parameter settings the goal is to stabilize the pendulum around its upright equilibrium point. The difference lies in the running and terminal cost matrices. In Case I running cost is low and the terminal cost high. It results in a control strategy that is passive in the beginning and then quickly swings up the pendulum to the upright position close to the end of the time interval. In Case II the strategy is the opposite, thus controlling the pendulum as quick as possible to the upright position and then balancing it at this point for the remaining time. The computing times were approximately as follows, Case I: The naive controller 7000 seconds, the FBSDE and DPE controllers 8000 seconds. Case II: The naive controller 15000 seconds, the FBSDE and DPE controllers 17000 seconds. Note that no effort has been made to optimize the algorithm from a computing time perspective.

One problem when evaluating the algorithms is that we cannot determine how close the approximate controllers are to the optimal Markov control policy. What we can do, however, is to compare the naive controller, the DPE controller and the FBSDE controller by means of the cost during training. We also want to approximate the cost during training to make sure that the cost actually decreases. To do this we denote the i:th initial value and control by x_0^i and $u^i = \{u_k^i\}_{k\in\{0,1,\dots,N-1\}}$ respectively and let the step size $\Delta t = \frac{T}{N+1}$. We train 5 different networks with the same initial parameters and initial values. Then for $i = 1, 2, \dots, M_{\text{samples}}$ we use the formula

$$J(0,\, x_0^i,\, ;\, u^i) \approx \frac{1}{5}\sum_{j=1}^{5}\left(\sum_{k=0}^{N}(|RX_k^i|^2 + r^2(u_k^i)^2)\Delta t + |GX_N^i|^2\right). \tag{6.5}$$

In Figure 6.6 we see that in Case I, the cost is similar when using the FBSDE controller and the DPE controller and the cost is slightly higher when the naive controller is used. In Case II, on the other hand, we see a significant difference in the cost during training where the FBSDE and the DPE controllers have significantly lower cost than the naive controller. Note that the number of training samples, N, are chosen larger when the algorithms are evaluated in order to guarantee that the values of the cost have stabilized. After comparing the different algorithms, the

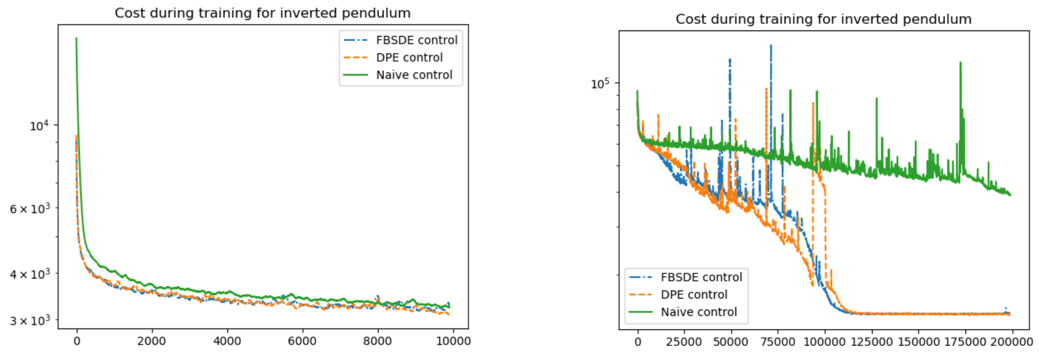


Figure 6.6: Cost plotted against the number of training iterations for different algorithms. Left: Case I. Right: Case II.

remaining part of this chapter is about the FBSDE controller. Figures 6.7 (Case I) and 6.8 (Case II) show the average of 500 controlled processes. In Figures 6.9 (Case I) and 6.10 (Case II) we choose at random one of the 500 samples and compare the controlled and the uncontrolled processes. Figure 6.11 shows the average control signal at each time point with a 95% empirical confidence interval. We have seen that the FBSDE controller is able to control the pendulum to an upright position in both Case I and Case II. To emphasize the difference in the different control

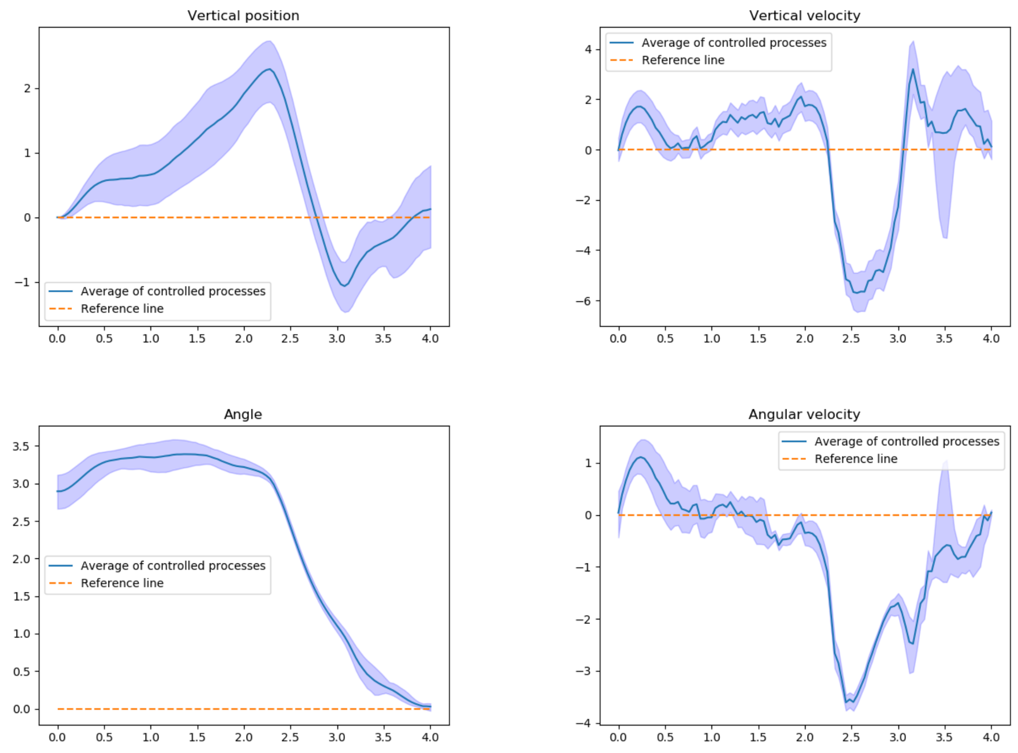


Figure 6.7: The average of 500 test samples of the uncontrolled and controlled processes with an empirical 95% confidence interval with settings as in Case I.

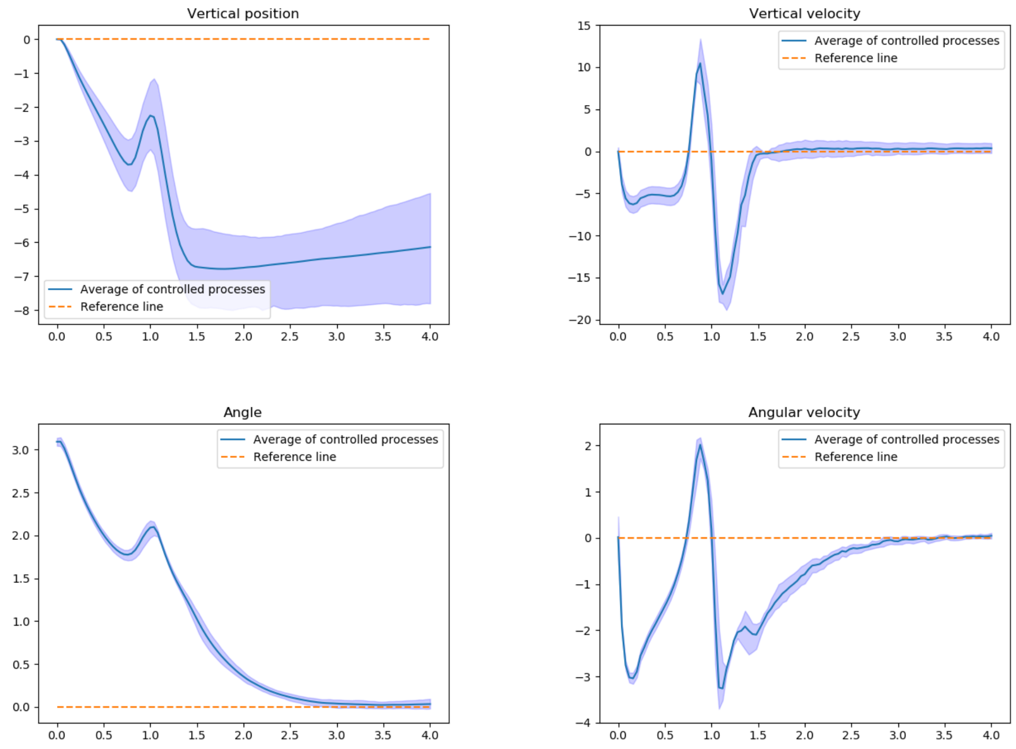


Figure 6.8: The average of 500 test samples of the uncontrolled and controlled processes with an empirical 95% confidence interval with settings as in Case II.

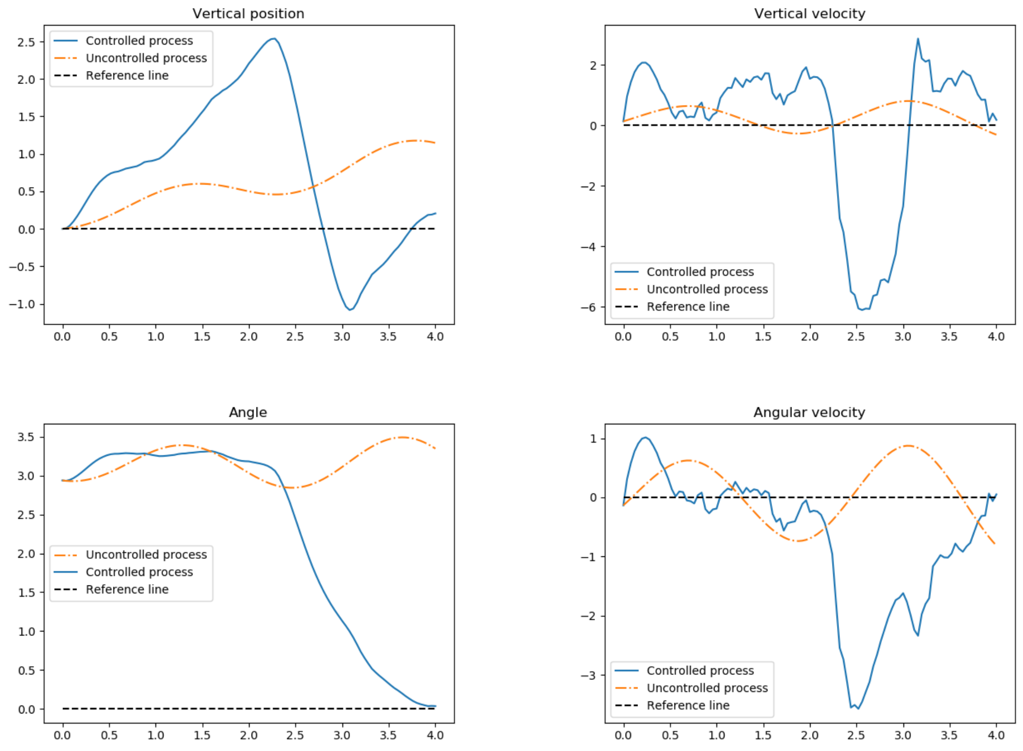


Figure 6.9: One typical, randomly chosen sample of a controlled and uncontrolled process with settings as in Case I.

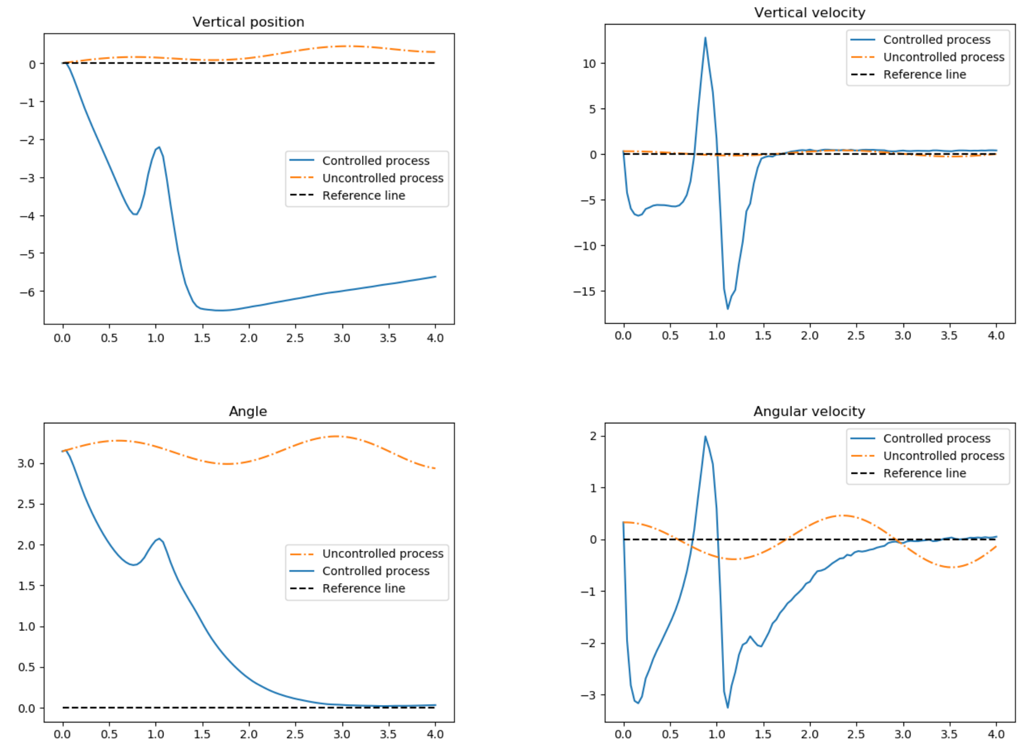


Figure 6.10: One typical, randomly chosen sample of a controlled and uncontrolled process with settings as in Case II.

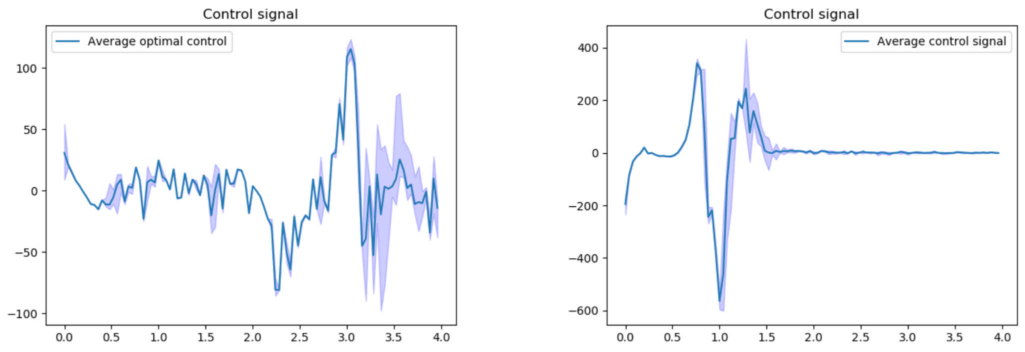
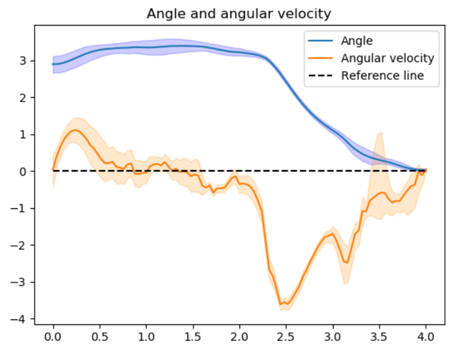


Figure 6.11: The average control signal with an empirical 95% confidence interval. Left: Case I. Right: Case II.

strategies the average angle and angular velocity for controlled processes in Case I and Case II are compared in Figure 6.12.



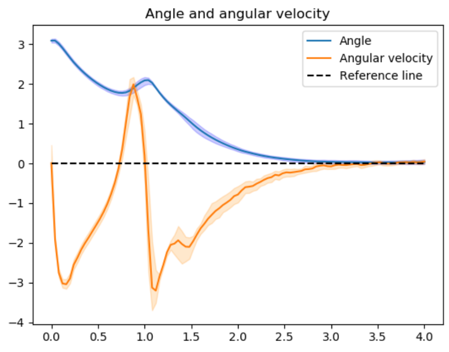


Figure 6.12: The average of 500 test samples of the angles and the angular velocities with an empirical 95% confidence interval. Left: Case I. Right: Case II.

6.2.2 Some strategies to improve the robustness of the control

As described in Section 4.2 the control signal at time t_k is given by the mapping $(t_k,\, X_k) \mapsto u_k$. A problem with our algorithms is that they are not reliable for (t, X) outside the domain in which the neural network is trained which in turn may lead to a less robust control. In an attempt to make the control more robust, the network was trained over a larger domain of initial values. We use the same settings as in Case I but with $\dot{x}_0 \sim U([-1,\, 1])$ and $\dot{\theta}_0 \sim U([-4,\, 2])$. In Figure 6.13 we see that the controller works for a large domain of different initial values but after $t = 2$ the shaded areas becomes narrower, indicating a less robust control for $t > 2$. By increasing the noise to $s = 0.3$ we obtain wider empirical confidence intervals, which is displayed in Figure 6.14. One problem with this procedure is that by increasing the noise, the state equation describing the pendulum becomes less realistic. An attempt to resolve this problem was done by using $s = 0.3$ when training the neural network and removing the noise, *i.e.* letting $s = 0$ when testing the algorithm. The corresponding plots are displayed in Figure 6.15. It should be emphasized that the optimal Markov Control policy for the two cases $s = 0$ and $s = 0.3$ are not the same. On the other hand, it is likely that the optimal Markov control policy for the problem with $s = 0.3$ works sufficiently well for the corresponding problem with $s = 0$.

6.3 Control of a double inverted pendulum on a cart

This section deals with numerical experiments of control of a double inverted pendulum on a cart. Recall that while a single pendulum has one unstable and one stable

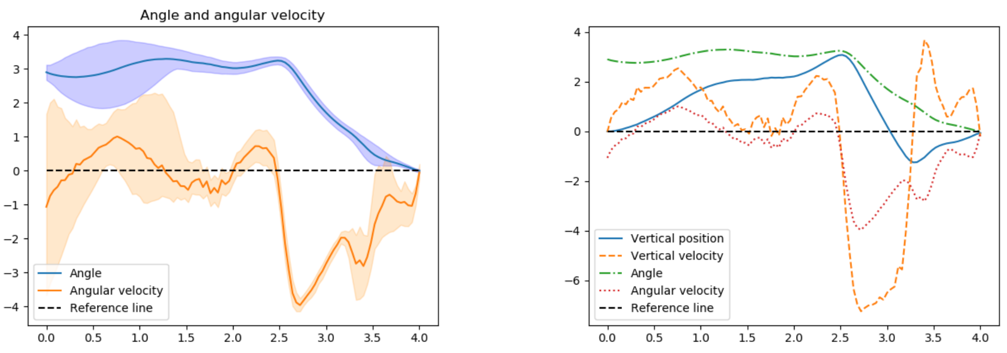


Figure 6.13: Settings as in Case I but with the larger domain of initial values. Left: The average of 500 test samples of angles and angular velocities with an empirical 95% confidence interval. Right: The average of 500 test samples.

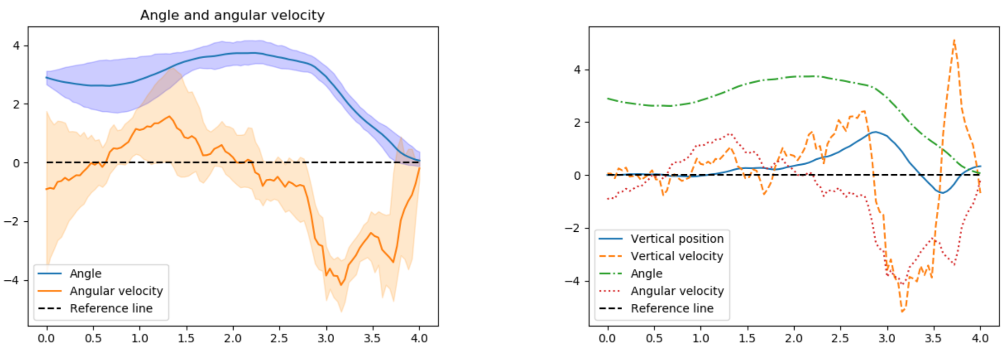


Figure 6.14: Settings as in Case I but with the larger domain of initial values and with $s = 0.3$. Left: The average of 500 test samples of angles and angular velocities with an empirical 95% confidence interval. Right: The average of 500 test samples.

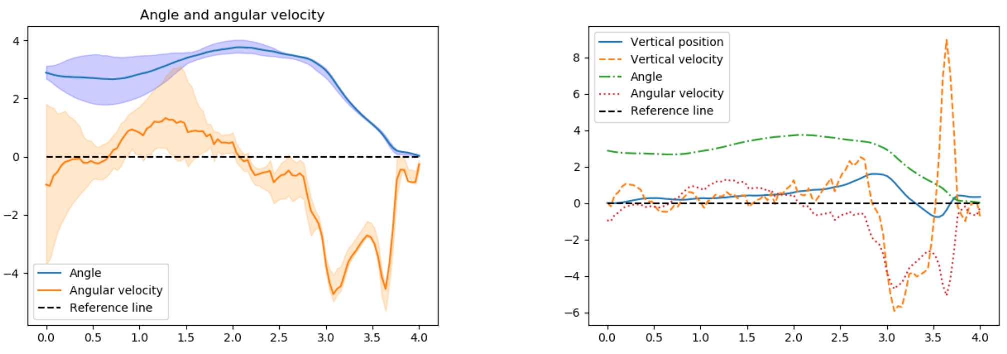


Figure 6.15: Settings as in Case I but with the larger domain of initial values. The algorithm is trained with $s = 0.3$ and tested with $s = 0$. Left: The average of 500 test samples of angles and angular velocities with an empirical 95% confidence interval. Right: The average of 500 test samples.

equilibrium point, a double pendulum has one stable and three unstable equilibrium points. In Section 6.3.1 the performances of the naive, DPE and FBSDE controllers by means of cost during training are compared. We further show that the FBSDE controller is able to stabilize the double pendulum around its upright equilibrium point for initial values close to any other equilibrium point. In Section 6.3.2 we use similar techniques to improve the robustness as for the single pendulum.

Recall the discussion in Section 5.4.2 about the numerical instability of the Euler-Maruyama scheme for approximations of the state equation of the double inverted pendulum. For that reason, we consider a shorter time interval than for the single inverted pendulum.

6.3.1 Different initial values

The structure of the FBSDE controller applied to the double pendulum is similar to the corresponding case with the single pendulum described in Section 6.2.1. One difference is that the dynamics is six dimensional instead of four. We also use 26 nodes in each hidden layer instead of 14. The choice of 26 is chosen by trial and error, although it is not surprising that a more complicated problem needs a larger neural network. For $N = 100$ we then have 105646 free parameters to be optimized. We consider five different cases in this section, with parameters given in Table 6.2. Case I - Case II only differ in the initial states and the reason for this is that the time interval is considerably shorter and different strategies are then difficult to apply. For the same reason, we have also prioritized to control the angles and the angular velocities in this section. The initial state in Case I is when both pendulums are close to the equilibrium point where both pendulums are pointing downwards. In Case II and Case III, the initial states are close to the equilibrium points where one pendulum is pointing downwards and the other one is pointing upwards. Case IV and Case V have higher noise in the diffusion term of the state equation and Case V also has a larger spatial domain of initial states. They are used to increase the robustness of the control and are further introduced in Section 6.3.2. Figure 6.16 shows a comparison of the performance of the naive controller, the DPE controller and the FBSDE controller by means of the cost (6.5) during training for Case I - Case V. We see that the naive controller has significantly higher cost than the two other algorithms in all cases except Case IV where the cost is similar. To determine definitely which one of the DPE controller and the FBSDE controller that performs best would need further investigation since the variance is too large with only 5 trained networks. As in the previous section after the different algorithms have been compared the continuation of this section deals exclusively with the FBSDE controller. After having compared the performance of the different algorithms we only consider the FBSDE controller from now on. Figure 6.17 show the average of 500 test samples of angles and angular velocities with empirical 95% confidence intervals for Case I - Case III. In Figure 6.18 a randomly chosen controlled process are displayed component wise and compared with its uncontrolled counterpart with settings as in Case I. From these figures we can conclude that the FBSDE controller is able to swing up the double inverted pendulum to its upright position with both

Double pendulum					
	Case I	Case II	Case III	Case IV	Case V
N	100	100	100	100	100
M_s	10000	10000	10000	10000	10000
T	0.5	0.5	0.5	0.5	0.5
M	1	1	1	1	1
m_1	0.5	0.5	0.5	0.5	0.5
m_2	0.75	0.75	0.75	0.75	0.75
l_1	0.5	0.5	0.5	0.5	0.5
l_2	0.75	0.75	0.75	0.75	0.75
s	0.05	0.05	0.05	0.3	0.3
r	0.1	0.1	0.1	0.1	0.1
R_{11}	10	10	10	10	10
R_{22}	25	25	25	25	25
R_{33}	25	25	25	25	25
R_{44}	1	1	1	1	1
R_{55}	1	1	1	1	1
R_{66}	1	1	1	1	1
G_{11}	5	5	5	5	5
G_{22}	150	150	150	150	150
G_{33}	150	150	150	150	150
G_{44}	5	5	5	5	5
G_{55}	25	25	25	25	25
G_{66}	25	25	25	25	25
x_0	$U([0,\ 0])$	$U([0,\ 0])$	$U([0,\ 0])$	$U([0,\ 0])$	$U([0,\ 0])$
$\dot{x}_0$	$U([-0.5, 0.5])$	$U([-0.5, 0.5])$	$U([-0.5, 0.5])$	$U([-0.5, 0.5])$	$U([-0.5, 0.5])$
θ_0^1	$U([\pi-0.1, \pi])$	$U([0, 0.1])$	$U([\pi-0.1, \pi])$	$U([\pi-0.1, \pi])$	$U([\frac{2\pi}{3}, \pi])$
$\dot{\theta}_0^1$	$U([-0.5, 0.5])$	$U([-0.5, 0.5])$	$U([-0.5, 0.5])$	$U([-0.5, 0.5])$	$U([-12, 7])$
θ_0^2	$U([\pi-0.1, \pi])$	$U([\pi-0.1, \pi])$	$U([0, 0.1])$	$U([\pi-0.1, \pi])$	$U([\frac{2\pi}{3}, \pi])$
$\dot{\theta}_0^2$	$U([-0.5, 0.5])$	$U([-0.5, 0.5])$	$U([-0.5, 0.5])$	$U([-0.5, 0.5])$	$U([-12, 7])$

Table 6.2: Parameters and constants for the double inverted pendulum on a cart used in the numerical experiments.

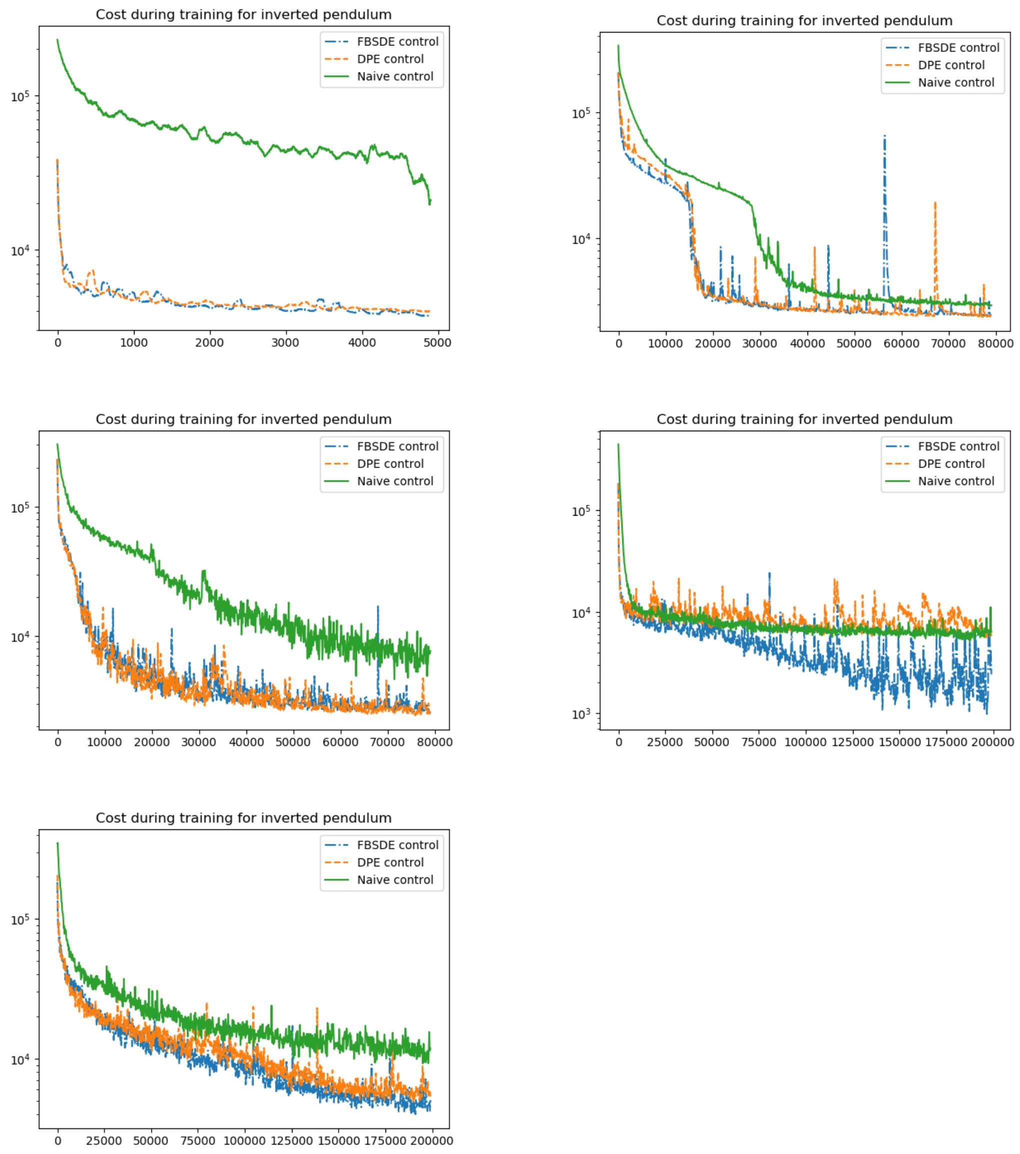


Figure 6.16: Cost plotted against the number of training iterations for different algorithms. Upper left: Case I. Upper right: Case II. Mid left: Case III. Mid right: Case IV. Lower: Case V

angular velocities close to zero. Figure 6.19 shows the cost during training and the average control against time. In Figure 6.18 we choose at random one of the 500

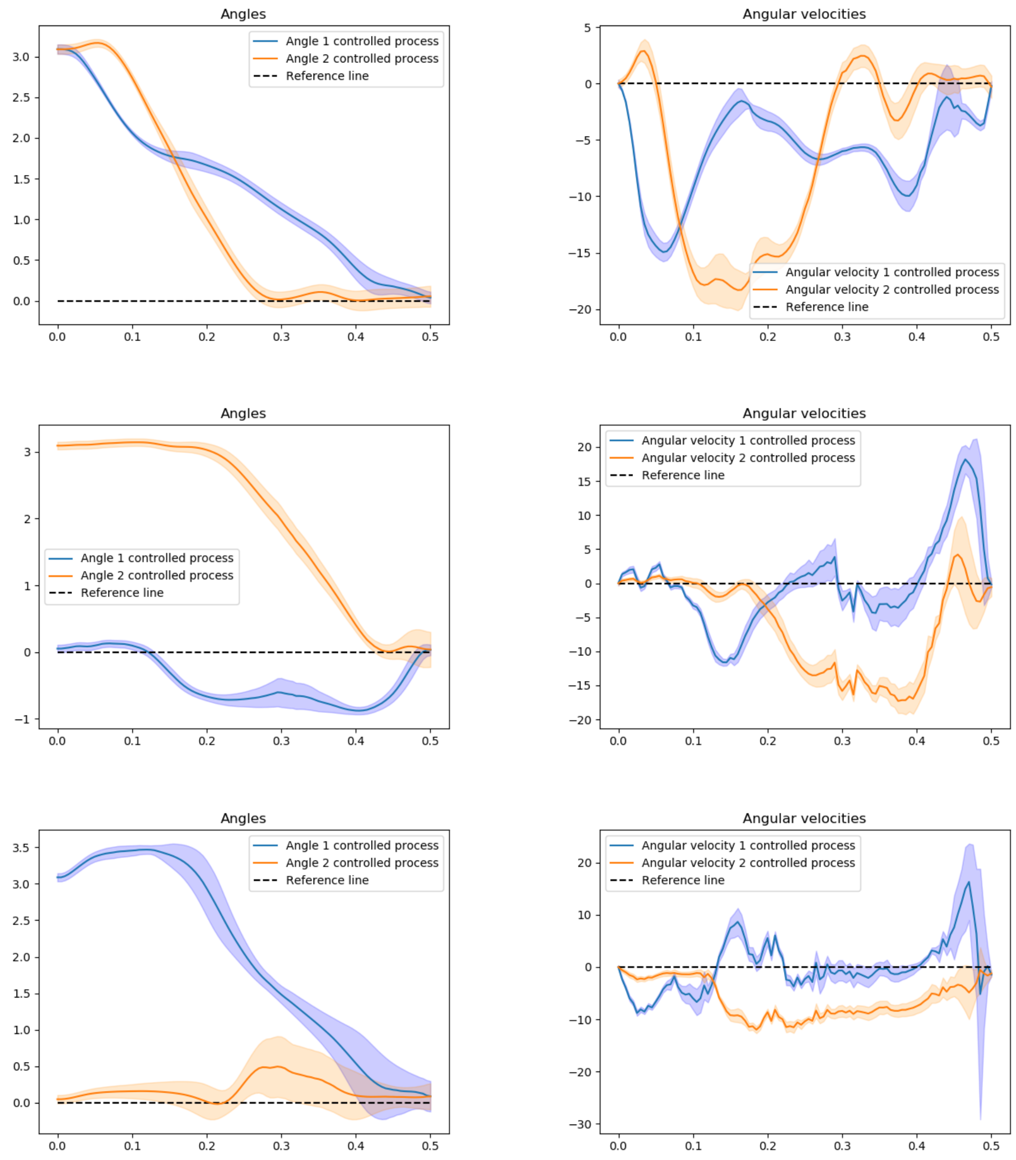


Figure 6.17: The average of 500 test samples of Angle 1 and angle 2 to the left and angular velocity 1 and angular velocity 2 to the right with empirical 95% confidence intervals. Upper: Case I. Mid: Case II. Lower: Case III.

samples and compares the controlled and the uncontrolled processes with settings as in Case I. Figure 6.19 shows the average control signal at each time point with a 95% empirical confidence interval with settings as in Case I. To save space, we do not show the corresponding plots for Cases II and Case III.

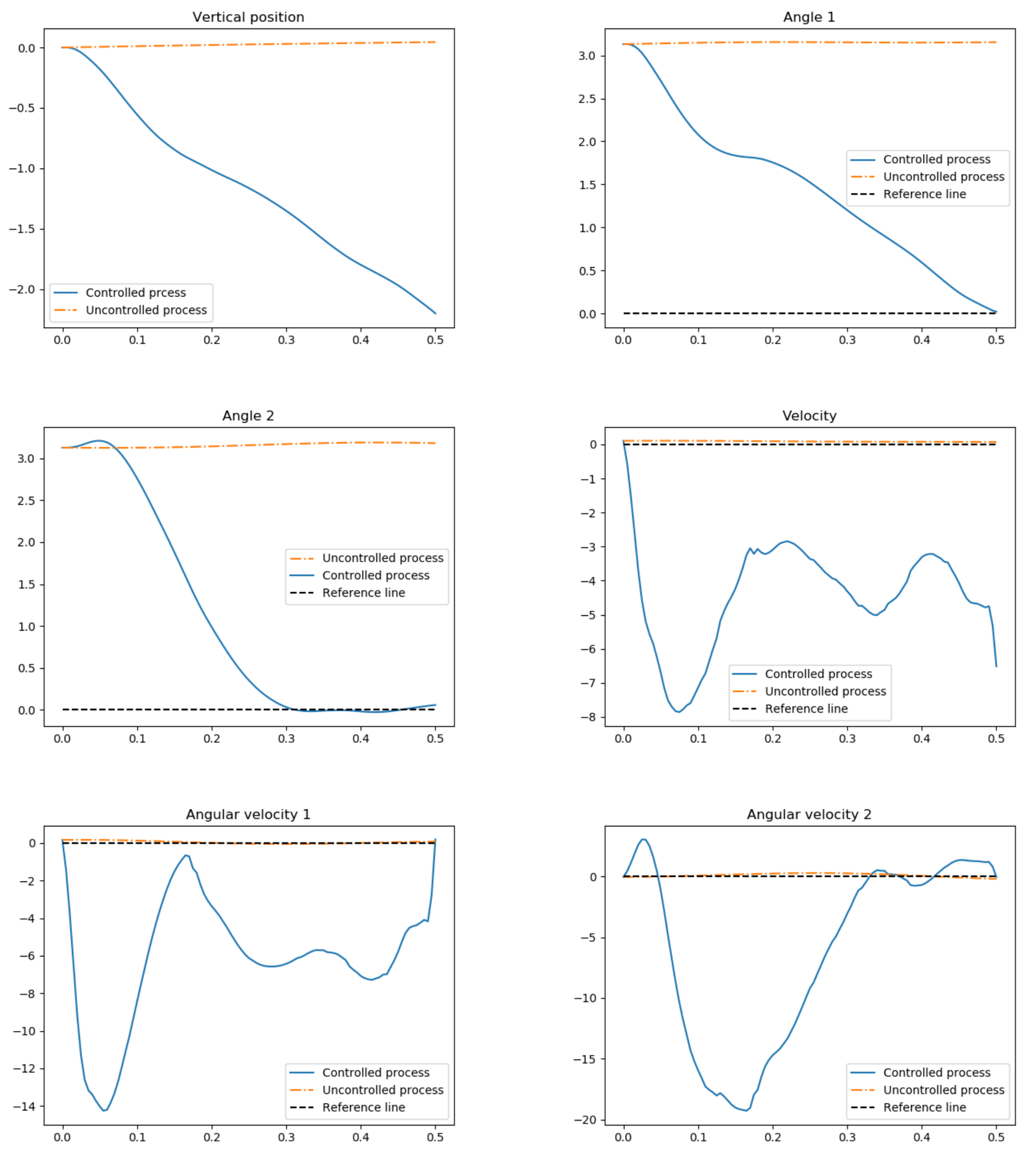


Figure 6.18: One typical, randomly chosen sample of a controlled and uncontrolled process with settings as in Case I.

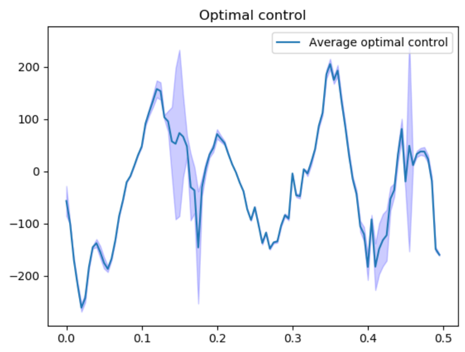


Figure 6.19: The average control signal with an empirical 95% confidence interval with settings as in Case I.

6.3.2 Some strategies to improve the robustness of the control

In the same way as for the single pendulum we try to increase the robustness of the control by using initial values over a larger spatial domain and increasing the noise level in the state equation. In our first experiment we use the settings as in Case IV. Note that the only difference between Case I and Case IV is the higher noise level in Case IV. The average angles and angular velocities are displayed in the upper plots in Figure 6.20 and by comparing to the upper plots in Figure 6.17 we see that the empirical 95% confidence intervals are significantly wider, especially for larger t. The left plot in Figure 6.21 shows the average control signal and an empirical 95% confidence interval. We see that the confidence interval for the control is narrow for small t and wider for $t > 0.35$ which indicates that the control is more robust for larger t. By comparing the average control signal to the left in Figure 6.21 and Figure 6.19 we see that by increasing s to 0.3 the oscillation in the control increased, which indicates a less optimal control. For the next experiment we use the settings as in Case V, which is to keep the high noise level and to increase the spatial domain of initial states. The average angles and angular velocities are displayed to the right in Figure 6.20 and wee see that the empirical 95% confidence intervals have increased significantly. We see that it is possible to control processes with very different initial values. The plot to the right in Figure 6.21 shows the average control signal and an empirical 95% confidence interval. The confidence interval is wide for all t but it oscillates notably.

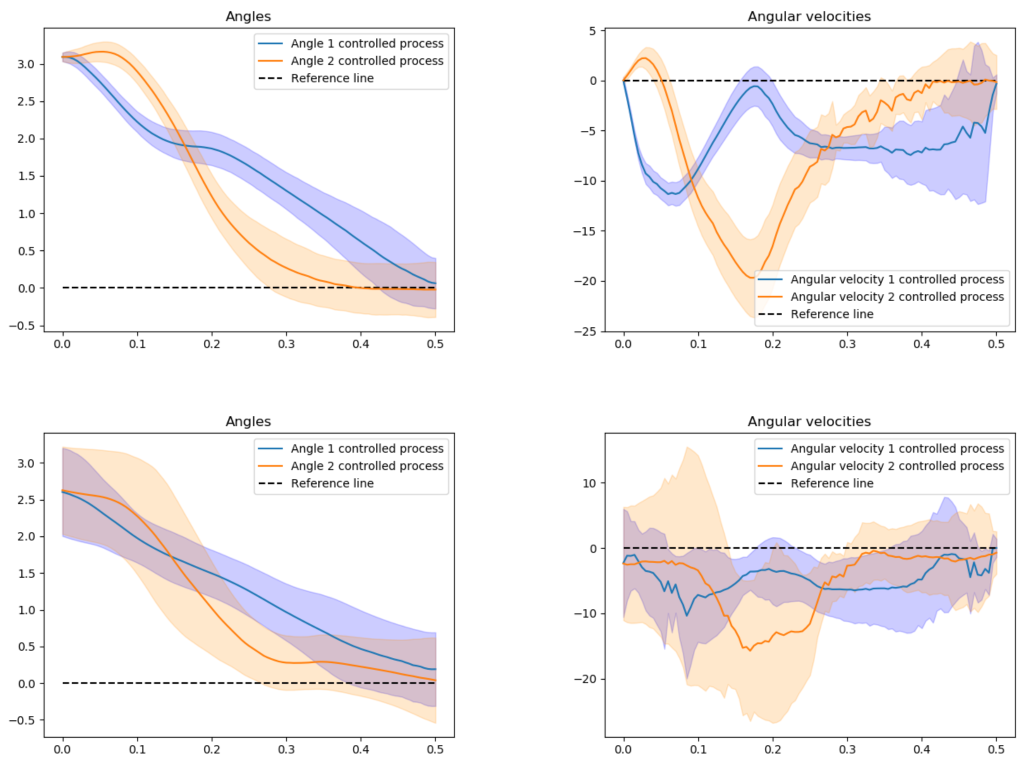


Figure 6.20: The average of 500 test samples of Angle 1 and angle 2 to the left and angular velocity 1 and angular velocity 2 to the right with empirical 95% confidence intervals. Upper: Case IV. Lower: Case V.

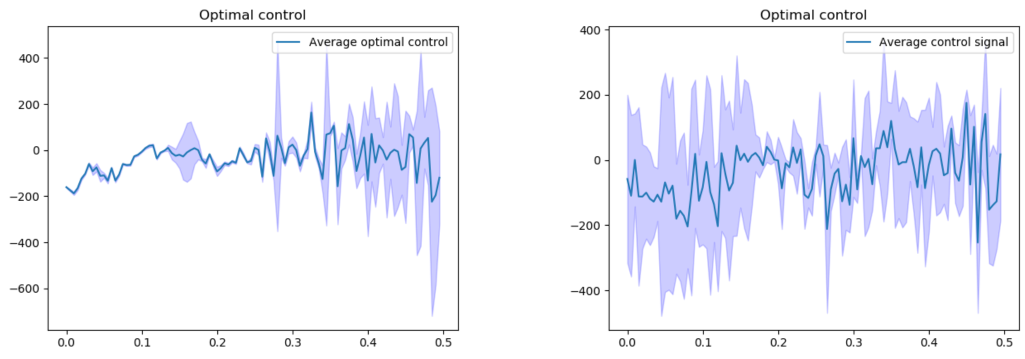


Figure 6.21: The average of 500 control signals with an empirical 95% confidence interval. Left: Case IV. Right: Case V.

7
Discussion

In this chapter we briefly discuss the results of this thesis. In particular we point to some aspects that need further investigation in order to properly determine how well the algorithms perform. Furthermore, we discuss some ideas for further development of the algorithms.

7.1 Evaluation of the results

Recall Chapter 6, where the different algorithms, ability to control the single and double pendulums was evaluated. This was done by studying the controlled processes and by comparing the cost during training for the different algorithms. For the basic cases, *i.e.*, those cases where we did not aim to make the control robust, we could safely draw the following conclusions:

- All three algorithms were able to swing up the pendulum/pendulums and control it around its fully inverted unstable equilibrium point,
- The FBSDE controller and the DPE controller performed significantly better in terms of the cost in most cases. On the whole, it seems that the FBSDE controller performs slightly better than the DPE controller.

From a practical point of view, this may be sufficient, but from a mathematical perspective we would like to know how close to optimality our approximate controls are. This could be done by considering problems where an analytic solution to the HJB equation exists and then comparing the optimal Markov control policy to our approximate control signal. One problem with this approach is that we are limited to a few control problems, which may not be representative. Another option is to adopt an engineering approach and redo experiments done by others with our control algorithms and compare the results.

In order to better assess the performance of the various algorithms, a hyperparameter optimization would have been required. The optimization should be done over all parameters associated with the neural network, and separately for the different algorithms.

7.2 Future work

The first and arguably most important improvement of the algorithms would be to change to a energy preserving numerical scheme for the state equations. It is a well known fact that the energy in the system increases with time when the forward

Euler method (the Euler-Maruyama method's counterpart for deterministic state equations) is used to approximate a dynamical system. An alternative would have been to use the Crank-Nicolson method, which is an energy preserving method (in the deterministic case) and also more stable to initial values. To be precise, instead of the Euler-Maruyama method given by (5.18) we would use the Crank-Nicolson method which reads as

$$X_{k+1} = X_k + \frac{1}{2}\left(f(X_k) - f(X_{k+1})\right)\Delta t + \sigma(W_{k+1} - W_k); \quad X_0 = x_0, \tag{7.1}$$

to approximate the state equation. This would allow us to use a larger T in the case of the double pendulum.

As we saw in Section 3.6.2 the BSDE controller did perform well in the LQR problem but could not be used in nonlinear problems because the neural network was trained in the wrong spatial domain. This problem could possibly be resolved by a modification of the training algorithm.

Set $k = N - 1$ and repeat until $k = 0$:

(1) Choose an initial value x_k from a spatial domain,
(2) View x_k as initial data, and for fixed θ_m for $m > k > N$, use the training algorithm for the BSDE solver to approximate θ_k, *i.e.* all parameters corresponding to the k:th subnetwork are trained,
(3) Fix θ_k and let $k = k - 1$.

With this algorithm we train the $N+1$ subnetworks individually N plus $N-2$ extra networks to approximate Y_k for $k = 1, 2, \ldots, N-1$. With the algorithm above we solve a BSDE instead of a FBSDE. If this is an advantage or not needs to be further investigated.

In addition to the suggested algorithms above, we briefly mention some other ways to formulate the stochastic optimal control problem, which would lead to completely different methods. The first suggestion is to consider a constrained control problem instead. With this setting we would introduce restrictions on the control, *e.g.* that the force used needs to be less than a certain limit. The final suggestion is to instead consider an infinite horizon control problem. Without going into details we would loose the time dependency of the control and the optimal control would be given by an algebraic HJB equation.

Bibliography

[1] Bellman, R. *Dynamic programming.* Courier Corporation, 2013 (Reprint of the 1957 edition).

[2] W, E., Han, J., & Jentzen, A. *Deep learning-based numerical methods for high-dimensional parabolic partial differential equations and backward stochastic differential equations.* Communications in Mathematics and Statistics, 2017.

[3] Han, J., & Jentzen, A. *Solving high-dimensional partial differential equations using deep learning.* arXiv preprint arXiv:1707.02568, 2017.

[4] Berg, J., & Nyström, K. *A unified deep artificial neural network approach to partial differential equations in complex geometries.* arXiv preprint arXiv:1711.06464, 2017.

[5] Raissi, M. *Forward-Backward Stochastic Neural Networks: Deep Learning of High-dimensional Partial Differential Equations.* arXiv preprint arXiv:1804.07010, 2018.

[6] Berg, J., & Nyström, K. *A unified deep artificial neural network approach to partial differential equations in complex geometries.* arXiv preprint arXiv:1711.06464, 2017.

[7] Bismut, J-M. *An introductory approach to duality in optimal stochastic control.* SIAM review 20.1, 1978.

[8] Pardoux, E., & Peng, S. *Adapted solution of a backward stochastic differential equation.* Systems & Control Letters, 1990.

[9] Pardoux, E., & Peng, S. *Backward stochastic differential equations and quasilinear parabolic partial differential equations.* Stochastic partial differential equations and their applications. Springer, Berlin, Heidelberg, 1992.

[10] El Karoui, N., Peng, S., & Quenez, M.C. *Backward stochastic differential equations in finance.* Mathematical finance, 1997.

[11] Zhang, J. *Backward Stochastic Differential Equations: From Linear to Fully Nonlinear Theory* Springer Vol 86, 2017.

[12] Zhang, J. *Backward Stochastic Differential Equations: From Linear to Fully Nonlinear Theory.* Springer, 2017.

[13] Delong, L. *Backward stochastic differential equations with jumps and their actuarial and financial applications.* Springer, 2013.

[14] Touzi, N. *Optimal stochastic control, stochastic target problems, and backward SDE (Vol. 29).* Springer Science & Business Media, 2012.

[15] Oksendahl, B. *Stochastic differential equations.* Springer, 1995.

[16] Yamada, T. and Watanabe, S. *On the uniqueness of solutions of stochastic differential equations.* Journal of Mathematics of Kyoto University, 1971.

[17] Skorokhod, A.V. *Studies in the theory of random processes.* Vol. 7021. Courier Dover Publications, 1965.

[18] Klebaner, F.C. *Introduction to stochastic calculus with applications.* World Scientific Publishing Company, 2005.

[19] Fleming, W.H., & Soner, H.M. *Controlled Markov processes and viscosity solutions.* Springer Science & Business Media, 2006.

[20] Yong, J., & Zhou, X. Y. *Stochastic controls: Hamiltonian systems and HJB equations* Springer Science & Business Media, 1999.

[21] Nisio, M. *Stochastic Control Theory. Dynamic Programming Principle, 2nd edn.* Probability Theory and Stochastic Modelling, 2015.

[22] Boubaker, O. *The inverted pendulum: A fundamental benchmark in control theory and robotics.* Education and e-Learning Innovations (ICEELI), 2012 international conference on. IEEE, 2012.

[23] Frisk, D. *A Chalmers University of Technology Master's thesis template for LaTeX.* Unpublished, 2016.

[24] Grimmett, G. & Stirzaker, D. *Probability and random processes.* Oxford university press, 2001.

[25] Hu, Y., & Peng, S. *Solution of forward-backward stochastic differential equations.* Probability Theory and Related Fields, 1990.

[26] Kac, M. *On distributions of certain Wiener functionals.* Transactions of the American Mathematical Society, 1949.

[27] Bogdanov, A. *Optimal control of a double inverted pendulum on a cart.* Oregon Health and Science University, Tech. Rep. CSE-04-006, OGI School of Science and Engineering, Beaverton, OR, 2004.

[28] Karatzas, I., Lehoczky, J. P., & Shreve, S. E. *Optimal portfolio and consumption decisions for "a small investor" on a finite horizon.* SIAM journal on control and optimization, 1987.

[29] Lokenath, D., & Mikusiński, P. *Hilbert spaces with applications.* Academic press, 2005.

[30] Dragomir, S.S. *Some Gronwall type inequalities and applications.* New York: Nova Science Publishers, 2003.

[31] Percival, I.C., & Richards, D.*Introduction to dynamics.* Cambridge University Press, 1982.

[32] Kloeden, P.E., Platen, E., & Schurz, H. *Numerical solution of SDE through computer experiments.* Springer Science & Business Media, 2002.

[33] Folland, G.B. *Real analysis: modern techniques and their applications.* John Wiley & Sons, 2013.

[34] Debnath, L., & Mikusiński, P. *Hilbert spaces with applications.* Academic press, 2005.

[35] Hodgkin, A.L., & Huxley, A.F. *A quantitative description of membrane current and its application to conduction and excitation in nerve.* The Journal of physiology, 1952.

[36] Goodfellow, I *Deep learning.* Vol. 1. Cambridge: MIT press, 2016.

[37] Hahnloser, R, et al. *Digital selection and analogue amplification coexist in a cortex-inspired silicon circuit.* Nature 405.6789, 2000.

[38] Glorot, X, Bordes, A & Bengio, Y *Deep sparse rectifier neural networks.* Proceedings of the fourteenth international conference on artificial intelligence and statistics, 2011.

[39] Dodge, S., & Karam, L. *A study and comparison of human and deep learning recognition performance under visual distortions. Computer Communication and Networks (ICCCN)*, 2017.

[40] Cireşan, D. C., Meier, U., Gambardella, L. M., & Schmidhuber, J. *Deep, big, simple neural nets for handwritten digit recognition.* Neural computation,, 2010.

[41] Sirignano, J, & Spiliopoulos, K. *DGM: A deep learning algorithm for solving partial differential equations.* Journal of Computational Physics, 2018.

[42] Thibault, J., & Grandjean, B. P. *A neural network methodology for heat transfer data analysis.* International Journal of Heat and Mass Transfer, 1991.

[43] Da Silva, I.N., Spatti, D.H., Flauzino, R.A., Liboni, L.H.B., & dos Reis Alves, S.F. *Artificial neural networks.* Springer International Publishing, 2017.

[44] Ruder, S. *An overview of gradient descent optimization algorithms.* arXiv preprint arXiv:1609.04747, 2016.

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

[46] W, E. & Han, J. *Deep Learning Approximation for Stochastic Control Problems.* arXiv preprint arXiv:1611.07422, 2016.