Stochastic Differential Games and Decentralized Control

A Robust Deep Learning Solver for Multi Agent Games

Master's Thesis in Engineering Mathematics and Computational Science

Benjamin Elm Jonsson
Bekir Fazlija

DEPARTMENT OF MATHEMATICAL SCIENCES

CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden 2025
www.chalmers.se

Master's thesis 2025

Stochastic Differential Games and Decentralized Control

A Robust Deep Learning Solver for Multi Agent Games

BENJAMIN ELM JONSSON
BEKIR FAZLIJA

CHALMERS
UNIVERSITY OF TECHNOLOGY

Department of Mathematical Sciences
Division of Analysis and Probability Theory
Chalmers University of Technology
Gothenburg, Sweden 2025

Stochastic Differential Games and Distributed Control
A Robust Deep Learning Solver for Multi-Agent Games
BENJAMIN ELM JONSSON
BEKIR FAZLIJA

Supervisor: Kristoffer Andersson, Department of Economics, Universitá di Verona.
Supervisor: Per Ljung, SAAB Surveillance AB.
Examiner: Peter Helgesson, Department of Mathematical Sciences, Chalmers University of Technology.

Master's Thesis 2025
Department of Mathematical sciences
Division of Analysis and Probability Theory
Chalmers University of Technology
SE-412 96 Gothenburg
Telephone +46 31 772 1000

Typeset in LaTeX
Printed by Chalmers Reproservice
Gothenburg, Sweden 2025

Stochastic Differential Games and Distributed Control
A Robust Deep Learning Solver for Multi-Agent Games
BENJAMIN ELM JONSSON
BEKIR FAZLIJA
Department of Mathematical Sciences
Chalmers University of Technology

Abstract

Multi-agent stochastic control games naturally give rise to coupled forward backward stochastic differential equations (FBSDE) for each agent. In such cases, dependencies exist both between the forward and backward equations and across agents, resulting in a highly non-trivial system to solve. In this thesis, a novel solution algorithm for such systems is presented, that is more robust compared to the existing Deep Fictitious Play method. The algorithm presented has been validated through several numerical examples. An analytical solution for linear–quadratic–Gaussian differential games is derived to validate the algorithm for problems where the Deep Fictitious Play algorithm has been shown to fail. This demonstrates the improved capabilities compared to the algorithms in the literature. In addition, a complex numerical example is designed that models a heterogeneous stochastic control game, which describes a swarm of drones following a predefined trajectory.

Keywords: Stochastic differential equation, Forward backward differential equation, Game theory, Stochastic control, Fictitious play, Multi-agent games.

Acknowledgements

First of all, we would like to thank our two supervisors Per Ljung and Kristoffer Andersson. Pers' continuous ambition to learn with us and eagerness to help throughout the thesis regardless of his previous expertise on the subject has been invaluable. Kristoffers expert help has allowed us to get a deeper understanding of the subject and a greater intuition, without which the results we managed to achieve would not be possible. Furthermore, we would like to thank Adam Andersson for greatly interesting discussions on the possible games as well as incredible lecture notes. We would also like to thank Peter Helgesson whose feedback has had a tremendous impact on the thesis.

Lastly, we would like to thank SAAB Surveillance and Per Gustafsson for the opportunity to conduct our thesis at SAAB this spring. It has been an incredible experience.

Benjamin Elm Jonsson & Bekir Fazlija, Gothenburg, June 2025

List of Acronyms

Below is the list of acronyms that have been used throughout this thesis listed in alphabetical order:

BMO	Bounded Mean Oscillation
BSDE	Backward Stochastic Differential Equation
C-FBSDE	Control–Forward Backward Stochastic Differential Equation
FBSDE	Forward Backward Stochastic Differential Equation
FDM	Finite Difference Method
FEM	Finite Element Method
FVM	Finite Volume Method
HJB	Hamilton–Jacobi–Bellman
LQG	Linear–Quadratic–Gaussian
NN	Neural Network
ODE	Ordinary Differential Equation
PDE	Partial Differential Equation
SDE	Stochastic Differential Equation
SPDE	Stochastic Partial Differential Equation

Nomenclature

Below is the nomenclature of indices, sets, parameters, and variables that have been used throughout this thesis.

Indices

i	Player index
d	Dimension of the state process X_t^i
k	Dimension of the Brownian motion W_t^i
ℓ	Dimension of the control process u_t^i

Sets

$C^{\ell,k}$	Functions ℓ-times differentiable in time and k-times differentiable in space
$C_{\mathrm{b}}^{\ell,k}$	As above, with every such derivative bounded
$\mathbb{H}_T^2$	Mean-square continuous and predictable processes with finite expected square integral
$\mathbb{L}^2$	$\mathcal{F}_T$-measurable random variables with finite expectation
$\mathbb{S}_T^2$	Mean-square continuous and predictable processes with finite supremum over time of the expectation of the process
$\mathbb{S}^k$	Space of all $k \times k$ symmetric matrices
$\mathbb{S}_+^k$	Space of all $k \times k$ symmetric, positive semi-definite matrices
$N(\mu, \sigma^2)$	Normal distribution with mean μ and variance σ^2

Parameters

N	Number of players

T	Final time
x_0^i	Initial state of player i
$b^i(t, \mathbf{x}, \mathbf{u})$	Drift coefficient for player i
$\sigma^i(t, \mathbf{x})$	Diffusion coefficient for player i
$f^i(t, \mathbf{x}, \mathbf{u})$	Running cost for player i
$g^i(\mathbf{x})$	Terminal cost for player i

Variables

$X_t^{i,\mathbf{u}}$	State of player i
$\mathbf{X}_t^{\mathbf{u}}$	Concatenated state of all players
Y_t^i	Backward process of the FBSDE for player i
Z_t^i	Control process of the FBSDE for player i
u_t^i	Control applied by player i
$\mathbf{u}_t$	Concatenated control of all players
W_t	k-dimensional Brownian motion
$V^i(t, \mathbf{x})$	Value function of player i
$J^i(t, \mathbf{x}; \mathbf{u}_t)$	Cost functional of player i

Contents

List of Figures

1
Introduction

In an increasingly hostile world, the demand for autonomous defensive capabilities is growing tremendously. Enhanced defensive measures are essential to protect citizens' freedom and lives. Effectively countering advanced weapon systems requires efficient and reliable autonomous defense systems. Such systems, e.g., drones or swarms of drones, offer substantial defensive capabilities while minimizing risk. The defensive capabilities of such systems are highly dependent on the control algorithms governing the drones [1]. In fact, the thesis concerns itself with autonomous swarms of drones where the control is decentralized. In detail, this means that each drone controls itself given the control and positions of the other drones in the swarm. Drone swarms are well suited for surveillance missions, which may involve following a designated path and gathering information on a specific location and its perimeter.

Beyond defense, stochastic control systems of this kind have applications in a number of fields, such as medicine, quantitative finance, and engineering. For instance, in [2], it is shown that such systems can be used in cancer treatment, where stochastic control models are used to model the unpredicted growth of tumors. This leads to more effective and personalized treatment for patients. The finance industry, moreover, commonly uses stochastic partial differential equations (SPDEs) models to achieve optimal profit in option pricing, which is often accomplished by solving the Black–Scholes SPDE, as in [3]. Furthermore, in the field of engineering, stochastic optimal control is applied to regulate several systems optimally. An example is the swarm of autonomous drones, as presented in this thesis.

As the dimension of the stochastic control problem increases, classical methods struggle to provide solutions at a reasonable computational cost. This is due to *curse of dimensionality*, where the computational cost grows exponentially as the dimension of the problem increases. Methods such as the finite element method (FEM), the finite volume method (FVM), or the finite difference method (FDM) become practically intractable. To circumvent the increased computational cost, the problem is reformulated into a forward backward stochastic differential equation (FBSDE), which is given by

$$\mathrm{d}X_t = b(t, X_t, Y_t, Z_t)\,\mathrm{d}t + \sigma(t, X_t, Y_t, Z_t)\,\mathrm{d}W_t, \quad X_0 = x_0, \tag{1.1a}$$

$$\mathrm{d}Y_t = f(t, X_t, Y_t, Z_t)\,\mathrm{d}t - Z_t\,\mathrm{d}W_t, \qquad Y_T = g(X_T), \tag{1.1b}$$

for $t \in [0, T]$. The FBSDE describes a system of two stochastic differential equations (SDEs). The forward equation (1.1a) describes the dynamics of the system, governing how the players' states evolves. Meanwhile, the backward equation (1.1b)

describes the cost associated with the players' actions. In [4], the authors developed the Deep FBSDE method, showing that the proposed method can efficiently solve high-dimensional problems with the help of neural networks (NNs). A further generalization of this was presented in [5], which introduces the Deep fictitious play algorithm for N-player games based on the work in [4]. The generalized solution algorithm utilizes the concept of fictitious play, in which players interact in an iterative process to find the optimal strategy.

The details regarding (1.1) are presented in Chapter 2. To combat the curse of dimensionality, the process $Z = (Z_t)_{t\in[0,T]}$ is parameterized using NNs, which was, as far as we know, first done in [4]. Since the initial value of the backward equation (1.1b) is unknown, the Deep FBSDE also approximates this condition using NN. However, it has been showed in [6] that Deep FBSDE fails for problems derived from stochastic optimal control, that is, when the drift and driver depend on the state process and the Z-process, $b(t, X_t, Z_t)$ and $f(t, X_t, Z_t)$. The authors presented a more robust algorithm, namely, the Deep C-FBSDE. It has been shown that this algorithm converges even when the forward and backward equations are strongly dependent. In the robust algorithm, the initial value is not a free parameter and instead a consequence of the system. Moreover, the loss function is modified to also minimize the cost associated with the stochastic control problem.

The Deep fictitious play is based on the Deep FBSDE method. It is reasonable to conclude that if the Deep FBSDE method fails to solve the stochastic control problem, the Deep fictitious play will also struggle when applied to multiple agents. The complexity of the resulting system increases in such cases. Thus, this thesis aims to generalize the robust solver Deep C-FBSDE to a multi-agent setting. The motivation behind this generalization is to allow the consideration of more complex systems, which in turn allows modeling more realistic games. This mainly includes problems stemming from stochastic control, where a stronger coupling between the forward and backward equations is common. This presents a clear research gap for a robust multi-agent solution algorithm, which we develop in this thesis. Further, the proposed algorithm is validated with numerical examples that the Deep fictitious play algorithm handles, as well as examples that it does not.

The thesis is structured as follows: Chapter 2 introduces the foundational theory on measure theory, stochastic processes, SDEs, and machine learning. Then in Chapter 3, we introduce the stochastic control problem and derive the Hamilton–Jacobi–Bellman (HJB) equation. A central theorem is stated, which shows that the solution of the HJB equation coincides with the solution of the stochastic control problem. We reformulate the HJB equation to derive the FBSDE, which is the system treated by the solvers. The same steps are repeated for the stochastic differential game with N players. In Chapter 4, stochastic control solvers, Deep FBSDE and Deep C-FBSDE, are examined. In Chapter 5, we present Deep fictitious play along with the proposed Multi C-FBSDE. In Chapter 5, the multi-agent methods are tested for different games, demonstrating that the Deep fictitious play fails for several games. Chapter 6 highlights a more complex numerical example that show-

cases the capabilities of the novel robust algorithm. Finally, Chapter 7 discusses the results presented throughout the thesis and explores potential future work.

2
Mathematical background

In this chapter, we introduce the mathematical theory needed for the thesis. We begin with measure theory, stochastic processes, and stochastic calculus, and in addition, we present the theoretical background of neural networks. Hence, the following chapter aims to include the necessary mathematics for the rest of the thesis to stand upon.

2.1 Measure theory

Measure theory is a very impactful field of mathematics. It forms the foundation for several important concepts, such as the Lebesgue integral. Furthermore, the idea of measurability is vital in the field of stochastic calculus. This section introduces the concept of a measure and a measurable space. With these in mind, we present the filtered probability space on which the SDEs of the thesis are defined. Then, key concepts from stochastic calculus are discussed. Finally, the SDE is introduced in full, and a theorem is stated that guarantees the existence of a unique strong solution. The interested reader is referred to [7] for further material on measure theory.

We begin this section by presenting the problem of measure, described in e.g., [7]. The idea of a measure is intuitive for geometric bodies E in one to three dimensions, it corresponds to length, area, or volume, respectively. However, when considering the real line $\mathbb{R}$ instead of geometric bodies, problems arise. The trivial notion of a measure is to count the number of points in the set. However, to show that this notion is inadequate, consider the intervals $[0, 1]$ and $[0, 2]$. One can form a bijection $x \mapsto 2x$ between the two intervals. That is, one can then disassemble the first interval in uncountably many points and reconfigure them into a set twice as long. Therefore, two sets with the same number of points need not have the same measure. To remedy the situation, one might only consider finite partitions. The *Banach–Tarski* paradox shows that problems remain, see [8]. It states that the unit ball can be partitioned into finitely many pieces. This partition can then be assembled into two copies of the original ball. Here, the original ball is reassembled to two copies of itself, resulting in the original ball having the same measure as two copies of itself. Thus, the measure can not distinguish between a single ball and two copies of it, which naturally, is a problem. This motivates the need for a formal and rigorous definition of a measure. Before we present the definition of a measure, the notion of a σ-algebra is needed.

Definition 1 (σ-algebra)**.** *A σ-algebra, or σ-field, of an arbitrary set E is a collection $\mathcal{M}$ of subsets of E. Then, for the collection $\mathcal{M}$ the following holds*
(i) $E \in \mathcal{M}$.
(ii) $\emptyset \in \mathcal{M}$.
(iii) $A_1, A_2, \ldots \in \mathcal{M}$ *implies* $\bigcup_{i=1}^{\infty} A_n \in \mathcal{M}$.
(iv) $A \in \mathcal{M}$ *implies* $A^c \in \mathcal{M}$, *i.e.,* $\mathcal{M}$ *is closed under complementation.*
This means that the collection $\mathcal{M}$ is closed under countable set theoretic operations.

One common and important σ-algebra is the Borel σ-algebra of $\mathbb{R}$, denoted $\mathcal{B}$. It is generated by all open subsets of $\mathbb{R}$, and the subsets in $\mathcal{B}$ are called Borel sets. Another important σ-algebra is the smallest σ-algebra which contains a collection $\mathcal{E}$ of subsets of a set E. This is often denoted $\sigma(\mathcal{E})$. The smallest σ-algebra which contains a collection of subsets of some set is said to be generated by that set.

We are now ready to define a measurable space and a measure space. A measure space is specifically a measurable space that is equipped with a *measure*.

Definition 2 (Measurable space)**.** *Let E be an arbitrary set and $\mathcal{M}$ a σ-algebra of subsets of E. Then the pair $(E, \mathcal{M})$ is called a measurable space.*

We now introduce the natural extension of the σ-algebra, the product σ-algebra. Let $(E, \mathcal{M})$ and $(R, \mathcal{N})$ be two measurable spaces. We say that $A \times B$, where $A \in \mathcal{M}$ and $B \in \mathcal{N}$, is a measurable rectangle, which is a subset of the product space $E \times R$. The σ-algebra on $E \times R$ is generated by the collection of all measurable rectangles. This is defined by the product σ-algebra

$$\mathcal{M} \otimes \mathcal{N} = \sigma(\{A \times B \,:\, A \in \mathcal{M},\, B \in \mathcal{N}\}).$$

Definition 3 (Measure space)**.** *Let $(E, \mathcal{M})$ be a measurable space. If $\mu : \mathcal{M} \to [0, \infty]$ satisfies*
(i) $\mu(\emptyset) = 0$,
(ii) for every pairwise, disjoint arbitrary set $A_1, A_2, \ldots \in \mathcal{M}$

$$\mu\left(\bigcup_{i=1}^{\infty} A_i\right) = \sum_{i=1}^{\infty} \mu(A_i),$$

then μ is called a measure and the triple $(E, \mathcal{M}, \mu)$ a measure space. The latter property (ii) is commonly referred to as countable additivity and it implies finite additivity.

A measure space $(E, \mathcal{M}, \mu)$ with $\mu(E) = 1$ is a probability space $(\Omega, \mathcal{F}, \mathbb{P})$. The probability space consists of a sample space Ω, a σ-algebra $\mathcal{F}$ and a probability measure $\mathbb{P} : \mathcal{F} \to [0, 1]$. Setting $E = \Omega$, $\mathcal{M} = \mathcal{F}$ and $\mu = \mathbb{P}$ shows that every probability space is simply a measure space whose total mass equals 1. In this thesis, we will mostly consider complete probability spaces. For this, we present the definition of a complete measure space.

Definition 4 (Complete measure space)**.** *A measure space $(E, \mathcal{M}, \mu)$ is said to be complete if whenever $B \in \mathcal{M}$ with $\mu(B) = 0$ and $A \subseteq B$, then the set $A \in \mathcal{M}$ and thus $\mu(A) = 0$. That is, every subset of a null set are themselves in the σ-algebra $\mathcal{M}$.*

We are now ready to present the concept of a measurable function. This is a central concept throughout the thesis since non-measurable functions cannot be integrated or treated probabilistically.

Definition 5 (Measurable function)**.** *Let $(E, \mathcal{M})$ and $(R, \mathcal{N})$ be arbitrary measurable spaces. Then a function $f : E \to R$ is measurable if for all $B \in \mathcal{N}$*

$$f^{-1}(B) \coloneqq \{\omega \in E : f(\omega) \in B\} \in \mathcal{M}.$$

More precisely, f is said to be $(\mathcal{M}, \mathcal{N})$-measurable. A measurable function on a probability space is called a random variable.

To build intuition for the concepts introduced in this section, we will consider the classic example of a coin toss.

Example 1 (Fair coin)**.** *Consider a game of tossing a coin twice. Then the set of all possible outcomes, the sample space, is $\Omega \coloneqq \{HH, HT, TH, TT\}$. Here, H denotes heads and T denotes tails, the first letter records the result of the first toss, and the second letter indicates the results of the second toss. In this scenario, the σ-algebra $\mathcal{F}$ contains all subsets of Ω as well as the empty set and Ω itself. The size of the σ-algebra grows quickly as it contains $2^{|\Omega|} = 16$ elements, here $|\Omega|$ denotes the number of events $\omega \in \Omega$. The probability measure $\mathbb{P}$ is defind as uniform. That is, for every $\omega \in \Omega$, $\mathbb{P}(\{\omega\}) = 1/4$. The resulting probability space is $(\Omega, \mathcal{F}, \mathbb{P})$. Then, we define the random variable $X : \Omega \to \mathbb{R}$ by*

$$X(\omega) \coloneqq \begin{cases} +5, & \omega = HH, \\ +1, & \omega = TH, \\ -1, & \omega = HT, \\ -5, & \omega = TT. \end{cases}$$

Hence, the player gets a score of $+5$ for two heads, a score of -5 for two tails, a score of $+1$ for a tail followed by a head and a score of -1 for a head followed by a tail. It is trivial that the random variable is measurable as $\mathcal{F}$ contains every subset of Ω. Thus, $X^{-1}(B) \in \mathcal{F}$ for every Borel set $B \subseteq \mathbb{R}$.

Finally, we state the Lebesgue differential theorem for measurable functions, which is required for derivations in Section 3.1.2. Further, we refer to e.g., [9] for the proof.

Theorem 1 (Lebesgue differentiation theorem)**.** *Let $B_h(x)$ denote a small ball with diameter h centered at x. Then, if $f : \mathbb{R}^d \to \mathbb{C}$ is locally integrable, then we have for almost every $x \in \mathbb{R}^d$*

$$\lim_{h \to 0} \frac{1}{|B_h(x)|} \int_{B_h(x)} |f(x) - f(t)| \, \mathrm{d}t = 0,$$

and

$$\lim_{h\to 0} \frac{1}{|B_h(x)|} \int_{B_h(x)} f(t)\, \mathrm{d}t = f(x).$$

2.2 Stochastic processes

In this thesis, we approximate solutions of SDEs. These solutions are, in turn, stochastic processes. Before defining an SDE, we first recall stochastic processes and focus primarily on the process used for all SDEs in this thesis, namely, the Brownian motion. We begin by giving a formal definition of a stochastic process. This section is based on [10].

Definition 6 (Stochastic process)**.** *A stochastic process is a family* $X = (X_t)_{t\in[0,T]} = (X_t(\omega))_{t\in[0,T]}$*, for some* $T > 0$*, of random variables on the probability space* $(\Omega, \mathcal{F}, \mathbb{P})$*. In other words, a stochastic process is a collection of random variables. For each* $\omega \in \Omega$*, the map* $t \mapsto X_t(\omega)$ *is said to be a sample path.*

For the derivation in Section 3.1.2 we require Fubini's Theorem. Therefore, we now present it as it states an important property for the integration of a stochastic process. Moreover, we refer to [10] for the proof.

Theorem 2 (Fubini's Theorem)**.** *For a continuous and integrable stochastic process* $X = (X_t)_{t\in[0,T]}$*, it holds that*

$$\mathbb{E}\left[\int_0^T X_t\, \mathrm{d}s\right] = \int_0^T \mathbb{E}\left[X_t\right]\, \mathrm{d}s.$$

When working with stochastic processes, one usually works with a filtered probability space. That is, a probability space equipped with a *filtration*, an increasing σ-algebra. Thus, the natural next step is to introduce the concept of filtration.

Definition 7 (Filtration)**.** *Let* $(\Omega, \mathcal{F}, \mathbb{P})$ *be a probability space. A filtration* $(\mathcal{F}_t)_{t\in[0,T]}$ *is an increasing family of* σ*-algebras, where* $\mathcal{F}_s \subseteq \mathcal{F}$ *for all* $s \in [0, T]$*. For all* $s \leq t$*,* $\mathcal{F}_s \subseteq \mathcal{F}_t$*, indicating that the filtration consists of increasing* σ*-fields. A probability space equipped with a filtration is called a filtrated probability space, commonly written as* $(\Omega, \mathcal{F}, (\mathcal{F}_t)_{t\in[0,T]}, \mathbb{P})$*.*

For a σ-algebra, a filtration contains all information known about a system up to the current time as well as all of its history. Moreover, recall the definition of a measurable random variable. When stochastic processes are considered, the process is said to be *adapted* if the random variables for each time t are measurable with respect to the filtration $(\mathcal{F}_t)_{t\in[0,T]}$. In this thesis, we will require processes to be adapted in order to get well-behaved stochastic integrals. A formal definition of adapted processes is outlined below.

Definition 8 (Adapted process)**.** *Let $(\Omega, \mathcal{F}, (\mathcal{F}_t)_{t\in[0,T]}, \mathbb{P})$ be a filtrated probability space. Further, let the measurable space $(R, \mathcal{N})$ denote the state space of the process. A stochastic process $X = (X_t)_{t\in[0,T]}$ is said to be adapted with respect to the filtration $(\mathcal{F}_t)_{t\in[0,T]}$ if the random variable $X_t : \Omega \to R$ is $(\mathcal{F}_t, \mathcal{N})$-measurable for all $t \in [0, T]$.*

We are now ready to define the vector space $\mathbb{L}^2_T(\mathbb{R})$ which denotes the space of all $\mathcal{F}_T$-measurable random variable $X : \Omega \to \mathbb{R}$ such that the norm $\|X\|^2_{\mathbb{L}^2_T(\mathbb{R})} \coloneqq \mathbb{E}\left[|X|^2\right]$ is finite. For the definition of the stochastic integral, presented in Section 2.3, we need a stronger condition on the process. That is, we need a simple adapted process instead, which we now define.

Definition 9 (Simple adapted process)**.** *Let $X = (X_t)_{t\in[0,T]}$ be a stochastic process on $(\Omega, \mathcal{F}, (\mathcal{F}_t)_{t\in[0,T]}, \mathbb{P})$. If there exists a grid of the time interval $0 = t_0 < t_1 < \ldots < t_n = T$, and square integrable random variables $\xi_0, \xi_1, \ldots, \xi_{n-1}$ that satisfies*

(i) ξ_0 is constant,

(ii) for each $i = 0, 1, \ldots n-1$, ξ_i is $\mathcal{F}_{t_i}$-measurable,

(iii) X_t can be written as piecewise constant of random variables ξ_i

$$X_t = \xi_0 \mathbb{1}_{\{0\}}(t) + \sum_{i=1}^{n} \xi_{i-1} \mathbb{1}_{(t_{i-1}, t_i]}(t), \quad t \in [0, T],$$

where $\mathbb{1}$ is an indicator function.

Then, X is a simple adapted process.

Another class of stochastic processes that is incredibly useful and serves a great purpose throughout the thesis are predictable processes. A predictable process in discrete time is a process such that X_t is measurable with respect to $\mathcal{F}_{t-1}$ for $t > 0$, i.e., we can predict the value of the random variable at time t only with information up to the time $t-1$. However, in continuous time, we need to define the predictable σ-algebra $\mathcal{P}$. Predictable processes are then processes that are measurable with respect to the predictable σ-algebra. We now formally define the predictable σ-algebra and the class of predictable processes.

Definition 10 (Predictable process)**.** *Let $\mathcal{F}_{t-} = \sigma(\cup_{s<t}\mathcal{F}_s)$ which is the smallest σ-field that contains $\mathcal{F}_s$ for all $s < t$. The predictable σ-algebra $\mathcal{P}$ is then generated by the stochastic sets $[s, t) \times A$ with $s < t$ and $A \in \mathcal{F}_{t-}$. A stochastic process $X = (X_t)_{t\in[0,T]}$ is then said to be predictable if it is measurable with respect to $\mathcal{P}$.*

Using the definition of a predictable process, we now state the definition of all spaces for stochastic processes that are used throughout the thesis. Firstly, we let $\mathbb{H}^p_T(\mathbb{R}^d)$, for $p \in \{1, 2\}$, denote the vector space of all mean-square continuous and predictable processes, i.e.

$$\mathbb{H}^p_T(\mathbb{R}^d) \coloneqq \left\{ X : [0, T] \times \Omega \to \mathbb{R}^d \middle| \|X_t\|_{\mathbb{H}^p_T(\mathbb{R}^d)} < \infty \right\},$$

where the norm is defined as

$$\|\cdot\|_{\mathbb{H}_T^p(\mathbb{R}^d)} \coloneqq \mathbb{E}\left[\left(\int_0^T |\cdot|^2 \,\mathrm{d}s\right)^{p/2}\right].$$

Further, we let $\mathbb{S}_T^2(\mathbb{R})$ denote the vector space of all mean-square continuous and predictable processes which have finite norm, where the norm is instead defined as

$$\|\cdot\|_{\mathbb{S}_T^2(\mathbb{R})} \coloneqq \sup_{t\in[0,T]} \mathbb{E}\left[(\cdot)^2\right].$$

The last thing needed before the introduction of a general SDE is the concept of Brownian motion. Brownian motion, also known as a Wiener process, denoted $W = (W_t)_{t\in[0,T]}$, is an adapted and continuous stochastic process. Let $W = (W_t)_{t\in[0,T]}$ be a Brownian motion on the filtrated probability space $(\Omega, \mathcal{F}, (\mathcal{F}_t)_{t\in[0,T]}, \mathbb{P})$, where the filtration is generated by W. In detail, the filtration generated by Brownian motion is, for each t, the increasing family of σ-algebras $\mathcal{F}_t = \sigma(\{W_s,\ 0 \le s \le t\})$. Then, the process has the following properties

(i) $W_0 = 0$,
(ii) $W_t - W_s$ is independent of $\mathcal{F}_r$ where $r \le s$,
(iii) $W_t - W_s$ is normal distributed $N(0, t-s)$ with mean zero and variance $t-s$,
(iv) W has $\mathbb{P}$-a.s. continuous sample paths.

The first property (i) indicates that all Brownian motion considered in this thesis starts at zero. The properties (ii) and (iii) show that all increments are independent of the past and are Gaussian with zero mean and variance equal to the increment size. Finally, the Brownian motion is $\mathbb{P}$-a.s. continuous over the entire interval $[0, T]$, but is differentiable nowhere, which is proved in [11].

2.3 Stochastic calculus

In this subsection, we present concepts from stochastic calculus. Along with measure theory, stochastic analysis is a popular field with great applicability in several areas, such as finance and, as in this thesis, control theory. We present the SDE that builds on the concepts presented in the previous sections. We note that all theorems are well known and we refer to e.g., [10, 12, 13] for the proofs. To conclude the section, a central result is stated, namely Itô's formula. It serves as the chain rule for stochastic processes driven by a Brownian motion.

To begin, we define the integral with respect to a Brownian motion. This integral is undefined as a classical integral, since almost every Brownian motion path has an infinite variation for any sub interval. Intuitively, the nowhere differentiability of Brownian motion and its infinite variation prevents the integral from converging. Therefore, another definition is needed for the second integral term, leading us to present the Itô integral process.

Theorem 3 (Itô integral process of simple adapted processes)**.** *Let $X = (X_t)_{t\in[0,T]}$ be a simple adapted process. Then, the Itô integral process is defined as*

$$\int_0^T X_t \,\mathrm{d}W_t = \sum_{i=1}^{n} \xi_{i-1} \left(W_i - W_{i-1}\right).$$

Further, the Itô integral process $(\int_0^t X_s \,\mathrm{d}W_s)_{t\in[0,T]}$ is adapted to $(\mathcal{F}_t)_{t\in[0,T]}$ and has continuous sample paths with zero-mean property $\mathbb{E}[\int X_t \,\mathrm{d}W_t] = 0$.

Since simple adapted processes are restrictive, and regular adapted processes are more common and general, we state a theorem that claims that the Itô integral process is well defined even when considering mean square-integrable adapted processes.

Theorem 4 (Itô integral process of adapted processes)**.** *Let $X = (X_t)_{t\in[0,T]}$ be a mean square-integrable stochastic process defined on the complete filtered probability space $(\Omega, \mathcal{F}, (\mathcal{F}_t)_{t\in[0,T]}, \mathbb{P})$. Then, there exists a sequence of simple adapted processes $(X_t^n)_{t\in[0,T]}$ each defined as*

$$X_t^n \coloneqq \xi_0 \mathbb{1}_{\{0\}}(t) + \sum_{i=1}^{n} \xi_{i-1} \mathbb{1}_{(t_{i-1},t_i]}(t), \quad t \in [0,T],$$

such that

$$\lim_{n\to\infty} \mathbb{E}\left[\int_0^T \left(X_t^n - X_t\right)^2 \mathrm{d}t\right] = 0.$$

Then, the Itô integral process $(\int_0^t X_s \,\mathrm{d}W_s)_{t\in[0,T]}$ is well defined as the mean square limit

$$\lim_{n\to\infty} \mathbb{E}\left[\left(\int_0^T X_t^n \,\mathrm{d}W_t - \int_0^T X_t \,\mathrm{d}W_t\right)^2\right] = 0.$$

Further, the zero-mean property $\mathbb{E}[\int X_t \,\mathrm{d}W_t] = 0$ holds.

The natural extension of the Itô integral process is the Itô process, which we now introduce. This is needed in order to state the powerful Itô's formula in Theorem 5, which is used, inter alia, for the derivation in Section 3.1.2.

Definition 11 (Itô process)**.** *An Itô process is a stochastic process $X = (X_t)_{t\in[0,T]}$ defined on the filtered probability space $(\Omega, \mathcal{F}, (\mathcal{F}_t)_{t\in[0,T]}, \mathbb{P})$ where*

$$X_t = x_0 + \int_0^t b(s) \,\mathrm{d}s + \int_0^t \sigma(s) \,\mathrm{d}W_s,$$

where x_0 is $\mathcal{F}_0$-measurable, $b = (b_t)_{t\in[0,T]}$ and $\sigma = (\sigma_t)_{t\in[0,T]}$ are adapted stochastic process, that satisfies

$$\int_0^T |b(s)| \,\mathrm{d}s < \infty, \qquad \int_0^T \sigma(s)^2 \,\mathrm{d}s < \infty.$$

Theorem 5 (Itô's formula)**.** *Let $\varphi \in C^{1,2}([0,T] \times \mathbb{R}^n)$, where $C^{1,2}([0,T] \times \mathbb{R}^n)$ is the space of all real-valued functions with one continuous time derivative and two continuous spatial derivatives. Let also $(\Omega, \mathcal{F}, (\mathcal{F}_t)_{t\in[0,T]}, \mathbb{P})$ be a filtered probability space. Then, let $W = (W_t)_{t\in[0,T]}$ denote a k-dimensional Brownian motion for some $k \in \mathbb{N}$. Further, let $b = (b_t)_{t\in[0,T]}$ and $\sigma = (\sigma_t)_{t\in[0,T]}$ be adapted stochastic processes w.r.t the filtration $(\mathcal{F}_t)_{t\in[0,T]}$ which is generated by Brownian motion and completed by the $\mathbb{P}$-null sets of Ω. Then, for any Itô process $X = (X_t)_{t\in[0,T]}$, it holds for all $0 \leq t_0 < t_1 \leq T$*

$$\varphi(t_1, X_{t_1}) - \varphi(t_0, X_{t_0}) = \int_{t_0}^{t_1} (\partial_s + L)\varphi(s, X_s)\,\mathrm{d}s + \int_{t_0}^{t_1} \nabla_x \varphi(s, X_s)^\top \sigma(s)\,\mathrm{d}W_s,$$

where ∂_t denotes the first order time derivative and the differential operator L is defined as

$$L\varphi(t,x) = \frac{1}{2}\mathrm{Tr}(\sigma(t)^\top \Delta_x \varphi(t,x)\sigma(t)) + b(t)^\top \nabla_x \varphi(t,x),$$

where ∇_x and Δ_x denote the gradient and the Hessian matrix, respectively. Further, $\mathrm{Tr}(A) \coloneqq \sum_{i=1}^n A_{ii}$ denotes the trace operator for any square matrix $A \in \mathbb{R}^{n\times n}$.

We are now ready to present a general SDE. Let $(\Omega, \mathcal{F}, (\mathcal{F}_t)_{t\in[0,T]}, \mathbb{P})$ be a filtered probability space where the filtration $(\mathcal{F}_t)_{t\in[0,T]}$ is generated by a k-dimensional Brownian motion $W = (W_t)_{t\in[0,T]}$ for some $k \in \mathbb{N}$. Moreover, the filtration is completed by the $\mathbb{P}$-null sets of Ω. Then the state process $X = (X_t)_{t\in[0,T]}$ is an Itô process, that is governed by the stochastic differential equation

$$\mathrm{d}X_t = b(t, X_t)\,\mathrm{d}t + \sigma(t, X_t)\,\mathrm{d}W_t, \quad X_0 = x_0, \tag{2.1}$$

where $b : [0,T] \times \Omega \times \mathbb{R}^d \to \mathbb{R}^d$ is the so-called drift coefficient which is allowed to itself be a stochastic process $b = (b_t)_{t\in[0,T]}$. Note that the drift term can be written as $b(t, \omega, x)$. However, the dependence on $\omega \in \Omega$ is understood and omitted. The same applies to the so-called diffusion coefficient $\sigma : [0,T] \times \Omega \times \mathbb{R}^d \to \mathbb{R}^{d\times k}$. If the drift or diffusion coefficient is a deterministic function, then it has no dependence on the sample space Ω. Moreover, we assume that they are adapted. Finally, the SDE is well-posed with the inclusion of an initial condition x_0 that is $\mathcal{F}_0$-measurable, and we require $X_0 = x_0$ $\mathbb{P}$-a.s. Then, by definition the SDE (2.1) can be equivalently expressed on integral form as

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

where the first integral term is a classic Lebesgue integral, and the second term is an Itô integral process.

Remark 1. *Throughout the thesis, only deterministic functions for the drift and diffusion coefficients will be considered. Therefore, the implicit dependency on the sample space Ω is dropped. Furthermore, all initial conditions will be deterministic.*

To have a well-founded discussion on SDEs it is necessary to ensure that a solution to (2.1) exists. Therefore, we present a theorem guaranteeing the existence of a unique, strong solution, given that the drift and diffusion coefficients satisfy some regularity conditions. In detail, the regularity conditions are Lipschitz continuity in space and uniformly in time, as well as linear growth in time.

Theorem 6 (Existence and uniqueness of strong solution)**.** *Let X_0 be an $\mathcal{F}_0$-measurable random variable with finite moment $\mathbb{E}\left[|X_0|^2\right] < \infty$. Then, if the coefficients b and σ are globally Lipschitz in x and uniformly in t for some $K_1 \in \mathbb{R}$ and $t \in [0,T]$, i.e,*

$$|b(t,x) - b(t,y)| + |\sigma(t,x) - \sigma(t,y)| \leq K_1|x-y|,$$

for all $x, y \in \mathbb{R}^d$, and as well as satisfy linear growth in space and uniform growth in time, i.e.,

$$|b(t,x)| + |\sigma(t,x)| \leq K_2(1+|x|),$$

for all $x \in \mathbb{R}^d, t \in [0,T]$ and some $K_2 \in \mathbb{R}$. Here, $|A| \coloneqq \sqrt{\mathrm{Tr}(AA^\top)}$ denotes the Frobenius norm for some matrix A. Then (2.1) *has a unique strong solution X and the following holds*

$$\mathbb{E}\left[\sup_{0\leq t\leq T}|X_t|^2\right] \leq C\left(1+\mathbb{E}\left[|X_0|^2\right]\right),$$

where $C = C(K_2, T)$ is a function of the Lipschitz constant and the final time.

The solution of (2.1), is in general a so-called Markov process. This property is satisfied by some processes considered in this thesis. Thus, we introduce it here with the motivation that it will be used in Section 3.1.1 and beyond. Let $X = (X_t)_{t\in[0,T]}$ solve (2.1). If X satisfies

$$\mathbb{P}(X_t \in E|\mathcal{F}_s) = \mathbb{P}(X_t \in E|\sigma(X_s)), \quad 0 \leq s \leq t,$$

where $E \in \mathcal{B}(\mathbb{R}^d)$ and $\mathcal{F}_s$ denotes the filtration generated by X up to time s, then X is said to be a Markov process. Recall that the filtration $\mathcal{F}_s$ at time s includes all information up to time s. Thus, this conditional probability means that given the current state X_s, the future is independent of its history.

To approximate the SDE (2.1), numerical schemes are used. More specifically, Euler–Maruyama are used in this thesis, which is a generalization of the Euler method for ordinary differential equations. Consider a partition in time

$$0 = t_0 < t_1 < \cdots < t_n < t_{n+1} < \cdots < t_N = T,$$

for the interval $[0,T]$. Then, the approximation $\hat{X}$ of X in (2.1) is the Markov process given by

$$\hat{X}_i = \hat{X}_{i-1} + b(t_{t_{i-1}}, \hat{X}_{i-1})(t_i - t_{i-1}) + \sigma(t_{i-1}, \hat{X}_{i-1})(W_i - W_{i-1}),$$

for $i = 1, \ldots, n$ and where $\hat{X}_{t_0} = x_0$. Then $\hat{X}$ is said to be the Euler–Maruyama approximation of X. We note that the increment of Brownian motion is, by definition, a normally distributed random variable $(W_i - W_{i-1}) \sim N(0, t_i - t_{i-1})$. We now present a theorem that states the convergence order of this numerical scheme, for the proof we refer to [13].

Theorem 7. *(Strong convergence of Euler–Maruyama). If the expectation of the difference in initial conditions between the numerical approximation $\hat{X}$ and the solution X can be bound by a constant times the time step i.e., for any $n \in \mathbb{N}$, it holds that*

$$\mathbb{E}\left[|\hat{X}_0 - X_0|^2\right] < C_1(t_n - t_{n-1}),$$

and further that the initial condition for the solution has finite second moment, that is

$$\mathbb{E}\left[|X_0|^2\right] \leq C_2,$$

then the following holds

$$\mathbb{E}\left[\max_{0 \leq t \leq T} |\hat{X}_t - X_t|^2\right] \leq C_3 h,$$

where $C_3 = C(L, C_1, C_2, T)$ and $C_1, C_2 \in \mathbb{R}$. Furthermore,

$$\mathbb{E}\left[|\hat{X}_n - X_n|\right] \leq Ch^{1/2}, \quad n = 0, \ldots, N,$$

where N is the number of time steps.

2.4 Machine learning

The solvers considered in this thesis are based on deep learning to circumvent the curse of dimensionality. Thus, we introduce the mathematical definition of fully connected neural networks. More specifically, the forward propagation of a network is used to make predictions. Moreover, the backward propagation is used to find the optimal parameters for the network.

Definition 12. *(Fully connected feedforward network) A layer of a fully connected feedforward neural network is the mapping defined by*

$$\mathfrak{Z}^i : x^i \mapsto \mathfrak{R}^i \circ \varrho^i(x^i), \tag{2.2}$$

where $\circ$ denotes the composition operator. Furthermore, $\varrho^i(x^i) \coloneqq Q^i x^i + \vartheta^i$ is the affine transformation for layer i, with the corresponding weight matrix $Q^i \in \mathbb{R}^{d_i \times \ell_i}$ and bias $\vartheta^i \in \mathbb{R}^{\ell_i}$. Here, d_i and ℓ_i denote the input and output dimension of layer i, and thus $x^i \in \mathbb{R}^{d_i}$ corresponds to the input data for that layer. Furthermore, $\mathfrak{R}^i : \mathbb{R}^\ell \to \mathbb{R}^\ell$ denotes the element-wise activation function. Then, $\mathfrak{Z}^i$ gives the

output of layer i, given some input x, and the entire network is the composition of all layers

$$\psi : x \mapsto \mathfrak{Z}^{n+1} \circ \cdots \circ \mathfrak{Z}^0(x),$$

where $n \in \mathbb{N}$ is the number of hidden layers, and $\mathfrak{Z}^0$ and $\mathfrak{Z}^{n+1}$ are input and output layers, respectively. Moreover, we let Θ denote the parameter space such that the concatenation of all parameters lies in the parameter space, that is, $\theta \coloneqq (Q^0, \dots, Q^{n+1}, \varrho^0, \dots, \varrho^{n+1}) \in \Theta$.

Remark 2. *Note that the definition of a layer* (2.2) *is not strict. Many augmentations can be made to individual layers and the overarching structure of the network to suit the current application. Definition 12 presents the simplest structure of a fully connected feedforward network. In this thesis in particular, the activation function for all hidden layers is the so-called ReLU function $\mathfrak{R}(x) = \max(0, x)$ and in the output layer the identity mapping $\mathfrak{R}^{n+1}(x) \coloneqq \iota(x) = x$ instead. This is due to the fact that the network is tasked with mapping to the entire set of real numbers $\mathbb{R}$ instead of only $\mathbb{R}_+$.*

The use of feedforward neural networks is motivated by the *Universal Approximation Theorem*, which states that any function $f \in L^p(E)$, for some arbitrary domain E, can be approximated arbitrarily well by neural networks, i.e. the class of neural networks is dense in $L^p(E)$ for $1 \leq p < \infty$. To understand this fully, we present the definition of a dense subset.

Definition 13. *A subset $S \subset L^p(E)$ for $1 \leq p < \infty$ is said to be dense if for an arbitrary function $f \in L^p(E)$ and $\varepsilon > 0$ there exists a function $g \in S$ such that*

$$\|f - g\|_{L^p(E)} < \varepsilon.$$

That is, given a sequence of a family of neural networks $\{\psi^n\}_{n=1}^{\infty}$ it can be shown that $\psi^n \to f$ as $n \to \infty$. We now state the Universal Approximation Theorem, in [14].

Theorem 8. *(Universal Approximation Theorem) For $p \in [1, \infty)$ the vector space containing every fully connected ReLU feedforward neural network, ψ, of width $h \in \mathbb{N}$ is dense in $L^p\left(\mathbb{R}^d; \mathbb{R}^\ell\right)$ if and only if $h \geq \max\{d + 1, \ell\}$.*

The motivation for including the Universal Approximation Theorem is to say that the class of feedforward neural networks is dense in $L^p(\mathbb{R}^d; \mathbb{R}^\ell)$, meaning that any function $f \in L^p(\mathbb{R}^d, \mathbb{R}^\ell)$ can be approximated arbitrarily well by neural networks. This is especially useful since there are many sources of numerical errors for the algorithm. These include the time discretization and the optimization error from the neural network. However, since any functions in $L^p(\mathbb{R}^d; \mathbb{R}^\ell)$ can be approximated arbitrarily well, the remaining error can attributed to the time discretization of the numerical scheme, i.e., Euler–Maruyama. That is, given that the global minimum for the neural network has been found. In practice, this translates to only a small error derived from the neural network approximation in comparison to that of the

time discretization. However, the theorem presumes an infinite number of hidden layers. Therefore, a larger error will be procured when using finite number of hidden layers.

What remains is to specifically present the network structure used in this thesis. Unless otherwise specified networks with the structure

$$\psi : x \mapsto \iota \circ \varrho^4 \circ \mathfrak{R} \circ \varrho^3 \circ \mathfrak{R} \circ \varrho^2 \circ \mathfrak{R} \circ \varrho^1 \circ \mathfrak{R} \circ \varrho^0(x). \tag{2.3}$$

The network has three sizes to be determined, input size d, size of hidden layers h and output size ℓ. Then $W^1 \in \mathbb{R}^{h\times d}$, $W^2 \in \mathbb{R}^{h\times h}$, $W^3 \in \mathbb{R}^{\ell\times h}$ and $\theta^1 \in \mathbb{R}^d$, $\theta^2 \in \mathbb{R}^h$, $\theta^3 \in \mathbb{R}^\ell$ and thus $\theta = (W^0, W^1, W^2, \vartheta^0, \vartheta^1, \vartheta^2) \in \Theta$.

Remark 3. *In the networks used throughout the thesis, the hidden layers have been chosen to have the same width. However, this is not a general restriction. Furthermore, the size of a layer and width are used interchangeably.*

To optimize the performance of the network, one seeks the parameters θ such that

$$\theta^* = \arg\min_{\theta\in\Theta} \frac{1}{N} \sum_{j=1}^{N} \mathcal{L}(\psi(x_j), y_j), \tag{2.4}$$

for some distance measure $\mathcal{L} : \mathbb{R}^d \times \mathbb{R}^\ell \to \mathbb{R}$ often called the *loss function*. Further, $x_n \in \mathbb{R}^d$ denotes the input data and $y_n \in \mathbb{R}^\ell$ the so-called ground truth vector. The choice of loss function is non-trivial and highly dependent on the application. The choice of the loss function is discussed in Section 4.1. To find the optimal parameters, an optimization algorithm such as mini-batch gradient descent is used. Instead of summing over the entire dataset as in (2.4), small chunks of data, so-called batches, is considered. For algorithms of this sort, the concept of backpropagation is used. This involves finding the partial derivative of the loss function with respect to a specific parameter $q_{ij}^k \in Q^k$ or ϑ^i and taking a step η in the direction of the negative partial derivative with respect to the parameter being updated, where η is the learning rate. Choosing the learning rate is non-trivial. In certain applications, a fixed learning rate is sufficient, while in others the learning rate scheduling is used. In this thesis, we restrict ourselves to a fixed learning rate. The partial derivative of the loss function with respect to a general weight is given by applying the chain rule to a network ψ with n layers and its set of activation functions $\mathfrak{R} \coloneqq (\mathfrak{R}^{n+1}, \dots, \mathfrak{R}^0)$,

that is,

$$\frac{\partial}{\partial Q^k}\mathcal{L}(\psi(x_m), y_m) = \frac{\partial}{\partial \psi(x_m)}\mathcal{L}(\psi(x_m), y_m) \prod_{j=k+1}^{n}\left(\frac{\partial}{\partial \varrho^j(\mathfrak{Z}^{j-1}(x_m))}\mathfrak{R}^j(\varrho^j(\mathfrak{Z}^{j-1}(x_m)))\frac{\partial}{\partial \mathfrak{Z}^{j-1}}\varrho^j(\mathfrak{Z}^{j-1}(x_m))\right) \frac{\partial}{\partial \varrho^k(x_m)}\mathfrak{R}^k(\varrho^k(x_m))\frac{\partial}{\partial Q^k}\varrho^k(x_m),$$

$$\frac{\partial}{\partial \vartheta^k}\mathcal{L}(\psi(x_m), y_m) = \frac{\partial}{\partial \psi(x_m)}\mathcal{L}(\psi(x_m), y_m) \prod_{j=k+1}^{n}\left(\frac{\partial}{\partial \varrho^j(\mathfrak{Z}^{j-1}(x_m))}\mathfrak{R}^j(\varrho^j(\mathfrak{Z}^{j-1}(x_m)))\frac{\partial}{\partial \mathfrak{Z}^{j-1}}\varrho^j(\mathfrak{Z}^{j-1}(x_m))\right) \frac{\partial}{\partial \varrho^k(x_m)}\mathfrak{R}^k(\varrho^k(x_m))\frac{\partial}{\partial \vartheta^k}\varrho^k(x_m),$$

for $0 \leq k \leq n-1$. Then, the parameters are updated by

$$Q^k \leftarrow Q^k + \eta\partial_{Q^k}\mathcal{L}(\psi(x_m), y_m),$$
$$\vartheta^k \leftarrow \vartheta^k - \eta\partial_{\vartheta^k}\mathcal{L}(\psi(x_m), y_m).$$

3

Stochastic control theory

This chapter presents relevant theory regarding stochastic control theory needed to grasp the algorithms examined and developed in the thesis. This includes the single agent setting and its generalization to multiple agents where stochastic differential games are presented. Key results such as existence and uniqueness are presented for each relevant equation.

3.1 Stochastic optimal control

In this section, the topic of stochastic optimal control is introduced, that is, the single agent setting. First, the problem formulation is presented. The goal is to control an SDE governing the dynamics of a system in such a way that an associated cost functional is minimized. Thereafter, the derivation of the HJB equation starting from a stochastic control problem is presented. Finally, the reformulation of the HJB equation is stated in terms of its corresponding FBSDE, which allows the problem to be solved while circumventing the curse of dimensionality. The material throughout this chapter is based on [15].

3.1.1 Problem formulation

Consider a stochastic control problem defined on the filtered probability space $(\Omega, \mathcal{F}, (\mathcal{F}_t)_{t\in[0,T]}, \mathbb{P})$ for some terminal time $T > 0$. The filtration $(\mathcal{F}_t)_{t\in[0,T]}$ is generated by the k-dimensional standard Brownian motion $W = (W_t)_{t\in[0,T]}$ and completed by the $\mathbb{P}$-null sets of Ω. Then, the controlled state process $X^u = (X_t^u)_{t\in[0,T]}$ is governed by the forward SDE

$$\mathrm{d}X_t^u = b(t, X_t^u, u_t)\,\mathrm{d}t + \sigma(t, X_t^u)\,\mathrm{d}W_t, \quad X_0^u = x_0, \tag{3.1}$$

where $b : [0,T] \times \mathbb{R}^d \times \mathbb{R}^{d\times\ell} \to \mathbb{R}^d$ and $\sigma : [0,T] \times \mathbb{R}^d \to \mathbb{R}^{d\times k}$ are deterministic drift and diffusion coefficients, respectively. Moreover, the state $X^u = (X_t^u)_{t\in[0,T]}$, with initial condition $x_0 \in \mathbb{R}^d$ such that $X_0^u = x_0$ $\mathbb{P}$-a.s., is controlled by the control policy $u = (u_t)_{t\in[0,T]}$ which is assumed to be an $\mathcal{F}_t$-adapted process taking values in a set $\mathbb{U} \subseteq \mathbb{R}^\ell$, which we call the control space.

Remark 4. *Note that the diffusion coefficient σ in* (3.1) *does not take the control process u_t as an input. In financial applications, it is common that the diffusion also depends on the control. However, this thesis is limited to control problems for which the diffusion does not depend on the control process.*

The controls used throughout the thesis are required to be *admissible*. For this concept, we first introduce the definition of a progressively measurable process. Subsequently, the definition of an admissible control process is presented.

Definition 14. *The control process is said to be progressively measurable if for every* $s \in [t, T]$ *the mapping* $u : [t, T] \times \Omega \to \mathbb{U}$ *is* $\mathcal{B}([t, s]) \otimes \mathcal{F}_t$*-measurable.*

Definition 15. *A progressively measurable control process* $u = (u_t)_{t\in[0,T]}$ *is said to be admissible, and we write* $u \in \mathcal{U}$*, if it fulfills the condition of having finite moments, i.e.,*

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

To provide the reader with intuition regarding the definitions introduced above and to facilitate understanding, an example is presented below. The example illustrates the concept of an admissible control policy in a simple context.

Example 2. *Consider a car with the goal of reaching the top of a hill. The control process can then be viewed as the possible ways of controlling the car. This includes, for instance, breaking, applying throttle, steering left, and steering right. However, a car can not take on arbitrarily large values for these controls. We thus define admissible controls by specifying the maximum steering angle and the maximum applied force. Anything beyond these limits is not admissible.*

Moreover, the drift and diffusion coefficient functions b and σ are assumed to be measurable Lipschitz continuous functions, ensuring the existence and uniqueness of strong solutions to (3.1). That is, there exists constants $C_b, C_\sigma > 0$ such that

$$\begin{aligned} |b(t, x, u) - b(t, x', u')| &\leq C_b(|x - x'| + |u - u'|), \\ |\sigma(t, x) - \sigma(t, x')| &\leq C_\sigma |x - x'|, \end{aligned}$$

for all $(t, x, x', u, u') \in [0, T] \times \mathbb{R}^d \times \mathbb{R}^d \times \mathbb{U} \times \mathbb{U}$, and $|\cdot|$ is the Frobenius norm. Furthermore, the drift and diffusion coefficient functions also satisfy a linear growth condition in space and uniform growth in time, i.e., there exists some constant C' such that

$$|b(t, x, u)| + |\sigma(t, x)| \leq C'(1 + |x|), \qquad (t, x, u) \in [0, T] \times \mathbb{R}^d \times \mathbb{U}.$$

For a stochastic control problem, a cost functional is introduced in order to describe the cost of the agent over the time interval $0 \leq t \leq T$. Here, the term *agent* refers to the state whose dynamics are described by the SDE (3.1), e.g., a drone, a car, etc. The cost functional is then defined as

$$J^u(t, x) \coloneqq \mathbb{E}\left[\int_t^T f(s, X_s^u, u_s) \, \mathrm{d}s + g(X_T^u) \Big| X_t^u = x\right], \tag{3.2}$$

where $f : [0, T] \times \mathbb{R}^d \times \mathbb{U} \to \mathbb{R}$ is a function describing the running cost incurred by the agent, and $g : \mathbb{R}^d \to \mathbb{R}$ is a function describing the terminal cost. Thus, the aim

is to find a control policy u that minimizes the cost functional and thereby optimally regulates the state process X_t^u. To provide intuition regarding the stochastic control problem another example is presented below.

Example 3. *Consider an autonomous car intending to reach some predetermined destination. The state process X_t^u describing the the car at time t, may include physical quantities such as its position and velocity. In contrast, the control policy u_t represents the inputs to the car, for instance, acceleration or steering in either direction. It is then natural to define the running cost as the fuel consumption and the terminal cost as the vehicles deviation from the desired location at the final time. The aim is thus, to find an optimal policy which minimizes the total cost. Thus, this means that minimal fuel usage is sought, given that the car reaches its intended destination.*

3.1.2 The Hamilton–Jacobi–Bellman equation

The derivation of the HJB equation is based on Bellman's dynamic programming principle. We start by defining the the value function by

$$V(t,x) \coloneqq \inf_{u \in \mathcal{U}} J^u(t,x), \quad (t,x) \in [0,T] \times \mathbb{R}^d,$$

where $J^u(t,x)$ is the cost functional defined in (3.2). Since the value function V describes the best possible outcome of the control problem, it is fundamental in dynamic programming and the derivation of the HJB equation.

We now provide an overview of the derivation that is to come. By applying Bellman's dynamic programming principle, one can derive that the value function satisfies a semi-linear parabolic partial differential equation, namely, the HJB equation. More precisely, the application of Itô's formula together with the principle of optimality leads to the HJB equation, which describes the value function. Furthermore, an optimal Markov control policy can be expressed in terms of the solution to the HJB equation and the spatial derivatives of that solution. Consequently, the solution of the HJB equation is equivalent to the control problem under some regularity assumptions. The assumptions are primary based on the smoothness of the solution to the HJB equation. If there exists a classical solution, it can be shown that it coincides with the value function for the control problem. These results are presented in the Verification theorem. However, we begin by stating Bellman's dynamic programming principle.

Theorem 9 (Bellman's dynamic programming principle)**.** *Let $t \in [0,T]$, $\tau \in [t,T]$, and $x \in \mathbb{R}^d$, then it holds*

$$V(t,x) = \inf_{u \in \mathcal{U}} \mathbb{E}\left[\int_t^\tau f(s, X_s^u, u_s)\, \mathrm{d}s + V(\tau, X_\tau^u) \middle| X_t^u = x \right]. \tag{3.3}$$

Proof. The claim is proved in [16]. □

Bellman's dynamic programming principle states that the optimal control can be determined by solving a sequence of smaller problems. In terms of the value function, an optimal control strategy for the entire time interval can be decomposed into first controlling optimally on an initial subinterval, and then continuing with the optimal strategy until the final time.

To derive the HJB equation, we begin with Bellman's dynamic programming principle (3.3). Let $v \in \mathcal{U}$ be an arbitrary admissible control and $\tau = t + h$ for some small $h > 0$. This gives

$$V(t,x) \leq \mathbb{E}\left[\int_t^{t+h} f(s, X_s^v, v)\,\mathrm{d}s + V(t+h, X_{t+h}^v)\Big| X_t^v = x\right]. \tag{3.4}$$

Next, subtract $V(t,x)$ from both sides of (3.4) and divide by h, giving

$$\begin{aligned}
0 &\leq \frac{1}{h}\mathbb{E}\left[\int_t^{t+h} f(s, X_s^v, v)\,\mathrm{d}s + V(t+h, X_{t+h}^v) - V(t,x)\Big| X_t^v = x\right] \\
&= \frac{1}{h}\mathbb{E}\left[\int_t^{t+h} f(s, X_s^v, v)\,\mathrm{d}s\Big| X_t^v = x\right] \\
&\qquad + \frac{1}{h}\mathbb{E}\left[V(t+h, X_{t+h}^v) - V(t,x)\big| X_t^v = x\right] \\
&= I_1 + I_2,
\end{aligned} \tag{3.5}$$

where the linearity of the expectation operator is used in the first equality. Next, let us study the limits of I_1 and I_2 as $h \downarrow 0$. For I_1, assume that the map

$$s \mapsto \mathbb{E}\left[f(s, X_s^v, v) | X_t^v = x\right]$$

is locally integrable in s. The next step requires Fubini's theorem (see Theorem 2), as well as the Lebesgue differentiation (see Theorem 1). Utilizing these two theorems yields

$$\lim_{h\downarrow 0} I_1 = \lim_{h\downarrow 0} \frac{1}{h}\int_t^{t+h} \mathbb{E}\left[f(s, X_s^v, v)\big| X_t^v = x\right]\,\mathrm{d}s = f(t,x,v). \tag{3.6}$$

For the second term I_2, applying Itô's formula (see Theorem 5), gives

$$\begin{aligned}
&\lim_{h\downarrow 0} I_2 \\
&= \lim_{h\downarrow 0} \frac{1}{h}\mathbb{E}\left[V(t+h, X_{t+h}^v) - V(t, X_t^v)\,\mathrm{d}s\Big| X_t^v = x\right] \\
&= \lim_{h\downarrow 0} \frac{1}{h}\mathbb{E}\left[\int_t^{t+h} (\partial_s + L^v)\, V(s, X_s^v)\,\mathrm{d}s + \int_t^{t+h} \nabla_x V(s, X_s^v)\sigma(s, X_s^v)\,\mathrm{d}W_s\Big| X_t^v = x\right] \\
&= \lim_{h\downarrow 0} \frac{1}{h}\mathbb{E}\left[\int_t^{t+h} (\partial_s + L^v)\, V(s, X_s^v)\,\mathrm{d}s\Big| X_t^v = x\right],
\end{aligned} \tag{3.7}$$

where the zero-mean property of the Itô integral is used in the last equality. Once again, by assuming the conditions for Fubini's theorem and Lebesgue's theorem of

differentiation are satisfied, it follows that

$$\lim_{h\downarrow 0} I_2 = \lim_{h\downarrow 0} \frac{1}{h} \int_t^{t+h} \mathbb{E}\left[(\partial_s + L^v)\, V(s, X_s^v) \middle| X_t^v = x \right] \, \mathrm{d}s = (\partial_t + L^v) V(t,x).$$

Inserting the limits obtained in (3.6), (3.7) into (3.5) we find, for every admissible control $v \in \mathcal{U}$, that

$$-\partial_t V(t,x) \leq L^v V(t,x) + f(t,x,v). \tag{3.8}$$

Recall that in (3.3) the value function is defined as an infimum over all admissible controls. In (3.4), an arbitrary control $v \in \mathcal{U}$ was considered, and the infimum is therefore removed. Thus, the equality was replaced by an inequality for the value function. By once again considering the infimum, (3.8) becomes

$$-\partial_t V(t,x) = \inf_{v\in\mathcal{U}} \left(L^v V(t, X_t^v) + f(t,x,v) \right). \tag{3.9}$$

The optimal policy is then obtained as

$$\pi(t,x) \in \arg\min_{v\in\mathcal{U}} \left(L^v V(t, X_t^v) + f(t,x,v) \right), \qquad (t,x) \in [0,T] \times \mathbb{R}^d, \tag{3.10}$$

where $\pi : [0,T] \times \mathbb{R}^d \to \mathcal{U}$ is a deterministic map. At each time t the control is obtained as $u_t = \pi(t, X_t^u)$, i.e., the decision only depends on the current time and current state. Hence, when this is valid, the control is called a *Markov control policy*. In this thesis we restrict ourselves to this case and we assume the set of minimizers above is nonempty. Under this assumption, we state a verification theorem in the next section. Since the diffusion term of the forward SDE, in this thesis, does not depend on the control u (3.9) is a second-order semi-linear PDE. By explicitly extracting the diffusion term from the operator L^v since it does not depend on the control, the HJB equation is expressed as

$$\partial_t V + \frac{1}{2} \operatorname{Tr}(\sigma\sigma^\top \Delta_x V) + \mathcal{H}(t,x,\nabla_x V) = 0, \quad (t,x) \in [0,T) \times \mathbb{R}^d, \tag{3.11a}$$

$$V(T,x) = g(x), \quad x \in \mathbb{R}^d. \tag{3.11b}$$

Note that the space-time dependence (t,x) has been excluded in the equation above for brevity. However, if the functions and coefficients have other dependencies, we specify that explicitly. Furthermore, Δ_x denotes the Hessian matrix which is the second order spatial derivative and the Hamiltonian $\mathcal{H}$ is defined as

$$\mathcal{H}(t,x,p) \coloneqq \inf_{v\in\mathcal{U}} \left(\left\langle b(t,x,v), p \right\rangle + f(t,x,v) \right), \qquad (t,x) \in [0,T] \times \mathbb{R}^d. \tag{3.12}$$

Here, $p \in \mathbb{R}^d$ is a placeholder for the spatial derivative of the value function. The terminal condition (3.11b) is derived by considering the value function (3.2) at the terminal time T, that is,

$$V(T,x) = \inf_{u\in\mathcal{U}} J^u(T,x) = \inf_{u\in\mathcal{U}} \mathbb{E}\left[g(X_T^u) | X_T^u = x \right] = g(x).$$

Remark 5. *Note that in* (3.10)*,* $\pi(t,x)$ *is an element in the set of minimizers of the Hamiltonian* $\mathcal{H}$*. Consequently, uniqueness is not guaranteed and is generally rare. However, for many important applications, including the one considered in this thesis, it is possible to derive an explicit function, denoted* κ*, which minimizes the Hamiltonian. This function is obtained by analytically solving the minimization problem associated with* $\mathcal{H}$*, and the optimal Markov control policy is given by* $\pi(t,x) = \kappa(t,x,\nabla_x V)$*.*

The HJB equation (3.11) is generally challenging to analyze due to its nonlinear nature. Classical, smooth solutions may not exist, and the concept of viscosity solutions needs to be considered. However, the HJB equation and the forward SDE have strong solutions under certain regularity conditions. That is when the system experiences sufficient noise in all directions of the state space. The intuition is that the diffusion term introduces a local averaging effect and making it smoother. This effect is characterized by the *uniformly parabolic condition* presented below.

Definition 16. *Let* $a(t,x) \coloneqq \sigma(t,x)\sigma(t,x)^\top$*, then the HJB equation is said to be uniformly parabolic if there exists a constant* $\hat{C} > 0$ *such that*

$$\sum_{i,j=1}^{n} a_{i,j}(t,x)\xi_i\xi_j \geq \hat{C}\|\xi\|^2, \qquad (t,x) \in [0,T] \times \mathbb{R}^d,\ \xi \in \mathbb{R}^d. \tag{3.13}$$

The intuition for this condition is that the smallest eigenvalue of $a(t,x)$ is uniformly bounded below by $\hat{C}$, which implies that the noise in the systems acts in all directions of the state space. When the problem satisfies (3.13), one typically refers to the problem as *non-degenerate*. This is one of the conditions for a classical solution of (3.11) to exist. Before stating the additional conditions required for the existence and uniqueness of the solution, we first introduce some notations.

Let $\ell, k \in \mathbb{N} \cup \{0\}$, and $n \in \mathbb{N}$. Then, for $\gamma = (\gamma_1, \ldots, \gamma_n) \in (\mathbb{N} \cup \{0\})^n$ set

$$|\gamma| \coloneqq \gamma_1 + \ldots + \gamma_n, \qquad \partial^\gamma \coloneqq \frac{\partial^{|\gamma|}}{\partial^{\gamma_1} \cdots \partial^{\gamma_n}}.$$

Then, we define

$$C^{\ell,k} \coloneqq \Big\{\phi \in C \,:\, \partial_t^j \partial_x^\gamma \phi \in C \text{ for } j = 0,1,\ldots,\ell,\ |\gamma| \leq k\Big\}, \tag{3.14}$$

where C is the space of all continuous functions. Thus, $C^{\ell,k}$ contains those functions that are ℓ-times continuously differentiable in the time variable t and k-times differentiable in space, with all mixed derivatives up to these order being continuous. Moreover, we define the bounded subspace of (3.14), that is,

$$C_{\mathrm{b}}^{\ell,k} \coloneqq \Big\{\phi \in C^{\ell,k} \,:\, \sup_{(t,x)} \|\partial_t^j \partial_x^\gamma \phi\| \leq M \ \text{ for } j = 0,1,\ldots,\ell,\ |\gamma| \leq k\Big\},$$

for some $M \in \mathbb{R}$. The space $C_{\mathrm{b}}^{\ell,k}$ includes all functions whose derivatives are bounded. To make the notation fully explicit, let $\mathbb{V}$ be a finite dimensional real

vector space, and let $\phi : [0,T] \times \mathbb{R}^d \to \mathbb{V}$. Suppose ϕ is ℓ-times continuously differentiable in time, k-times differentiable in space, and all mixed derivatives up to order (ℓ, k) are uniformly bounded. In that case, we simply write $\phi \in C_{\mathrm{b}}^{\ell,k}([0,T] \times \mathbb{R}^d; \mathbb{V})$. Further, let $C^k\big(\mathbb{R}^n; \mathbb{V}\big)$ and $C_{\mathrm{b}}^k(\mathbb{R}^d; \mathbb{V})$ denote the space of all time-invariant functions under the conditions stated above. We can now state the existence and uniqueness theorem for solutions to the HJB equation (3.11).

Theorem 10 (Existence and uniqueness of classical solutions of the HJB equation)**.** *Let $T > 0$ and let the set $\mathcal{U}$ be compact. Suppose*

(i) The diffusion matrix $a(t,x) \coloneqq \sigma(t,x)\sigma(t,x)^\top$ is uniformly parabolic, i.e, (3.13) *holds.*

(ii) The diffusion matrix a, the drift coefficient b, and running cost f satisfy

$$a \in C_{\mathrm{b}}^{1,2}([0,T] \times \mathbb{R}^d; \mathbb{R}^{k\times k}), \quad b \in C_{\mathrm{b}}^{1,2}([0,T] \times \mathbb{R}^d \times \mathbb{U}; \mathbb{R}^d),$$
$$f \in C_{\mathrm{b}}^{1,2}([0,T] \times \mathbb{R}^d \times \mathbb{U}; \mathbb{R}).$$

(iii) The terminal cost g has three continuous and bounded derivatives in space, that is, $g \in C_{\mathrm{b}}^3(\mathbb{R}^d; \mathbb{R})$.

Then there exists a unique solution $\Phi \in C_{\mathrm{b}}^{1,2}([0,T] \times \mathbb{R}^d; \mathbb{R})$ of the HJB equation (3.11)*.*

Proof. The claim is proved in [17]. □

We present a central verification theorem establishing that the cost functional of the control problem coincides with the solution of its associated HJB equation. Therefore, solving the HJB equation also solves the control problem, and Bellman's dynamic programming principle holds.

Theorem 11 (Verification theorem)**.** *Let $\Phi \in C^{1,2}([0,T] \times \mathbb{R}^d; \mathbb{R})$ be a classical solution of the HJB equation* (3.11)*. Then for every admissible control $u \in \mathcal{U}$ and all $(t,x) \in [0,T) \times \mathbb{R}^d$,*

$$\Phi(t,x) \leq J^u(t,x).$$

Further, if there also exists an optimal Markov policy π that is locally Lipschitz continuous, has linear growth in x and uniform growth in t and satisfies

$$\pi(t,x) \in \operatorname*{arg\,min}_{u\in\mathcal{U}} \big\{ b(t,x,u)\nabla_x \Phi(t,x) + f(t,x,u) \big\}.$$

Then

(i) For all $(t,x) \in [0,T] \times \mathbb{R}^d$ it holds that

$$\Phi(t,x) = J^\pi(t,x),$$

and hence, $\Phi = V$.

(ii) Bellman's dynamic programming principle holds.

Proof. We split the proof into two parts. First, we establish the dynamic programming principle for any admissible control. Then, we consider the optimal Markov policy π to show that the solution of the HJB equation coincides with the value function, which proves the claim. Let $(t, x) \in [0, T) \times \mathbb{R}^d$, and consider an arbritary control $u \in \mathcal{U}$. Further, let $t < \tau \leq T$. Since $\Phi \in C^{1,2}([0, T] \times \mathbb{R}^d; \mathbb{R})$, Itô's formula gives

$$\begin{aligned} &\Phi(\tau, X_\tau^u) - \Phi(t, X_t^u) \\ &\qquad = \int_t^\tau (\partial_s + L^u)\, \Phi(s, X_s^u)\, \mathrm{d}s + \int_t^\tau \nabla_x \Phi(s, X_s^u) \sigma(s, X_s^u)\, \mathrm{d}W_s. \end{aligned} \tag{3.15}$$

Since Φ solves the HJB equation, it follows that

$$-\partial_t \Phi(t, x) = \inf_{v \in \mathcal{U}} \left\{ L^v \Phi(t, X_t^v) + f(t, x, v) \right\} \leq L^u \Phi(t, x) + f(t, x, u),$$

so that

$$(\partial_t + L^u)\Phi(t, x) \geq -f(t, x, u). \tag{3.16}$$

Insertion of (3.16) in (3.15) gives

$$\begin{aligned} &\Phi(\tau, X_\tau^u) - \Phi(t, X_t^u) \\ &\qquad \geq \int_t^\tau \nabla_x \Phi(s, X_s^u) \sigma(s, X_s^u)\, \mathrm{d}W_s - \int_t^\tau f(s, X_s^u, u_s)\, \mathrm{d}s. \end{aligned} \tag{3.17}$$

Taking the conditional expectation of (3.17) on both sides with respect to X_t^u, and noting that the stochastic integral vanishes due to the martingale property resulting in zero-mean. By rearranging terms, we obtain the following inequality

$$\Phi(t, x) \leq \mathbb{E}\left[\int_t^\tau f(s, X_s^u, u_s)\, \mathrm{d}s + \Phi(\tau, X_\tau^u) \middle| X_t^u = x \right].$$

By letting $\tau = T$ and using the terminal condition $\Phi(T, \cdot) = g$, it gives

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

This proves the sub-optimality claim and the dynamic programming principle for Φ under any $u \in \mathcal{U}$. Now, we assume that there exists an optimal Markov control policy $\pi : [0, T] \times \mathbb{R}^d \to \mathcal{U}$. By repeating the same steps as above, (3.16) instead becomes

$$(\partial_t + L^\pi)\Phi(t, x) = -f(t, x, \pi(t, x)).$$

Repeating the Itô and expectation argument, and by letting $\tau = T$ and inserting the terminal condition $\Phi(T, \cdot) = g$, we get that

$$\Phi(t, x) = \mathbb{E}\left[\int_t^T f(s, X_s^\pi, \pi(s, X_s^\pi))\, \mathrm{d}s + g(X_T^\pi) \middle| X_t^\pi = x \right] = J^\pi(t, x).$$

Hence, $\Phi = J^\pi = V$. This completes the proof.

□

3.1.3 Reformulation to FBSDE

This subsection presents the reformulation of the HJB equation into the FBSDE. To recap, we began with a forward SDE that describes the system's dynamics, where the state is controlled by the process u. Thus, we want to determine an optimal control policy u^* that regulates the state by minimizing a cost function. In Section 3.1.2, the cost function was reformulated as its corresponding semi-linear PDE, namely the HJB equation. Next, we reformulate the HJB equation into a backward stochastic differential equation (BSDE). Together with the forward SDE, a coupled FBSDE system is obtained. In short, the derivation is done by applying Itô's formula to the solution of the HJB equation and introducing variables representing both the solution and its spatial gradient.

Under the regularity conditions stated in Section 3.1.2, it has been shown that the HJB equation solves the control problem. This gives rise to the motivation for considering a stochastic representation of the HJB equation. Classical numerical methods for solving PDEs, such as FEM, FVM, or FDM, do not scale efficiently in high dimensions. Thus, this thesis will focus on a stochastic representation of the HJB equation, where neural networks are utilized to solve the coupled FBSDE. The use of neural networks helps to avoid the curse of dimensionality. This thesis demonstrates that the proposed method performs effectively even for high-dimensional systems.

Next, we will derive the FBSDE system. Assume that the existence and uniqueness theorem (Theorem 10) and the Verification theorem (Theorem 11) are satisfied. Then there exists a unique solution $\Phi \in C_{\mathrm{b}}^{1,2}$ that solves the HJB equation (3.11). In addition, we assume that an optimal Markov policy π exists which minimizes the Hamiltonian (3.10). Applying Itô's formula to Φ, yields

$$\Phi(t, X_t^\pi) = \Phi(0, X_0^\pi) + \int_0^t (\partial_s + L^\pi)\Phi(s, X_s^\pi)\,\mathrm{d}s + \int_0^t \nabla_x \Phi^\top(s, X_s^\pi)\sigma(s, X_s^\pi)\,\mathrm{d}W_s. \tag{3.18}$$

By once again considering (3.9), it follows that

$$-\partial_t \Phi(t,x) = \inf_{v\in\mathcal{U}} \left\{ L^v \Phi(t, X_t^v) + f(t,x,v) \right\} = L^\pi \Phi(t, X_t^\pi) + f(t,x,\pi(t,x)),$$

where the fact that π attains the infimum in the HJB equation is used in the last equality. Insert the above identity into (3.18) to obtain

$$\begin{aligned}\Phi(t, X_t^\pi) = \Phi(0, X_0^\pi) - \int_0^t f(s, X_s^\pi, \pi(s, X_s^\pi))\,\mathrm{d}s + \\ \int_0^t \nabla_x \Phi^\top(s, X_s^\pi)\sigma(s, X_s^\pi)\,\mathrm{d}W_s.\end{aligned} \tag{3.19}$$

Define $Y_t \coloneqq \Phi(t, X_t^\pi)$ and $Z_t \coloneqq \nabla_x \Phi(t, X_t^\pi)$. Further, since the state X^π is contolled optimally, we define $X \coloneqq X^\pi$ for brevity. With this notation, the BSDE (3.19) is written as

$$Y_t = Y_0 - \int_0^t f(s, X_s, \pi(s, X_s))\,\mathrm{d}s + \int_0^t Z_s^\top \sigma(s, X_s)\,\mathrm{d}W_s.$$

Since $Y = (Y_t)_{t\in[0,T]}$ represents the solution to the HJB equation, its terminal value is known and given as $Y_T = V(T, X_T) = g(X_T)$. In contrast, the initial value Y_0 is not known a priori. Therefore, we integrate backwards in time instead, meaning applying Itô's formula with the times T and t, to (3.11). Thereafter inserting the above definitions of Y_t and Z_t and solving for Y_t yields

$$Y_t = g(X_T) + \int_t^T f(s, X_s, \pi(s, X_s))\,\mathrm{d}s - \int_t^T Z_s^\top \sigma(s, X_s)\,\mathrm{d}W_s. \tag{3.20}$$

The forward SDE together with (3.20) now gives the system of equations on FBSDE form

$$\begin{cases} X_t = x_0 + \int_0^t b(s, X_s, \pi(s, X_s))\,\mathrm{d}s + \int_0^t \sigma(s, X_s)\,\mathrm{d}W_s, \\ Y_t = g(X_T) + \int_t^T f(s, X_s, \pi(s, X_s))\,\mathrm{d}s - \int_t^T Z_s^\top \sigma(s, X_s)\,\mathrm{d}W_s. \end{cases}$$

For the problems considered in this thesis, the optimal control policy u^* will be obtained by

$$u_t^* = \pi(t, X_t) = \kappa(t, X_t, \nabla_x \Phi(t, X_t)),$$

where $\kappa(t, X_t, \nabla_x \Phi(t, X_t)$ is an explicit expression for the optimal control policy, which minimizes the Hamiltonian (3.12). Furthermore, the control appears only in the drift term, so the feedback map κ depends solely on the gradient $\nabla_x V$. If the control was also included in the diffusion coefficient, κ would also depend on $\Delta_x V$. If instead the definition $Z_t = \nabla_x \Phi(t, x)^\top \sigma(t, x)$ is used, which is a common choice in literature, see e.g., [5, 4, 6], then the functions b and f need to be rewritten. This is done by the following identity

$$\begin{aligned} \nabla_x \Phi(t, X_t) &= \nabla_x \Phi(t, X_t)^\top \sigma(t, X_t)\sigma(t, X_t)^\top \left(\sigma(t, X_t)\sigma(t, X_t)^\top\right)^{-1} \\ &= Z_t \sigma(t, X_t)^\top \left(\sigma(t, X_t)\sigma(t, X_t)^\top\right)^{-1}. \end{aligned}$$

This yields

$$\begin{aligned} \bar{b}(t, x, z) &= b\Big(t, x, \kappa\Big(t, x, z\sigma(t, x)^\top \left(\sigma(t, x)\sigma(t, x)^\top\right)^{-1}\Big)\Big), \\ \bar{f}(t, x, z) &= f\Big(t, x, \kappa\Big(t, x, z\sigma(t, x)^\top \left(\sigma(t, x)\sigma(t, x)^\top\right)^{-1}\Big)\Big). \end{aligned}$$

Which in turn yields the system of equations

$$X_t = x_0 + \int_0^t \bar{b}(s, X_s, Z_s)\,\mathrm{d}s + \int_0^t \sigma(s, X_s)\,\mathrm{d}W_s, \tag{3.21a}$$

$$Y_t = g(X_T) + \int_t^T \bar{f}(s, X_s, Z_s)\,\mathrm{d}s - \int_t^T Z_s\,\mathrm{d}W_s, \tag{3.21b}$$

with the solution triple $(X_t, Y_t, Z_t) \in \mathbb{S}_T^2(\mathbb{R}^d) \times \mathbb{S}_T^2(\mathbb{R}) \times \mathbb{H}_T^2(\mathbb{R}^d)$ which is said to be an adapted solution if it satisfies the equation $\mathbb{P}$-almost surley. Here, the forward

SDE (3.21a) describes the system's dynamics while the backward equation (3.21b) describes the cost. This is evident due to the backward equation simply being a reformulation of the initial cost functional (3.2). Furthermore, the backward equation also represents the solution to the HJB equation (3.11) with the same motivation.

Remark 6. *Note that the FBSDE*

$$\begin{cases} X_t = x_0 + \int_0^t \bar{b}(s, X_s, Z_s)\,\mathrm{d}s + \int_0^t \sigma(s, X_s)\,\mathrm{d}W_s, \\ Y_t = Y_0 - \int_0^t \bar{f}(s, X_s, Z_s)\,\mathrm{d}s + \int_0^t Z_s\,\mathrm{d}W_s, \\ Y_T = g(X_T), \end{cases}$$

is eqvivalent to (3.21)*. The backward equation in this reformulation, in contrast to* (3.21)*, is now written forward in time where the unknown initial value Y_0 replaces the terminal condition and the sign changes to the coefficients. While the equality $Y_T = g(X_T)$ is imposed at $t = T$.*

The following theorem, from [18], is introduced to guarantee that a unique solution exists to (3.21). It outlines the conditions for the drift, diffusion, running cost, and terminal condition for which a unique solution can be guaranteed.

Theorem 12 (Existence and uniqueness of solution of FBSDE)**.** *Assume that the uniform Lipschitz condition on the drift and diffusion of the forward equation presented in Section 3.1.1 holds. Then, under the following additional conditions*

(i) If there exists constants $C_{\bar{f}}, C_g > 0$ such that

$$|\bar{f}(t,x,z) - \bar{f}(t,x',z')| \leq C_{\bar{f}}(|x-x'| + |z-z']),$$
$$|g(t,x) - g(t,x')| \leq C_g|x-x'|,$$

for all $(t,x,x',z,z') \in [0,T] \times \mathbb{R}^d \times \mathbb{R}^d \times \mathbb{R}^d \times \mathbb{R}^d$.

(ii) For the Lipschitz constants for the diffusion process C_σ and the terminal condition C_g it holds that

$$C_\sigma C_g < 1.$$

(iii) The functions $g(\cdot,0) \in \mathbb{L}^2_T(\mathbb{R})$, $\bar{f}(\cdot,0,0) \in \mathbb{H}^2_T(\mathbb{R})$, $\bar{b}(\cdot,0,0) \in \mathbb{H}^2_T(\mathbb{R}^d)$ and $\sigma(\cdot,0) \in \mathbb{L}^2_T(\mathbb{R}^{k\times d})$.

Then there exists a constant $C' > 0$, independent of all Lipschitz constants, such that for $T < C'$ there exists a unique solution tripe $(X_t, Y_t, Z_t) \in \mathbb{S}^2_T(\mathbb{R}^d) \times \mathbb{S}^2_T(\mathbb{R}) \times \mathbb{H}^2_T(\mathbb{R}^d)$ to the FBSDE (3.21) *in the interval $t \in [0,T]$. Further, it holds that the solution is bounded by*

$$\left(\|X_t\|_{\mathbb{S}^2_T} + \|Y_t\|_{\mathbb{S}^2_T} + \|Z_t)\|_{\mathbb{H}^2_T}\right)^2 \leq C\mathbb{E}\left[\left(\int_0^T |b(t,0,0)| + |f(t,0,0)|\,\mathrm{d}t\right)^2 + \int_0^T |\sigma(t,0)|^2\,\mathrm{d}t + |g(0)|^2 + |x|\right],$$

for some constant $C \in \mathbb{R}$.

3.2 Stochastic differential games

In this section, an extension of Section 3.1 is presented in order to account for N agents. We begin by introducing the notation for N players, followed by the problem formulation and derivation of the corresponding HJB equation for the multi-agent system. Finally, the resulting system of FBSDEs is presented.

The motivation for this extension is that the topic of finite player stochastic differential games is a flourishing field. There exist techniques to solve these finite player stochastic differential games, see [5]. These build upon the method presented in [4]. It has been shown, as in e.g., [6, 19], that this method does not converge for various problems, often, but not always, related to stochastic control. Although the topic of when the deep BSDE method fails is far from trivial and yet not fully known, a robust version was derived in [6], which converges for problems stemming from stochastic control. However, this method only applies to single-agent settings, creating a clear research gap for extending the robust solver to finite player games.

3.2.1 Problem formulation

We now consider an N-player stochastic control problem for the players in the set $\mathcal{I} \coloneqq \{1, \dots, N\}$, which is the generalization of the problem presented in Section 3.1.1. The filtered probability space $(\Omega, \mathcal{F}, (\mathcal{F})_{t\in[0,T]}, \mathbb{P})$ is defined as in Section 3.1.1. Then, the generalized state process is given by

$$\mathbf{X}_t^{\mathbf{u}} \coloneqq (X_t^{1,\mathbf{u}}, \dots, X_t^{N,\mathbf{u}})^\top,$$

which is a vector concatenation of the state processes of all players $i \in \mathcal{I}$. The individual state processes are defined as in Section 3.1.1, where $X^{i,\mathbf{u}} \coloneqq (X_t^{i,\mathbf{u}})_{t\in[0,T]}$ is state process for player $i \in \mathcal{I}$. Then, the state process of all players is defined as a family of stochastic processes $\mathbf{X}^{\mathbf{u}} \coloneqq (\mathbf{X}_t^u)_{t\in[0,T]}$. Moreover, $\mathbf{u}$ is a vector concatenation, as above, of all the admissible control policies $(u_t^i)_{t\in[0,T]}$ for all players, which belongs to the product σ-algebra $\mathcal{U} \coloneqq \otimes_{i=1}^N \mathcal{U}^i$, where $\mathcal{U}^i$ is the space of admissible controls for player i, defined as in Section 3.1.1. Here, a processes $u^i \in \mathcal{U}^i$ takes on values in $\mathbb{U}^i$, and we define $\mathbb{U} \coloneqq \mathbb{U}^1 \times \cdots \times \mathbb{U}^N$ and note that a process $\boldsymbol{u} \in \mathcal{U}$ takes on values in $\mathbb{U} \subset \mathbb{R}^\ell$. Then, the generalized state process is governed by the SDE

$$\mathrm{d}\mathbf{X}_t^{\mathbf{u}} = \mathbf{b}(t, \mathbf{X}_t^{\mathbf{u}}, \mathbf{u}_t)\,\mathrm{d}t + \Sigma(t, \mathbf{X}_t^{\mathbf{u}})\,\mathrm{d}W_t, \quad \mathbf{X}_0^{\mathbf{u}} = \mathbf{x}_0, \tag{3.22}$$

where the drift and diffusion coefficients $\mathbf{b} : [0,T] \times \mathbb{R}^{Nd} \times \mathbb{U} \times \mathcal{I} \to \mathbb{R}^{Nd}$ and $\Sigma : [0,T] \times \mathbb{R}^{Nd} \times \mathcal{I} \to \mathbb{R}^{Nd\times k}$ are concatenations of the drift and diffusion coefficients for all players respectively. In detail, these coefficients are defined by

$$\mathbf{b} \coloneqq (b^1, \dots, b^N)^\top, \quad \Sigma \coloneqq (\sigma^1, \dots, \sigma^N)^\top, \tag{3.23}$$

where $\mathbf{b}$ utilizes vector concatenation and Σ utilizes matrix concatenation. These link each player's drift and diffusion terms together into drift and diffusion terms, for the system as a whole. The individual contributions from each player, b^i and σ^i for all $i \in \mathcal{I}$, are defined as in Section 3.1.1. Finally, the deterministic initial condition of the SDE is given by $\mathbf{x}_0 = (x_0^1, \dots, x_0^N)^\top \in \mathbb{R}^{dN}$.

Remark 7. *Note that the state $X_t^{i,\mathbf{u}}$ of player i is not determined solely by its own control u_t^i, it is also coupled with the other players' states as well as controls. In particular, the coefficients b^i and σ^i depend on the N-player state vector $\mathbf{X}_t^{\mathbf{u}}$. Moreover, b^i considers the controls of all players.*

Furthermore, the drift and diffusion coefficients are, as in Section 3.1.1, assumed to be measurable Lipschitz continuous functions. That is, there exist constants $C_{\mathbf{b}}, C_\Sigma > 0$ such that

$$|\mathbf{b}(t,\mathbf{x},\mathbf{u}) - \mathbf{b}(t,\mathbf{x}',\mathbf{u}')| \leq C_{\mathbf{b}}(|\mathbf{x}-\mathbf{x}'| + |\mathbf{u}-\mathbf{u}'|), \tag{3.24}$$

$$|\Sigma(t,\mathbf{x}) - \Sigma(t,\mathbf{x}')| \leq C_\Sigma|\mathbf{x}-\mathbf{x}'|, \tag{3.25}$$

for all $(t,\mathbf{x},\mathbf{x}',\mathbf{u}_t,\mathbf{u}_t') \in [0,T]\times\mathbb{R}^{Nd}\times\mathbb{R}^{Nd}\times\mathbb{U}\times\mathbb{U}$. Along with being Lipschitz continuous, the drift and diffusion coefficients are assumed to satisfy a linear growth condition in space as well as uniform growth in time, i.e., there exists a constant $\hat{C}_{\mathbf{b},\Sigma} > 0$ such that

$$|\mathbf{b}(t,\mathbf{x},\mathbf{u})| + |\Sigma(t,\mathbf{x})| \leq \hat{C}_{\mathbf{b},\Sigma}(1+|\mathbf{x}|), \quad (t,\mathbf{x},\mathbf{u}) \in [0,T]\times\mathbb{R}^{Nd}\times\mathbb{U}. \tag{3.26}$$

Note that, as in Section 3.1.1, the norm in the conditions above is the Frobenius norm. Then by Theorem 6 there exists a strong unique solution to (3.22). A further constraint is placed upon the drift and diffusion for the N-player game. That is, for any $\mathbf{u}\in\mathcal{U}$

$$\mathbb{E}\left[\int_0^T |\mathbf{b}(t,\mathbf{0}_{dN},\mathbf{u}_t)|^2 + |\Sigma(t,\mathbf{0}_{dN})|^2\,\mathrm{d}t\right] < \infty.$$

To have a well-posed N-player stochastic control problem we define the multi-agent cost functional

$$J^i(t,\mathbf{x};\mathbf{u}_t) \coloneqq \mathbb{E}\left[\int_t^T f^i(s,\mathbf{X}_s^{\mathbf{u}},\mathbf{u}_s)\,\mathrm{d}s + g^i(\mathbf{X}_T^{\mathbf{u}})\Big|\mathbf{X}_t^{\mathbf{u}} = \mathbf{x}\right], \quad i\in\mathcal{I}, \tag{3.27}$$

where $f^i : [0,T]\times\mathbb{R}^{dN}\times\mathbb{U}\to\mathbb{R}$ and $g^i:\mathbb{R}^{dN}\to\mathbb{R}$ denote the running and terminal cost for player i respectively. For both of these functions there exist constants $C_f, C_g > 0$ such that

$$|f^i(t,\mathbf{x},\mathbf{u}) - f^i(t,\mathbf{x}',\mathbf{u}')| \leq C_f(|\mathbf{x}-\mathbf{x}'| + |\mathbf{u}+\mathbf{u}'|), \tag{3.28}$$

$$|g^i(\mathbf{x}) - g^i(\mathbf{x}')| \leq C_g|\mathbf{x}-\mathbf{x}'|, \tag{3.29}$$

for all $(t,\mathbf{x},\mathbf{x}',\mathbf{u},\mathbf{u}',i)\in[0,T]\times\mathbb{R}^{dN}\times\mathbb{R}^{dN}\times\mathbb{U}\times\mathbb{U}\times\mathcal{I}$. Furthermore, there exist a constant C_{fg} such that

$$|f^i(t,\mathbf{x},\mathbf{u})| + |g^i(\mathbf{x})| \leq C_{fg}(1+|\mathbf{x}|), \quad (t,\mathbf{x},\mathbf{u})\in[0,T]\times\mathbb{R}^{dN}\times\mathbb{U}, \tag{3.30}$$

for all $i\in\mathcal{I}$. The control problem is then solved when a so-called Nash equilibrium is found, defined in the following.

Definition 17 (Nash equilibrium)**.** *A strategy, $\mathbf{u}_t^*$, is said to be a Nash-equilibrium if for every $i \in \mathcal{I}$*

$$J^i(t, x; \mathbf{u}_t^*) \leq J^i(t, x; (u_t^{1,*}, \dots, u_t^i, \dots, u_t^{N,*})),$$

for all strategies $u_t^i \in \mathcal{U}^i$, where $\mathbf{u}_t^*$ denotes the vector of optimal policies.

Definition 18. *Let $\boldsymbol{\pi} = (\pi^1, \dots, \pi^N)$, $\pi^i : [0,T] \times \mathbb{R}^{dN} \to \mathbb{U}^i$, $i = 1, \dots, N$, be a set of N deterministic, measurable Markov control policies. We call $\boldsymbol{\pi}$ a Markovian Nash equilibrium, if for every pair $(t, \mathbf{x}) \in [0,T] \times \mathbb{R}^{dN}$, the control $\mathbf{u}_t \in \mathcal{U}$ is given by $\boldsymbol{\pi}$. That is, for every $i \in \mathcal{I}$, $t \in [0,T]$ we have that $u_t^i = \pi^i(t, X_t^{\boldsymbol{\pi}})$, where $\mathbf{X}^{\boldsymbol{\pi}} = (\mathbf{X}_s^{\boldsymbol{\pi}})_{t \in [0,T]}$ is a unique, strong solution to the forward SDE* (3.22)*.*

The forward SDE (3.22) and the cost functional (3.27) constitute the whole game. However, these equations will use slightly different notation to align with the approach taken in this thesis. From now on, we will consider an active player i who applies the control policy β, while the remaning $N-1$ players apply the control $\mathbf{u}_t^{-i} = (u_t^1, \dots, u_t^{i-1}, u_t^{i+1}, \dots, u_t^N)$. Then, the dynamics of player i is given by

$$\mathrm{d}X_t^{i,\beta,\mathbf{u}^{-i}} = b^i\left(t, \mathbf{X}_t^{\beta,\mathbf{u}^{-i}}, (\beta_t, \mathbf{u}_t^{-i})\right)\,\mathrm{d}t + \sigma^i(t, \mathbf{X}_t^{\beta,\mathbf{u}^{-i}})\,\mathrm{d}\mathbf{W}_t.$$

The corresponding N-player state equation is written as

$$\mathrm{d}\mathbf{X}_t^{\beta,\mathbf{u}^{-i}} = \mathbf{b}\left(t, \mathbf{X}_t^{\beta,\mathbf{u}^{-i}}, (\beta_t, \mathbf{u}_t^{-i})\right)\,\mathrm{d}t + \Sigma(t, \mathbf{X}_t^{\beta,\mathbf{u}^{-i}})\,\mathrm{d}\mathbf{W}_t. \tag{3.31}$$

Note that the only difference between the previously defined forward SDE (3.22) and (3.31) is that in the latter, we specify the control that player i applies. Further, with a slight abuse of notation, we define that $(\beta_t, \mathbf{u}_t^{-i}) = (u_t^i, \dots, u_t^{i-1}, \beta_t, u_t^{i+1}, \dots u_t^N)$. Throughout the thesis, it is assumed that the control $\mathbf{u}^{-i}$ is given and fixed. With this assumption, it allows us to think that the entire strategy profile $\beta = (\beta_t)_{t \in [0,T]}$ becomes a solution to a standard stochastic problem. The associated cost functional is similarly defined as previously

$$J^i(t, \mathbf{x}; (\beta_t, \mathbf{u}_t^{-i})) \coloneqq \mathbb{E}\left[\int_t^T f^i\left(s, \mathbf{X}_s^{\beta,\mathbf{u}^{-i}}, (\beta_s, \mathbf{u}_s^{-i})\right)\,\mathrm{d}s + g^i(\mathbf{X}_T^{\beta,\mathbf{u}^{-i}})\middle|\mathbf{X}_t^{\beta,\mathbf{u}^{-i}} = \mathbf{x}\right],$$

for all $i \in \mathcal{I}$.

3.2.2 *N*-coupled HJB equations

This subsection presents the N-coupled HJB equations that arise when the derivation in Section 3.1.2 is extended to an N-player stochastic control game. The derivation closely follows the previous section. Thus, the derivation of N-coupled HJB equations is omitted to avoid repetition. Interested readers are referred to Appendix A.

The HJB system is, as previously, derived by assuming Bellman's dynamic programming principle. We begin by defining the value function of player i as

$$V^i(t,\mathbf{x}) = \inf_{\beta\in\mathcal{U}^i} \mathbb{E}\left[\int_t^T f^i\left(s, \mathbf{X}_s^{\beta,\mathbf{u}^{-i}}, (\beta_s, \mathbf{u}_s^{-i})\right) \mathrm{d}s + g^i(\mathbf{X}_T^{\beta,\mathbf{u}^{-i}})\Big|\mathbf{X}_t^{\beta,\mathbf{u}^{-i}} = \mathbf{x}\right].$$

The value function $V^i(t,\mathbf{x})$ describes the optimal cost over the interval $[t,T]$, for some $t \in [0,T)$. By repeating similar steps as the derivation of the HJB equation for stochastic control problems, the HJB system reads

$$\partial_t V^i + \frac{1}{2}\operatorname{Tr}(\Sigma\Sigma^\top \Delta_x V^i) + \mathcal{H}^i(t,\mathbf{x},\nabla_x V^i) = 0, \quad (t,\, x) \in [0,T)\times\mathbb{R}^{dN}, \tag{3.32a}$$

$$V^i(T,\mathbf{x}) = g^i(\mathbf{x}), \quad x \ \in \mathbb{R}^{dN}, \tag{3.32b}$$

for all $i \in \mathcal{I}$. Here, $\mathcal{H}^i$ denotes the Hamiltonian for player i which is defined as

$$\mathcal{H}^i(t,\mathbf{x},p) \coloneqq \inf_{\beta\in\mathcal{U}^i}\left\{\left\langle \mathbf{b}\left(t,\mathbf{x},(\beta,\mathbf{u}^{-i})\right), p\right\rangle + f^i\left(t,\mathbf{x},(\beta,\mathbf{u}^{-i})\right)\right\}, \tag{3.33}$$

where $p \in \mathbb{R}^{dN}$ is a placeholder for the spatial derivative of the value function for player i. As is evident, the HJB equation takes a similar form as for the single-agent case. Note that (3.32) gives a system of N-coupled HJB equation, as the value function V^i depends on both player i's control, as well as the other players' controls $\mathbf{u}^{-i}$. Previously, we mentioned that $\mathbf{u}^{-i}$ are considered both fixed and given. This yields almost a usual one-agent control structure. However, those frozen policies $\mathbf{u}^{-i}$ must themselves be the optimal control that solves the other players' HJB equations. In other words, V^i implicitly depends on the other players' control, which in turn depends on the other V^j for $j \neq i$.

Remark 8. *When considering the HJB equation of player i, we treat the control of the remaining players,* $\mathbf{u}^{-i}$*, as fixed. Thus, when minimizing the Hamiltonian of player i the minimization is done only with respect to the control of that player,* β*. Here, we write that the remaining players' control is fixed to provide an intuition that becomes useful from an algorithmic standpoint. If these HJB equations were to be solved analytically, all the controls* $\mathbf{u}$ *need to be solved simultaneously, which means that the equations are coupled. However, the problem formulation is presented this way, as the method developed in this thesis for solving these types of problems is based on fixing the inactive players' strategy and optimizing only for the active player until an equilibrium is reached. This allows for a more pedagogical explanation of the implementation in Section 5.*

Let $\mathbf{V} \coloneqq (V^1,\dots,V^N)^\top \in (C^{1,2}([0,T]\times\mathbb{R}^d))^N$ solve the N-coupled HJB equations. We now state an existence theorem, taken from [20], that guarantees the existence of a solution $\mathbf{V}$ to (3.32).

Theorem 13 (Existence and uniqueness)**.** *Let the Lipschitz conditions on the drift* (3.24)*, diffusion* (3.25)*, running cost* (3.28) *and terminal cost* (3.29) *hold. Furthermore, let the uniform growth condition hold for the drift and diffusion* (3.26) *coefficients as well as for the running cost and terminal cost* (3.30)*. Then under the following assumptions*

(i) *The drift coefficients* b^i *are bounded on* $[0,T] \times \mathbb{R}^{dN} \times \mathbb{U}$ *for all* $i \in \mathcal{I}$.
(ii) $\sigma^i \sigma^{i\top}$ *is uniformly non-degenerate, that is the smallest eigenvalue is uniformly bounded from below by a positive constant on* $[0,T] \times \mathbb{R}^{dN}$.
(iii) *The functions* f^i *are bounded on* $[0,T] \times \mathbb{R}^{dN} \times \mathbb{U}$.
(iv) *The minimizer* $\pi^i : [0,T] \times \mathbb{R}^{dN}$ *of* (3.33) *is Lipschitz continuous in* $\mathbf{x} \in \mathbb{R}^{dN}$ *and satisfies a uniform growth condition in* $(t,x) \in [0,T] \times \mathbb{R}^{dN}$.

Then (3.32a)–(3.32b) *has a unique solution* $\mathbf{V} \in (C_{\mathrm{b}}^{1,2}([0,T] \times \mathbb{R}^d))^N$, *and the time derivative and second-order space derivatives of* $\mathbf{V}$ *belongs to* $(L^p_{\mathrm{loc}}([0,T] \times \mathbb{R}^{dN}))^N$ *for any* $p > 1$, *where* L^p_{loc} *denotes local integrability over compact subsets, and is formally defined as*

$$L^p_{\mathrm{loc}}([0,T] \times \mathbb{R}^{dN}) \coloneqq \left\{ f : [0,T] \times \mathbb{R}^{dN} \to \mathbb{R} \middle| f|_{(t,\mathbf{x})=K} \in L^p(K),\ \forall K \subset [0,T] \times \mathbb{R}^{dN} \right\},$$

where K *is a compact set. Furthermore, for any bounded, measurable function* π^i *and any initial condition* $\mathbf{x} \in \mathbb{R}^{dN}$, *the SDE* (3.22) *has a unique, strong solution for all* $i \in \mathcal{I}$.

Proof. The claim is proved in [20]. □

3.2.3 Reformulation to system of FBSDEs

As previously stated, it is not trivial to solve the HJB equation and even more so for N-coupled HJB equations as (3.32). Furthermore, the techniques used to solve it scale poorly to high-dimensional settings. Therefore, it is reformulated to a system of coupled FBSDEs. The derivation of FBSDEs follows the same steps as the ones presented in Section 3.1.3. Therefore, to avoid repetition, the derivation is omitted.

By considering the optimal feedback functions that minimizes the Hamiltonian (3.33), in similarity to the previous section, we write

$$\beta^* = \pi^i(t,\mathbf{x}) = \kappa^i(t,\mathbf{x},p) = \underset{\beta \in \mathcal{U}^i}{\arg\min} \left\{ \mathbf{b}(t,\mathbf{x},\beta,\mathbf{u}^{-i})^\top p + f^i(t,\mathbf{x},\beta,\mathbf{u}^{-i}) \right\}.$$

Here, it is important to note that the Markovian control map π^i may be a completely different function for each player. That is the case when heterogeneous games are considered when different players have different dynamics and cost functional. Specifically, the feedback map differs when the drift of the forward SDE of player i, b^i and the running cost f^i in the cost functionals differs between the players. Since the optimal control π^i is considered for player i, we write $\mathbf{X}^{\beta^*,\mathbf{u}^{-i}}$.

Having established the existence and uniqueness theorem for the N-player HJB system, we now continue with deriving the system of N-coupled FBSDEs. That is, the stochastic representation of the HJB equations, which also is the fundamental concept of the multi-player stochastic problem. Assuming existence and uniqueness theorem is satisfied (Theorem 13). Then there exists a unique classical solution $\mathbf{\Phi} = (\Phi^1, \ldots, \Phi^N) \in (C^{1,2}([0,T] \times \mathbb{R}^{dN}))^N$ that solves the system of N-coupled HJB equations (3.32).

The next step is to derive the PDE solution Φ^i for each player i with a stochastic representation. To do this, we begin by applying Itô's formula to Φ^i, which yields

$$\Phi^i(t, \mathbf{X}_t^{\beta^*, \mathbf{u}^{-i}}) = \Phi^i(0, \mathbf{X}_0^{\beta^*, \mathbf{u}^{-i}}) + \int_0^t (\partial_s + L^{\beta^*})\Phi^i(s, \mathbf{X}_s^{\beta^*, \mathbf{u}^{-i}})\, \mathrm{d}s + \int_0^t \Sigma(s, \mathbf{X}_s^{\beta^*, \mathbf{u}^{-i}})^\top \nabla_x \Phi^i(s, \mathbf{X}_s^{\beta^*, \mathbf{u}^{-i}})\, \mathrm{d}W_s. \tag{3.34}$$

where L^{β^*} is, as previously, the second-order differential operator defined as

$$L^{\beta^*}\Phi^i(t, \mathbf{x}) = \frac{1}{2} \operatorname{Tr}\left(\Sigma\Sigma^\top \Delta_x \Phi^i\right)(t, \mathbf{x}) + \mathbf{b}\left(t, \mathbf{x}, (\beta^*, \mathbf{u}^{-i})\right)^\top \nabla_x \Phi^i(t, \mathbf{x}).$$

Now, recall that the HJB equation (3.32a) for each player i, which characterizes the value function $\Phi^i = V^i$, can be formulated as

$$\begin{aligned} -\partial_t \Phi^i(t, \mathbf{x}) &= \frac{1}{2} \operatorname{Tr}\left(\Sigma\Sigma^\top \Delta_x \Phi^i\right)(t, \mathbf{x}) + \mathcal{H}^i(t, \mathbf{x}, \nabla_x \Phi^i) \\ &= \inf_{\beta \in \mathcal{U}^i} \left\{ L^\beta \Phi^i(t, \mathbf{x}) + f^i\left(t, \mathbf{x}, (\beta, \mathbf{u}^{-i})\right) \right\}. \end{aligned} \tag{3.35}$$

At the optimal control $\beta_t^* = \kappa^i(t, \mathbf{x}, \nabla_x \phi^i)$, the infimum is achieved, and thus the HJB equation (3.35) simplifies to

$$-\partial_t \Phi^i(t, \mathbf{x}) = L^{\beta^*}\Phi^i(t, \mathbf{x}) + f^i\left(t, \mathbf{x}, (\beta_t^*, \mathbf{u}_t^{-i})\right).$$

Inserting the above into Itô's formula (3.34), yields the following backward stochastic differential equation

$$\Phi^i(t, \mathbf{X}_t^{\beta^*, \mathbf{u}^{-i}}) = \Phi^i(0, \mathbf{X}_0^{\beta^*, \mathbf{u}^{-i}}) - \int_0^t f^i(s, \mathbf{X}_s^{\beta^*, \mathbf{u}^{-i}}, (\beta_s^*, \mathbf{u}_s^{-i}))\, \mathrm{d}s + \int_0^t \Sigma(s, \mathbf{X}_s^{\beta^*, \mathbf{u}^{-i}})^\top \nabla_x \Phi^i(s, \mathbf{X}_s^{\beta^*, \mathbf{u}^{-i}})\, \mathrm{d}W_s. \tag{3.36}$$

In order to simplify the notation for the FBSDE formulation, we define the processes $Y_t^i := \Phi^i(t, X_t^{\beta^*, \mathbf{u}^{-i}})$ and $Z_t^i := \Sigma(s, \mathbf{X}_s^{\beta^*, \mathbf{u}^{-i}})^\top \nabla_x \Phi(t, X_t^{\beta^*, \mathbf{u}^{-i}})$. With this notation, the BSDE (3.36) becomes

$$Y_t^i = Y_0^i - \int_0^t f^i\left(s, \mathbf{X}_s^{\beta^*, \mathbf{u}^{-i}}, (\beta_s^*, \mathbf{u}_s^{-i})\right) \mathrm{d}s + \int_0^t Z_s^i\, \mathrm{d}W_s.$$

The process $Y^i = (Y_t^i)_{t \in [0,T]}$ represents the solution to the HJB equation (3.32) for player i. The initial value of the Y-process is not known in advance. Instead, the backward equation is formulated to evolve backward in time, starting from the known terminal condition. Thus, rearranging terms and reversing the direction of time as done in Section 3.1.3, we get

$$Y_t^i = g^i(\mathbf{X}_T^{\beta^*, \mathbf{u}^{-i}}) + \int_0^t f^i\left(s, \mathbf{X}_s^{\beta^*, \mathbf{u}^{-i}}, (\beta_s^*, \mathbf{u}_s^{-i})\right) \mathrm{d}s - \int_0^t Z_s^i\, \mathrm{d}W_s. \tag{3.37}$$

Now, using the fact that the optimal control is given by a feedback map $\beta_t^* = \kappa^i(t, \mathbf{x}, \nabla_x \Phi)$, we define

$$\bar{\mathbf{b}}(t, \mathbf{x}, z^i, \mathbf{u}^{-i}) \coloneqq \mathbf{b}\left(t, \mathbf{x}, (\kappa^i(t, \mathbf{x}, \psi^i), \mathbf{u}^{-i})\right),$$
$$\bar{f}^i(t, \mathbf{x}, z^i, \mathbf{u}^{-i}) \coloneqq f^i\left(t, \mathbf{x}, (\kappa^i(t, \mathbf{x}, \psi^i), \mathbf{u}^{-i})\right),$$

where $\psi^i = (\Sigma(t, \mathbf{x})\Sigma(t, \mathbf{x})^\top)^{-1}\Sigma(t, \mathbf{x})z^i$. Moreover, in order to simplify the notation, we drop the superscript β^* for the state and define $\mathbf{X}_t^{\mathbf{u}^{-i}} \coloneqq \mathbf{X}_t^{\beta^*, \mathbf{u}^{-i}}$. This, because the feedback map is explicitly included in both $\bar{\mathbf{b}}$ and $\bar{f}^i$, which means that the state is optimally controlled given the Z^i-process. Substituting these into the forward SDE (3.31) and the backward SDE (3.37), we obtain the final form of the FBSDE system for player i

$$\mathbf{X}_t^{\mathbf{u}^{-i}} = \mathbf{x}_0 + \int_0^t \bar{\mathbf{b}}\left(s, \mathbf{X}_s^{\mathbf{u}^{-i}}, Z_s^i, \mathbf{u}_s^{-i}\right) \mathrm{d}s + \int_0^t \Sigma(s, \mathbf{X}_s^{\mathbf{u}^{-i}}) \, \mathrm{d}W_s, \tag{3.38a}$$

$$Y_t^i = \mathbf{g}(\mathbf{X}_s^{\mathbf{u}^{-i}}) + \int_t^T \bar{f}^i\left(s, \mathbf{X}_s^{\mathbf{u}^{-i}}, Z_s^i, \mathbf{u}^{-i}\right) \mathrm{d}s - \int_t^T Z_t \, \mathrm{d}W_s. \tag{3.38b}$$

Remark 9. *One potentially confusing aspect of the FBSDE formulation for player i is that it consists of N forward SDEs but only a single backward SDE for that player. The intuition behind this is that the control β_t for player i is optimized given the controls $\mathbf{u}^{-i}$ of the remaining players, since they are considered frozen. The optimized control β directly influences not only the state $X^{i,\mathbf{u}^{-i}}$ of player i but also the dynamics of all other players. This is due to the coupling of the dynamics, as each players state depends on both the state and control of all other players. As a result, changes to β influence not only the player i, but the whole system $\mathbf{X}^{\mathbf{u}^{-i}} = (X^{1,\mathbf{u}^{-i}}, \dots, X^{N,\mathbf{u}^{-i}})$. Thus, when only optimizing for a single player at a time, it is still necessary to track the entire N-player state process.*

Remark 10. *If we consider the entire system at once without fixed strategies, we get the following formulation*

$$\mathbf{X}_t^{\mathbf{u}} = \mathbf{x}_0 + \int_0^t \overline{\mathbf{b}}(s, \mathbf{X}_s^{\mathbf{u}}, \mathbf{Z}_s) \, \mathrm{d}s + \int_0^t \Sigma(s, \mathbf{X}_s^{\mathbf{u}}) \, \mathrm{d}W_s,$$
$$\mathbf{Y}_t = \mathbf{g}(\mathbf{X}_T^{\mathbf{u}}) + \int_t^T \overline{\mathbf{f}}(s, \mathbf{X}_s^{\mathbf{u}}, \mathbf{Z}_s) \, \mathrm{d}s - \int_t^T \mathbf{Z}_t \, \mathrm{d}W_s,$$

where $\mathbf{X}^{\mathbf{u}} \in \mathbb{R}^{dN}$ denotes the state of all players and $\mathbf{Y} = (Y^1, \dots, Y^N)$ denotes the collection of all backward SDEs. Here, we are considering the entire system, instead of only what is needed to solve player i's control problem. Using this formulation, the entire system needs to be solved simultaneously. However, it is important to note that the above system is equivalent to the formulation (3.38a)–(3.38b)*. In the latter formulation, the forward and backward equations are written separately for each player i while considering the controls $\mathbf{u}^{-i}$ of the remaining players as fixed. The fact that in* (3.38) *the fixed control $\mathbf{u}^{-i}$ needs themselves to be solved as a part of a coupled system, shows that the systems are indeed equivalent. However, we use* (3.38) *for the purpose of the numerical methods, as the formulation aligns with the algorithmic approach. Thus, this formulation will be easier to adapt to the numerical schemes.*

4

Single agent setting

In this section, the two main solution techniques are introduced. This includes a description of the methods proposed in [4] and [6]. Both methods revolve around formulating the FBSDE as a stochastic optimization problem, which is described in detail. We present a theorem that under the rather strict condition of weakly coupled FBSDEs, there is an a priori error bound. This serves as a motivation for the choice of the objective function, which is to be minimized. Moreover, the pseudo code describing the implementation of each technique is presented. At the end of this section, a numerical example is presented, demonstrating the solutions generated by each technique. The example presented is a linear–quadratic–Gaussian (LQG) control problem in both one and multiple dimensions. It has been demonstrated that Deep FBSDE fails to converge for, inter alia, LQG problems in high dimension. However, in one dimension, the problem is nice enough to converge, even for the Deep FBSDE method.

The two methods mentioned above, Deep FBSDE and Deep C-FBSDE, are two proposals on solve (3.21), i.e., the stochastic representation of the control problem for a single agent. It has been shown that Deep FBSDE fails for e.g. linear-quadratic control problems. The problem of when exactly the Deep FBSDE method fails to converge is still open. However, advancements have been made, e.g. [6], where the authors state that control problems are typically too strongly coupled for the Deep FBSDE method to successfully converge. Moreover, the robust method, Deep C-FBSDE, addresses the limitations of the Deep FBSDE method. The more robust modifications of Deep FBSDE treat the initial value of the backward equation as a fixed parameter. Furthermore, the loss function is modified to increase the stability of the solution.

4.1 Deep FBSDE

We begin by formulating the variational formulation of the FBSDE (3.21) for the Deep FBSDE method as a stochastic optimization problem. Recall that we defined $X \coloneqq X^{u^*}$, where $u_t^* = \kappa(t, X_t, \nabla_x V)$. However, if the gradient $\nabla_x V$ is suboptimal, then the resulting control u_t, and hence the triple (X_t, Y_t, Z_t), is likewise suboptimal. To indicate that the processes in the variational formulation depend on the a priori unknown Z-process rather than the control (since we assume that κ is known), the Markov map $\zeta : [0, T] \times \mathbb{R}^d \to \mathbb{R}^k$ is included in the superscript. Thus, the processes are written $(X_t^{y_0,\zeta}, Y_t^{y_0,\zeta}, Z_t^{y_0,\zeta})$. Further, the initial value y_0 of the backward equation

is unknown a priori and hence treated as a free parameter that is optimized along with ζ. To underline the processes' dependence on the initial value as well as indicate the parameter we are optimizing, y_0 is also included in the superscript. Thus, the variational formulation is stated as

$$\underset{y_0,\,\zeta}{\text{minimize}} \quad \mathbb{E}\left[|g(X_T^{y_0,\zeta}) - Y_T^{y_0,\zeta}|^2\right], \tag{4.1a}$$

$$\text{subject to} \quad X_t^{y_0,\zeta} = x_0 + \int_0^t \bar{b}\left(s, X_s^{y_0,\zeta}, Z_s^{y_0,\zeta}\right)\,\mathrm{d}s + \int_0^t \sigma\left(s, X_s^{y_0,\zeta}\right)\,\mathrm{d}W_s, \tag{4.1b}$$

$$Y_t^{y_0,\zeta} = y_0 - \int_0^t \bar{f}\left(s, X_s^{y_0,\zeta}, Z_s^{y_0,\zeta}\right)\,\mathrm{d}s + \int_0^t Z_s^{y_0,\zeta}\,\mathrm{d}W_s, \tag{4.1c}$$

$$Z_t^{y_0,\zeta} = \zeta(t, X_t^{y_0,\zeta}), \tag{4.1d}$$

where $y_0 \in \mathbb{R}$ and $\zeta \in C^{1,1}([0,T] \times \mathbb{R}^d; \mathbb{R}^k)$.

Next, we consider the implementation of a solution algorithm for (4.1). For this we begin by describing the fully discretized variational formulation for one batch. Consider a partition $0 = t_0 < t_1 < \ldots < t_{\tilde{N}} = T$ of the interval $[0,T]$, and let $\Delta t_k = t_{k+1} - t_k$. We write $\hat{X} \approx X$ for the discretized state process. This is applied to all discretized processes. Then, the free parameters are the initial value y_0 of the backward equation and the feedback $\zeta(t,x)$. These parameters are approximated by neural networks. Let $\psi^{y_0}(x \mid \theta^{y_0})$ be a network for approximating the initial value y_0 with parameters θ^{y_0}. Further, $\psi^{\zeta}(t,x \mid \theta^{\zeta})$ denotes the network with parameters θ^{ζ} for approximating the Markov map $\zeta(t,x)$. Both networks are defined according to (2.3), with the only difference being the latter network incorporates time embedding, i.e., the current time t is an additional input. To train these parameters, consider M_{train} independent paths of Brownian motion increments. That is, $\Delta W(m) = (\Delta W_0(m), \ldots, \Delta W_{\tilde{N}-1}(m))$ for $m = 1, \ldots, M_{\text{train}}$ where $\Delta W_i = W_{i+1} - W_i \sim N(0, \Delta t_i)$. These M_{train} paths are divided into batches of size M_{batch}. Then, the formulation for a single batch is written as

$$\underset{\theta^{y_0},\,\theta^{\zeta}}{\text{minimize}} \quad \mathcal{L}(\theta^{y_0}, \theta^{\zeta}) = \frac{1}{M_{\text{batch}}} \sum_{m=1}^{M_{\text{batch}}} |g(\hat{X}_N^{\theta^{y_0},\theta^{\zeta}}(m)) - \hat{Y}_N^{\theta^{y_0},\theta^{\zeta}}(m)|^2, \tag{4.2a}$$

$$\begin{aligned}\text{subject to} \quad \hat{X}_n^{\theta^{y_0},\theta^{\zeta}}(m) = x_0 &+ \sum_{k=0}^{n-1} \bar{b}\left(t_k, \hat{X}_k^{\theta^{y_0},\theta^{\zeta}}(m), \hat{Z}_k^{\theta^{y_0},\theta^{\zeta}}(m)\right)\Delta t_k \\ &+ \sigma\left(t_k, \hat{X}_k^{\theta^{y_0},\theta^{\zeta}}(m)\right)\Delta W_k(m),\end{aligned} \tag{4.2b}$$

$$\begin{aligned}\hat{Y}_n^{\theta^{y_0},\theta^{\zeta}}(m) = y_0 &- \sum_{k=0}^{n-1} \bar{f}\left(t_k, \hat{X}_k^{\theta^{y_0},\theta^{\zeta}}(m), \hat{Z}_k^{\theta^{y_0},\theta^{\zeta}}(m)\right)\Delta t_k \\ &+ (\hat{Z}_k^{\theta^{y_0},\theta^{\zeta}}(m))^{\top}\Delta W_k(m),\end{aligned} \tag{4.2c}$$

$$\hat{Z}_k^{\theta^{y_0},\theta^{\zeta}}(m) = \psi^{\zeta}(t_k, \hat{X}_k^{\theta^{y_0},\theta^{\zeta}}(m) \mid \theta^{\zeta}), \tag{4.2d}$$

$$y_0(m) = \psi^{y_0}(\hat{X}_0^{\theta^{y_0},\theta^{\zeta}}(m) \mid \theta^{y_0}). \tag{4.2e}$$

The expectation in the objective is approximated by Monte-Carlo simulation over each batch of sample paths. The idea of the algorithm is simple. For every batch,

the neural networks are used to approximate the initial value and the Markov map, then simulate sample paths for the forward equation (4.2b) and backward equation (4.2c) using Euler–Maruyama. The loss associated with each batch is given by the mean square error between Y_T and the terminal condition $g(X_T)$. Using this loss, we can utilize the concept of backpropagation as described in Section 2.4 with a gradient-based optimizer to get a better fit of the free parameters. In this way, the networks are updated before moving on to the next batch. The implementation for this method is outlined in Algorithm 1.

Before introducing the algorithm, we motivate the choice of the objective functions (4.1a) and (4.2a). For this, we state a theorem from [21].

Theorem 14 (A priori error bound)**.** *Consider a partition* $0 = t_0 < t_1 < \ldots < t_{\tilde{N}} = T$*, as previously, where* $\Delta t_i \coloneqq t_{i+1} - t_i$*. Further, let* $h = \max_{0 \leq i \leq \tilde{N}-1} \Delta t_i$*. For weakly coupled FBSDEs, i.e., FBSDEs which satisfy the so-called monotonicity condition, and for sufficiently small* h*, there exists a constant* C *which is independent of* h *and the dimension* d *such that*

$$\| |X_t - \hat{X}_t| \|^2_{\mathbb{S}^2_T(\mathbb{R})} + \| |Y_t - \hat{Y}_t| \|^2_{\mathbb{S}^2_T(\mathbb{R})} + \| Z_t - \hat{Z}_t \|^2_{\mathbb{H}^2_T(\mathbb{R}^d)} \leq C \left(h + \mathbb{E}\left[|g(\hat{X}_T) - \hat{Y}_T|^2 \right] \right),$$

where (X_t, Y_t, Z_t) *are the true processes, and* $\hat{X}_t = X^{y_0,\zeta}_{t_i}$*,* $\hat{Y}_t = Y^{y_0,\zeta}_{t_i}$*,* $\hat{Z}_t = Z^{y_0,\zeta}_{t_i}$ *are the approximations with some* ζ*.*

Therefore, the total error is minimized by the time step as well as the terminal condition, which serves as the objective function. Thus, it is a sensible choice to minimize this term when the monotonicity condition holds.

Algorithm 1: Deep FBSDE

Input : Initialized neural networks ψ^{y_0} and ψ^{ζ} with paramters θ^{y_0} and θ^{ζ}.
Data set of M_{train} Brownian motion paths $\Delta W = \{\Delta W_{t_i}\}_{i=0}^{n-1}$.
Batch size M_{batch}, which gives $N_{\text{batch}} = M_{\text{train}}/M_{\text{batch}}$ batches.
Number of epochs N_{epoch}.

Output: Trained neural networks ψ^{y_0} and ψ^{ζ}.

for $i \leftarrow 1, \ldots, N_{\text{epoch}}$ **do**
  for $k \leftarrow 1, \cdots, N_{\text{batch}}$ **do**
    for $m \leftarrow 1, \ldots, M_{\text{batch}}$ **do**
      Set $\hat{X}_0^{\theta^{y_0},\theta^{\zeta}}(m) = x_0$.
      Approximate initial condition y_0 using the neural network ψ^{y_0}, $y_0(m) = \psi^{y_0}(\hat{X}_0^{\theta^{y_0},\theta^{\zeta}}(m) \mid \theta^{y_0})$.
      for $n \leftarrow 0, \ldots, \tilde{N} - 1$ **do**
        Approximate the Markov map $Z_n^{\theta^{y_0},\theta^{\zeta}}(m)$ using the neural network ψ^{ζ}, $\hat{Z}_n^{\theta}(m) = \psi^{\zeta}(t_n, \hat{X}_0^{\theta^{y_0},\theta^{\zeta}}(m) \mid \theta^{\zeta})$.
        Compute $\hat{X}_n^{\theta^{y_0},\theta^{\zeta}}(m)$ according to (4.2b).
        Compute $\hat{Y}_n^{\theta^{y_0},\theta^{\zeta}}(m)$ according to (4.2c).
      end
    end
    Compute loss $\mathcal{L}(\theta^{y_0}, \theta^{\zeta})$ for batch k according to (4.2a) and take optimization step according to chosen optimization algorithm, $\theta^{y_0}(k+1), \theta^{\zeta}(k+1) \leftarrow \arg\min_{\theta^{y_0},\theta^{\zeta}} \mathcal{L}(\theta^{y_0}(k), \theta^{\zeta}(k))$.
  end
end
return Trained neural networks, ψ^{y_0} and ψ^{ζ}.

4.2 Deep C-FBSDE

The Deep C-FBSDE method, first introduced in [6], generalizes Deep FBSDE by using a different variational formulation than (4.1). Here, the initial value y_0 is no longer a free parameter but is determined as the expected value of the so-called stochastic cost $\mathcal{Y}_0$. Thus, the key difference in making the Deep C-FBSDE method a robust alternative to the Deep FBSDE method is that it, by introducing the stochastic cost, removes a degree of freedom. Further, the objective is modified. In addition to Y_T satisfying the terminal condition, the stochastic cost is minimized. Intuitively, the expectation of the stochastic cost $\mathcal{Y}_0$ is equivalent to $J^{u(\zeta)}(0, x_0)$, where the control u is generated by ζ. This formulation explicitly minimizes the cost functional associated with the control problem. We present the variational

formulation according to [6] as

$$\begin{aligned}
\underset{\zeta}{\text{minimize}} \quad & \mathbb{E}\left[\mathcal{Y}_0^\zeta\right] + \lambda \text{Var}\left[\mathcal{Y}_0^\zeta\right], && \text{(4.3a)}\\
\text{subject to} \quad & X_t^\zeta = x_0 + \int_0^t \bar{b}(s, X_s^\zeta, Z_s^\zeta)\,\mathrm{d}s + \int_0^t \sigma(s, X_s^\zeta)\,\mathrm{d}W_s, && \text{(4.3b)}\\
& Y_t^\zeta = \mathbb{E}\left[\mathcal{Y}_0^\zeta\right] - \int_0^t \bar{f}(s, X_s^\zeta, Z_s^\zeta)\,\mathrm{d}s + \int_0^t Z_s^\zeta\,\mathrm{d}W_s, && \text{(4.3c)}\\
& \mathcal{Y}_0^\zeta = g(X_T^\zeta) + \int_0^T \bar{f}(s, X_s^\zeta, Z_s^\zeta)\,\mathrm{d}s - \int_0^T Z_s^\zeta\,\mathrm{d}W_s, && \text{(4.3d)}\\
& Z_t^\zeta = \zeta(t, X_t^\zeta), && \text{(4.3e)}
\end{aligned}$$

for some $\lambda > 0$. Here, it is important to note that the variance of the stochastic cost corresponds to (4.1a). This can be shown by

$$\begin{aligned}
\text{Var}[\mathcal{Y}_0] &= \mathbb{E}\left[|\mathbb{E}[\mathcal{Y}_0^\zeta] - \mathcal{Y}_0^\zeta|^2\right]\\
&= \mathbb{E}\left[\left|Y_T^\zeta + \int_0^T \bar{f}(s, X_s^\zeta, Z_s^\zeta)\,\mathrm{d}s - \int_0^T Z_s^\zeta\,\mathrm{d}W_s - \mathcal{Y}_0^\zeta\right|^2\right]\\
&= \mathbb{E}\left[\left|Y_T^\zeta - g(X_T^\zeta)\right|^2\right],
\end{aligned}$$

where (4.3c) and (4.3d) have been used in the second and third equality, respectively. This underlines the similarity between the two methods' objective functions.

Next, we describe the fully discrete variational problem of (4.3) for one batch. As for the Deep FBSDE method, we consider a partition $0 = t_0 < t_1 < \ldots < t_{\tilde{N}} = T$ of the interval $[0, T]$, and let $\Delta t_k = t_{k+1} - t_k$. Then, the free parameter is simply the feedback $\zeta(t, x)$, which is approximated by a neural network. Let $\psi^\zeta(\cdot|\,\theta^\zeta)$ be the network that approximates the Markov map $\zeta(t, x)$ with parameters θ^ζ. As for the Deep FBSDE method, the network is defined according to (2.3), with incorporated time embedding, i.e., the time t_k is an additional input to the state X. As previously, let M_{Train} denote independent paths of Browninan motion $\Delta W(m) = (\Delta W_0(m), \ldots, \Delta W_{N-1}(m))$ for $m = 1, 2, \ldots, M_{\text{Train}}$. Further, let $M_{\text{batch}} = 2\hat{M}_{\text{batch}}$ denote the size of each batch. Note that in objective (4.4a), we use two disjoint batches for the stochastic term and terminal condition. This is done in order to prevent bias and reduce gradient variance. The training of the parameters is done in the same manner as for the Deep FBSDE method, therefore we state the variational

formulation discretized for a single batch

$$\begin{aligned}
\underset{\theta^\zeta}{\text{minimize}} \quad \mathcal{L}(\theta^\zeta) \quad &= \frac{1}{\hat{M}_{\text{Batch}}} \sum_{m=1}^{\hat{M}_{\text{Batch}}} \hat{\mathcal{Y}}_0^{\theta^\zeta}(m) \\
&\quad + \frac{\lambda}{\hat{M}_{\text{Batch}}} \sum_{m=\hat{M}_{\text{Batch}}+1}^{2\hat{M}_{\text{Batch}}} \left| g\left(\hat{X}_N^{\theta^\zeta}(m)\right) - \hat{X}_N^{\theta^\zeta}(m) \right|^2, && \text{(4.4a)} \\
\text{subject to} \quad \hat{X}_n^{\theta^\zeta}(m) &= x_0 + \sum_{k=0}^{n-1} \bar{b}\left(t_k, \hat{X}_k^{\theta^\zeta}(m), \hat{Z}_k^{\theta^\zeta}(m)\right) \Delta t_k \\
&\quad + \sum_{k=0}^{n-1} \sigma\left(t_k, \hat{X}_k^{\theta^\zeta}(m)\right) \Delta W_k(m), && \text{(4.4b)} \\
\hat{Y}_n^{\theta^\zeta}(m) &= \frac{1}{\hat{M}_{\text{Batch}}} \sum_{r=1}^{\hat{M}_{\text{Batch}}} \hat{\mathcal{Y}}_0^{\theta^\zeta}(r) - \sum_{k=0}^{n-1} \bar{f}\left(t_k, \hat{X}_k^{\theta^\zeta}(m), \hat{Z}_k^{\theta^\zeta}(m)\right) \Delta t_k \\
&\quad + \sum_{k=0}^{n-1} (\hat{Z}_k^{\theta^\zeta}(m))^\top \Delta W_k(m), && \text{(4.4c)} \\
\hat{\mathcal{Y}}_0^{\theta^\zeta}(m) &= g(\hat{X}_N^{\theta^\zeta}(m)) + \sum_{k=0}^{N-1} \bar{f}(t_k, \hat{X}_k^{\theta^\zeta}(m), \hat{Z}_k^{\theta^\zeta}(m)) \Delta t_k \\
&\quad - \sum_{k=0}^{N-1} (\hat{Z}_k^{\theta^\zeta}(m))^\top \Delta W_k(m), && \text{(4.4d)} \\
\hat{Z}_k^{\theta^\zeta}(m) &= \psi^\zeta(t_k, \hat{X}_k^{\theta^\zeta}(m) \mid \theta^\zeta). && \text{(4.4e)}
\end{aligned}$$

The expectation and variation in the objective are approximated by Monte Carlo simulation over each batch of sample paths. The initial value of the backward equation is estimated using a Monte Carlo simulation of the stochastic cost on one set of sample paths. Then, the terminal error $|g(\hat{X}_N^{\theta^\zeta}) - \hat{Y}_N^{\theta^\zeta}|^2$ is calculated on an independent set of Brownian increments that has not been used in the previous step.

The spirit of the algorithm follows from the Deep FBSDE method. For every batch, the neural networks are used to approximate the Markov map and then simulate paths of the forward equation (4.4b). From these trajectories, we compute the expectation of the stochastic cost, which in turn is the initial condition for the backward equation (4.4c). Finally, we propagate both the forward equation (4.4b) and backward equation (4.4c) forward in time using Euler–Maruyama. The resulting loss is calculated according to (4.4a). With the loss, we can use backpropagation to update the networks. The implementation of this method is described in Algorithm 2.

Algorithm 2: Deep C-FBSDE method.

Input : Initialized neural network ψ^ζ with paramters θ^ζ.
Data set of M_{train} Brownian motion paths $\Delta W = \{\Delta W_i\}_{i=0}^{\tilde{N}-1}$.
Batch size $2\hat{M}_{\text{batch}}$, which gives $N_{\text{Batch}} = M_{\text{train}}/2\hat{M}_{\text{batch}}$ batches.
Number of epochs N_{epoch}.

Output: Trained neural network ψ^ζ.

for $i \leftarrow 1, \ldots, N_{epoch}$ **do**
  for $k \leftarrow 1, \cdots, N_{batch}$ **do**
    for $m \leftarrow 1, \ldots, \hat{M}_{\text{batch}}$ **do**
      Set $\hat{X}_0^{\theta^\zeta}(m) = x_0$.
      for $n \leftarrow 0, \ldots, \tilde{N} - 1$ **do**
        Approximate the Markov map using the neural network ψ^ζ,
        i.e. $\hat{Z}_n^{\theta^\zeta}(m) = \psi^\zeta(t_n, \hat{X}_n^{\theta^\zeta}(m) \mid \theta^\zeta)$.
        Compute $\hat{X}_{n+1}^{\theta^\zeta}(m)$ according to (4.4b).
      end
      Compute $\hat{\mathcal{Y}}_0^{\theta^\zeta}(m)$ according to (4.4d).
    end
    for $m \leftarrow \hat{M}_{\text{Batch}} + 1, \ldots, 2\hat{M}_{\text{Batchs}}$ **do**
      Set $\hat{X}_0^{\theta^\zeta}(m) = x_0$.
      Set $\hat{Y}_0^{\theta^\zeta}(m) = \frac{1}{\hat{M}_{\text{Batch}}} \sum_{m=1}^{\hat{M}_{\text{Batch}}} \hat{\mathcal{Y}}_0^{\theta^\zeta}(m)$.
      for $n \leftarrow 0, \ldots, \tilde{N} - 1$ **do**
        Approximate the Markov map using the neural network ψ^ζ,
        i.e. $\hat{Z}_n^{\theta^\zeta}(m) = \psi^\zeta(t_n, \hat{X}_k^{\theta^\zeta}(m) \mid \theta^\zeta)$.
        Compute $\hat{X}_{n+1}^{\theta^\zeta}(m)$ according to (4.4b).
        Compute $\hat{Y}_{n+1}^{\theta^\zeta}(m)$ according to (4.4c).
      end
      Compute loss $\mathcal{L}(\theta^\zeta)$ for batch k according to (4.4a) and take
      optimization step according to chosen optimization algorithm,
      $\theta^\zeta(k+1) \leftarrow \arg\min_{\theta^\zeta} \mathcal{L}(\theta^\zeta(k))$.
    end
  end
end
return Trained neural network, ψ^ζ.

4.3 Convergence of Deep FBSDE

The problem of when the Deep FBSDE method fails to converge, and why, is an open problem. However, cases and insights have been made, see e.g., [6]. It has been shown that the Deep FBSDE method converges when the forward and backward equation are weakly coupled. To get some intuition for this, consider the decoupled

FBSDE

$$X_t = x_0 + \int_0^t \bar{b}(s, X_s)\,\mathrm{d}s + \int_0^t \sigma(s, X_s)\,\mathrm{d}W_s, \tag{4.5a}$$

$$Y_t = y_0 - \int_t^T \bar{f}(s, X_s, Z_s)\,\mathrm{d}s + \int_t^T Z_s^\top\,\mathrm{d}W_s, \tag{4.5b}$$

where all parameters are defined as in Section 3.1.3. Here, the forward equation (4.5a) does not depend on Z, and thus X_T and the terminal condition is determined directly, by Euler–Maruyama. This means that the first term in the objective (4.1a) becomes trivial to evaluate, which simplifies the optimization problem to determine only y_0 and ζ in the backward equation (4.5b). Thus, the backward SDE is solved, given a known terminal value. When the FBSDE is strongly coupled, that is, when there is a dependence on Z in both the forward and backward equations, then y_0 and ζ do not only affect the Y-process but also the X-process. Which in turn influences the terminal condition $g(X_T)$. Thus, in the strongly coupled case, compared to decoupled FBSDE, there are additional degrees of freedom, which makes the problem more complex.

In [4], the authors stated that Theorem 14 holds only when the drift coefficient $\bar{b}$, the diffusion term σ and the running cost $\bar{f}$ depends on (X, Y) and not the Z-process. In [6], it is stated that with additional regularity and growth conditions on (b, σ, f, g), Theorem 14 can be generalized for strongly coupled FBSDE. However, the monotonicity condition is seldom satisfied for FBSDEs arising from stochastic control problems, which makes the Deep FBSDE method diverge for strongly coupled FBSDE.

The result one generates using the Deep FBSDE method for a problem where it does not converge is typically, for the case of the true initial condition being smaller than the computed initial condition, i.e. $y_0 < Y_0$, a good approximation of the forward SDE and a fulfilled terminal condition for the backward process. However, the backward process is distinct from any analytical solution in all times other than the terminal time T. Furthermore, the control process oscillates around the analytical solution for all times instead of approximating it more or less exactly. Such a solution solves the stochastic control problem, however it fails to solve the FBSDE and consequently the PDE.

Remark 11. *This has sparked further research where the control process is instead sought in a so-called bounded mean oscillation (BMO) space, see e.g. [22]. A BMO space is defined as the space of all integrable, predictable and positive processes that are a so-called BMO martingale, i.e. it holds*

$$\mathbb{E}\left[\int_\tau^T X_s^2\,\mathrm{d}s \middle| \mathcal{F}_\tau\right] \leq D, \quad \mathbb{P}\text{-}a.s. \tag{4.6}$$

for all stopping times $\tau \in [0, T]$ and some constant $D > 0$. Moreover, the space is equipped with a norm defined as the smallest D such that (4.6) *holds. This approach bears the weight of further research, however, time restrictions do not allow for it*

in this thesis.

Another typical behavior of the solution triple (X_t^u, Y_t, Z_t) in the case of divergence for the Deep FBSDE method is the flawed approximation of the control process that yields an incorrect control $u(\zeta)$, which in turn results in an altered forward trajectory. This happens in the case of the initial condition being smaller than the computed initial condition, i.e. $y_0 < Y_0$. Finally, this results in a flawed terminal condition for the backward process, which in turn results in the backward process being shifted away from the truth by a specific distance for all times.

4.4 Linear–Quadratic–Gaussian control problem

This section introduces a numerical example, namely the LQG differential game, and the results for each method in different dimensions. We will consider a one-dimensional case ($d = \ell = 1$), two-dimensional case ($d = \ell = 2$), and a higher dimensional example ($d = 6,\ \ell = 2$). The linear in LQG refers to the forward equation being composed of linear terms, i.e. the drift and diffusion are both linear. Then, the cost functional consists of quadratic terms, i.e. the running and terminal costs are both quadratic. For the numerical examples, the derivation of the HJB equation and its corresponding FBSDE is referred to in Section 3.1.2 and 3.1.3, respectively. Therefore, we simply state the resulting equations. We now present the problem formulation for an LQG problem with one player. We let $(\Omega, \mathcal{F}, (\mathcal{F}_t)_{t\in[0,T]}, \mathbb{P})$ be a filtered probability space with the filtration $(\mathcal{F}_t)_{t\in[0,T]}$ generated by the d-dimensional Brownian motion $W = (W_t)_{t\in[0,T]}$ and completed by the $\mathbb{P}$-null sets of Ω. Then let the state process $X^u = (X_t^u)_{t\in[0,T]}$ solve

$$\begin{aligned} \mathrm{d}X_t^u &= (A(C - X_t^u) + Bu)\ \mathrm{d}t + \sigma\ \mathrm{d}W_t, \quad X_0^u = x_0, \\ J^u(t,x) &= \mathbb{E}\left[\int_0^T \langle R_x X_s^u, X_s^u\rangle + \langle R_u u_s, u_s\rangle\ \mathrm{d}s + \langle G X_T^u, X_T^u\rangle\right], \end{aligned}$$

where the deterministic drift coefficient is made up of the matrices $A \in \mathbb{R}^{d\times d}$, $C \in \mathbb{R}^d$, and $B \in \mathbb{R}^{d\times \ell}$. Further, the deterministic diffusion coefficient is the matrix $\sigma \in \mathbb{R}^{d\times d}$. Lastly, the initial condition is a vector on the form $x_0 \in \mathbb{R}^d$. Moreover, for the cost functional we abuse the notation somewhat and let $R_x, G \in \mathbb{S}_+^d$ and finally $R_u \in \mathbb{S}_+^\ell$, where

$$\mathbb{S}_+^k = \left\{x \in \mathbb{S}^k : x \succeq 0\right\}, \quad \text{where} \quad \mathbb{S}^k = \left\{x : x^\top = x \in \mathbb{R}^{k\times k}\right\}. \tag{4.7}$$

Following the same steps as in Section 3.1.2 gives the corresponding HJB equation

$$\partial_t V + \inf_{u\in\mathcal{U}} \Big((A(\mathcal{C} - x) + Bu)^\top \nabla_{\mathbf{x}} V + \langle R_x x, x\rangle + \langle R_u u, u\rangle \Big) + \frac{1}{2}\mathrm{Tr}(\sigma\sigma^\top \Delta_{\mathbf{x}} V) = 0, \tag{4.8a}$$

$$V(T,x) = \langle Gx, x\rangle, \tag{4.8b}$$

where the spacetime dependence $(t,\, x) \in [0, T] \times \mathbb{R}^d$ is dropped for brevity unless specifically specified. Furthermore, reformulating the HJB equation to its' corresponding FBSDE according to the methodology in Section 3.1.3 gives

$$\mathrm{d}X_t^u = (A(C - X_t^u) + Bu)\ \mathrm{d}t + \sigma\, \mathrm{d}W_t, \qquad X_0^u = x_0, \tag{4.9a}$$

$$\mathrm{d}Y_t = -\left(\langle R_x X_t^u, X_t^u\rangle + \langle R_u u_t, u_t\rangle\right) \mathrm{d}s + Z_t\, \mathrm{d}W_t, \quad Y_t\ = \langle G X_T^u, X_T^u\rangle, \tag{4.9b}$$

which is the system that we solve according to Algorithm 1 and Algorithm 2. Analytically however, the system has a well known solution which is derived by the initial ansatz

$$V(t, x) = x^\top P(t) x + x^\top Q(t) + R(t),$$

for $P : [0, T] \to \mathbb{R}^{d\times d}$, $Q : [0, T] \to \mathbb{R}^d$ and $R : [0, T] \to \mathbb{R}$. Inserted into the HJB equation (4.8a)-(4.8b) and extracting terms of equal order with respect to x yield the so-called Riccati equations

$$\begin{aligned}
\dot{P}(t) - A^\top P(t) - P(t)A - P(t)BR_u^{-1}B^\top + R_x &= \mathbf{0}_{d\times d},\\
\dot{Q}(t) + 2P(t)AC - A^\top Q(t) - P(t)BR_u^{-1}B^\top Q(t) &= \mathbf{0}_d,\\
\dot{R}(t) + \mathrm{Tr}\left(\sigma\sigma^\top P(t)\right) + Q(t)^\top AC - \frac{1}{4}Q(t)^\top BR_u^{-1}B^\top Q(t) &= 0,\\
P(T) = G;\ Q(T) = \mathbf{0}_d;\ R(T) &= 0,
\end{aligned}$$

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

The Riccati system is easily solved by common ODE techniques, such as implicit Euler, and we simply state that the solution triple $(X_t^u, Y_t, Z_t) \in \mathbb{S}_T^2(\mathbb{R}) \times \mathbb{S}_T^2(\mathbb{R}) \times \mathbb{H}_T^2(\mathbb{R}^d)$ is given by computing sample paths of X_t^u, given the ansatz

$$\begin{aligned}
Y_t &= x^\top P(t) x + x^\top Q(t) + R(t),\\
Z_t &= 2P(t)x + Q(t).
\end{aligned}$$

4.4.1 One-dimensional example

The first example consists of a one-dimensional control ($\ell = 1$) and one-dimensional state ($d = 1$). The used parameters in (4.9a) are

$$A = -2, \qquad B = 1, \qquad C = -1, \qquad \sigma = 1.2, \qquad x_0 = 0.5.$$

while, for the backward equation (4.9b), the parameters are

$$R_x = 25, \qquad R_u = 0.25, \qquad G = 0.25.$$

The results for the LQC problem for the Deep FBSDE method are presented in Figure 4.1. The numerical results for the Deep C-FBSDE method are presented in Figure 4.2. The figures show that both method manages to approximate the state process $\hat{X}_t$, the backward process $\hat{Y}_t$ as well as the control process $\hat{Z}_t$. This holds true for the expectation in the left column as well as the sample paths in the right columns.

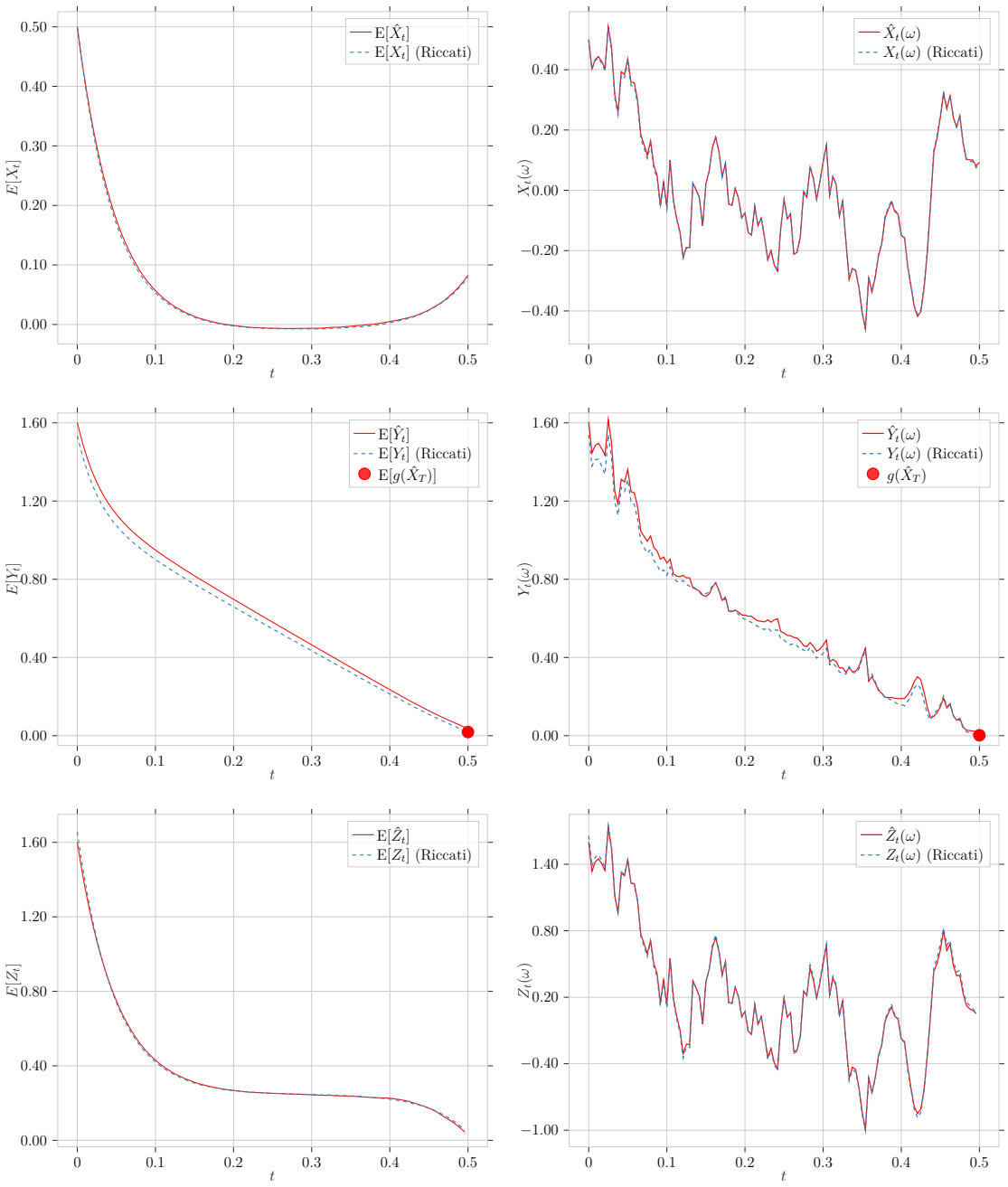

Figure 4.1: Results for the Deep FBSDE method with one-dimensional state and control. The top row illustrates the approximated state process $\hat{X}$ as well as the Riccati solution X. The second row illustrates the approximated backward process $\hat{Y}$, the Riccati approximation, and the computed terminal condition $g(\hat{X}_T)$. The last row illustrates the approximated control process $\hat{Z}$ as well as the analytical Riccati solution. The left column shows the expectation of the respective process, and the right column is an example sample path of the respective process.

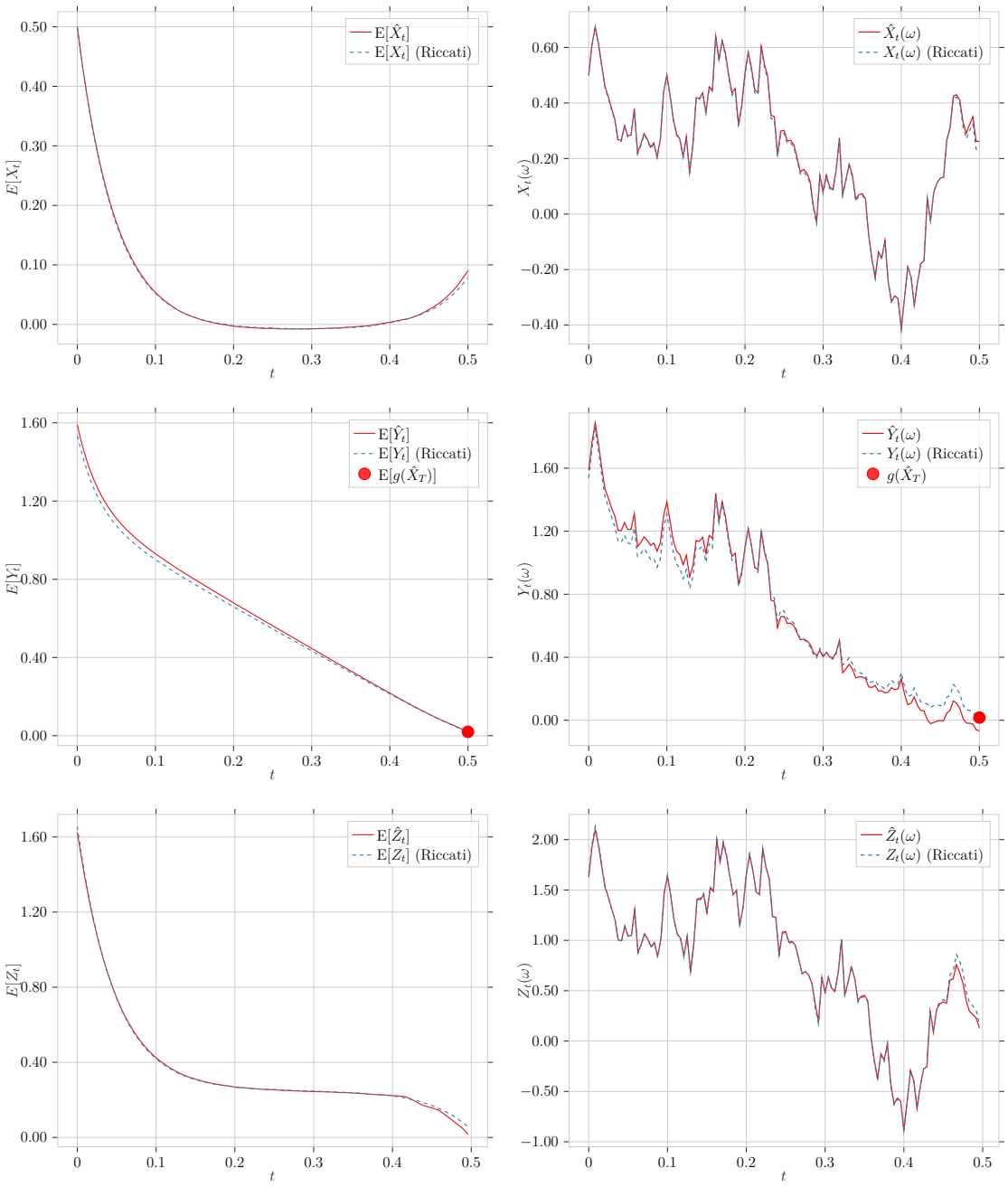

Figure 4.2: Results for the Deep C-FBSDE method with one-dimensional state and control. The left column illustrates the expectation of the processes and the right illustrates an example sample path. The top row illustrates the approximated state process $\hat{X}$ as well as the Riccati solution X. The second row illustrates the approximated backward process $\hat{Y}$, the Riccati solution, and the computed terminal condition $g(\hat{X}_T)$. The last row illustrates the approximated control process $\hat{Z}$ as well as the Riccati solution.

4.4.2 Two-dimensional example

The same numerical example was examined, with two-dimensional state and two-dimensional control. For this purpose, the following parameters

$$A = \begin{bmatrix} 1 & 0 \\ 0 & 2 \end{bmatrix}, \quad B = \begin{bmatrix} 1 & 0.5 \\ -0.5 & 2 \end{bmatrix}, \quad C = \begin{bmatrix} 0.1 \\ 0.2 \end{bmatrix}, \quad \sigma = \begin{bmatrix} 0.05 & 0.25 \\ 0.05 & 0.25 \end{bmatrix}, \quad x_0 = \begin{bmatrix} 0.1 \\ 0.1 \end{bmatrix},$$

was used for the forward equation (4.9a). Meanwhile, for the backward equation (4.9b), the considered parameters are

$$R_x = \begin{bmatrix} 100 & 0 \\ 0 & 1 \end{bmatrix}, \qquad R_u = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \qquad G = \begin{bmatrix} 1 & 0 \\ 0 & 100 \end{bmatrix}.$$

The numerical results for the two methods are presented in Figure 4.3 and Figure 4.4. For the Deep FBSDE method, in Figure 4.3, we can see in the middle row that the approximated $\hat{Y}$-process agrees with the terminal condition, $E[g(\hat{X}_T)]$, at the final time. This indicates that the objective is satisfied, which in turn means that the method has converged. However, for the state process $\hat{X}$ and $\hat{Z}$-process, there is a large discrepancy between the numerical approximations and the analytical solution. In contrast, the Deep C-FBSDE method results in the correct terminal condition as the Deep FBSDE method, but also results in the correct state process and control process for both components. Thereby, we have shown that the Deep C-FBSDE method is an improvement on the Deep FBSDE method. This holds for both the expectation in the left column and for a sample path in the right column.

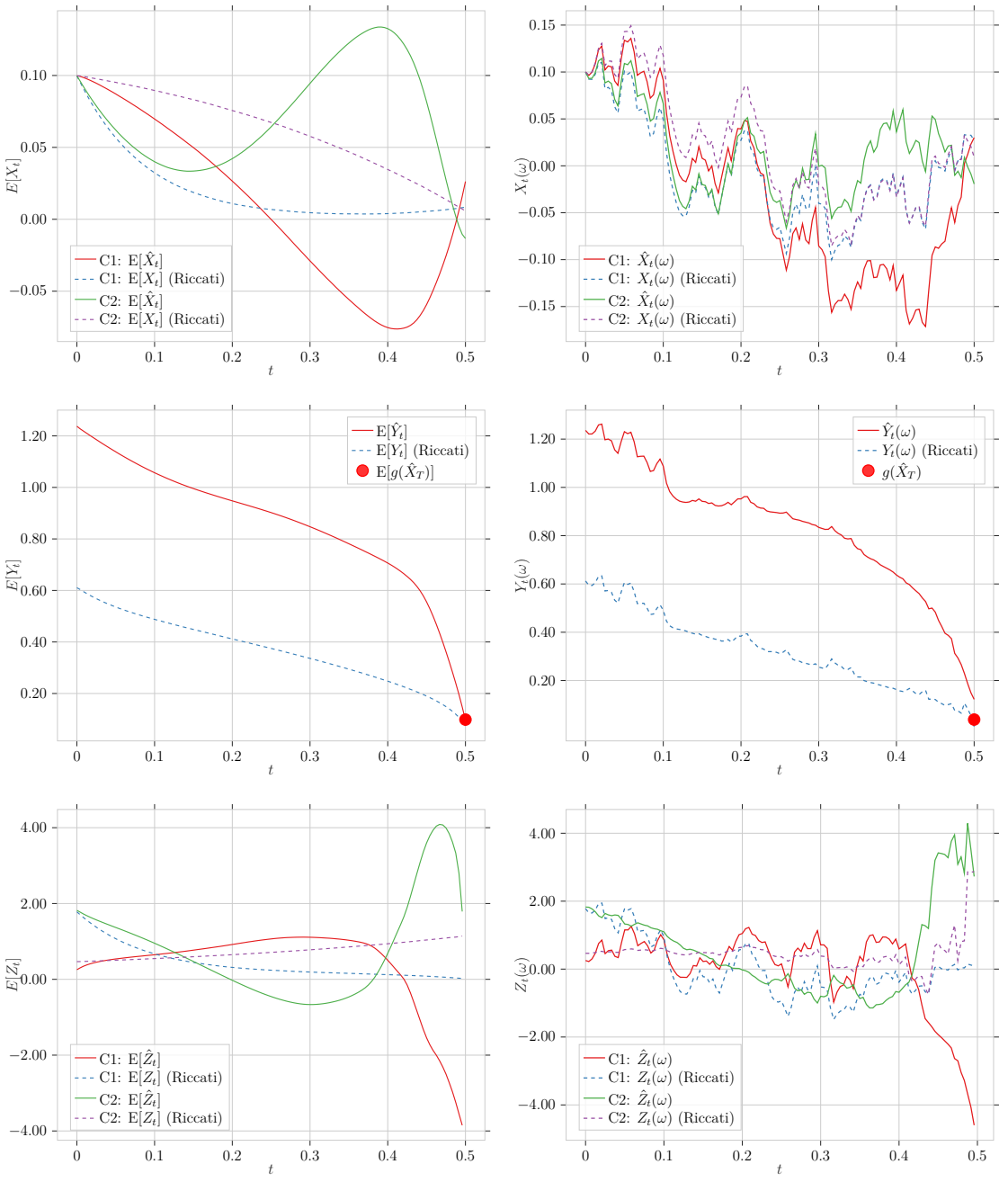

Figure 4.3: Results for the Deep FBSDE method for the two-dimensional example. Here, C1 and C2 refer to the first and second components of the two-dimensional processes. The left column shows the mean value of all sample paths, and the right is an arbitrary sample path. The rows illustrate the state process, the backward process (and computed terminal condition), and the control process, respectively. Recall that the backward process is a scalar and thus does not have multiple components.

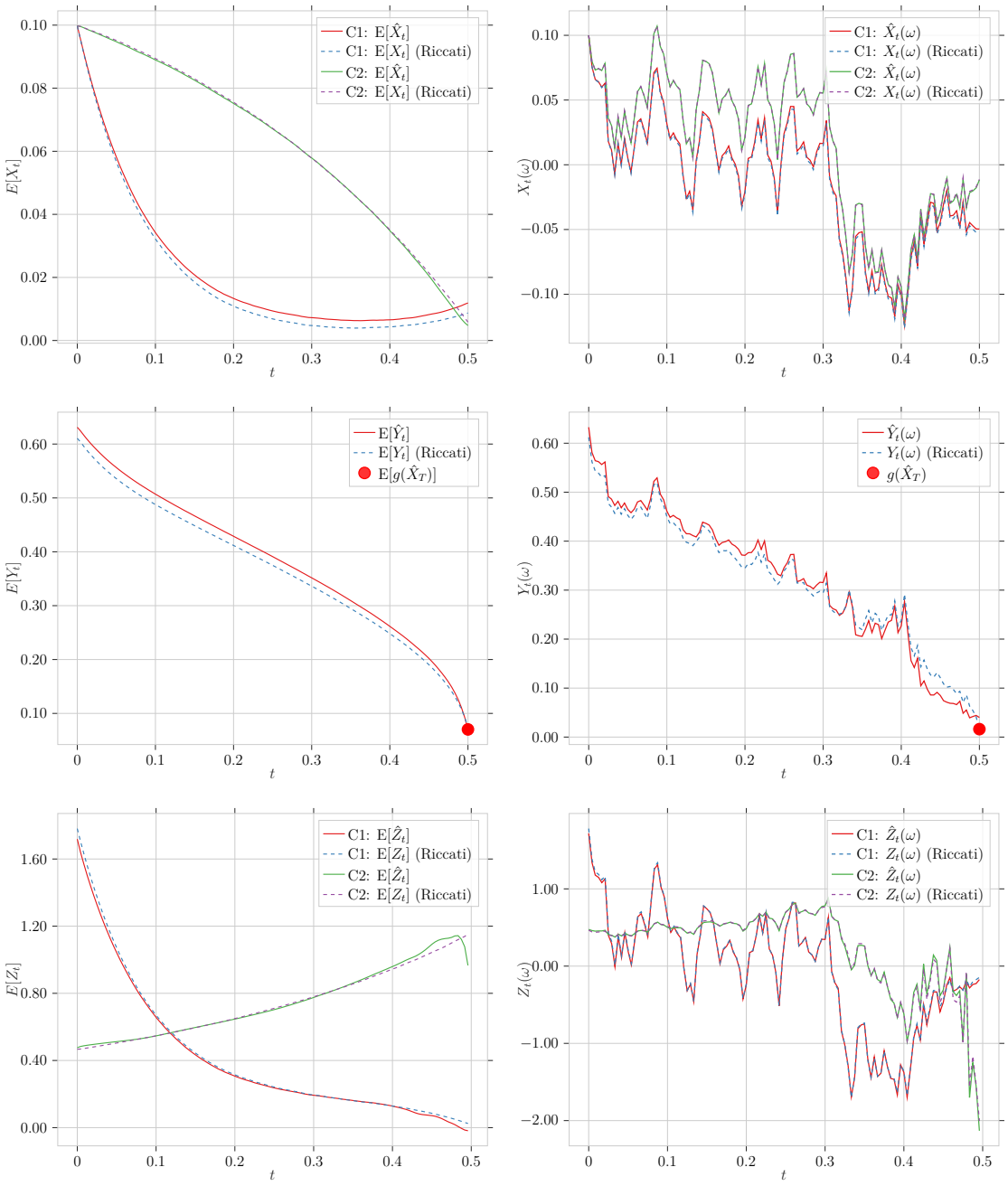

Figure 4.4: Results for the Deep C-FBSDE method for the two-dimensional example. The left column shows the expectation of all sample paths, and the right column illustrates an arbitrary sample path. The rows show the state process X, the backward process Y, and the control process Z, respectively.

4.4.3 High-dimensional example

For the LQG control problem, a slightly higher dimensional was considered for the state process. More specifically, the state process was assigned to be six-dimensional, while the control u is a two-dimensional process. In this case, the parameters for the state equations, are

$$A = \mathrm{diag}(1,\ 2,\ 3,\ 1,\ 2,\ 3), \qquad B = \begin{bmatrix} 1 & 1 & 0.5 & 1 & 0 & 0 \\ -1 & 1 & 1 & -1 & -1 & 1 \end{bmatrix}^\top,$$
$$C = (-0.2,\ -0.1,\ 0,\ 0,\ 0.1,\ 0.2)^\top, \quad \sigma = \mathrm{diag}(0.05,\ 0.25,\ 0.05,\ 0.25,\ 0.05,\ 0.25),$$
$$x_0 = (0.1,\ 0.1,\ 0.1,\ 0.1,\ 0.1,\ 0.1)^\top.$$

For the backward equation, the parameters are given by

$$R_x = \mathrm{diag}(5,\ 1,\ 5,\ 1,\ 5,\ 1), \qquad R_u = I_2, \qquad G = \mathrm{diag}(1,\ 25,\ 1,\ 25,\ 1,\ 25),$$

where I_2 denotes the identity matrix of size two.

Figures 4.5 and 4.6 show a similar performance as for the two-dimensional case. The Deep FBSDE method fails to approximate all processes, even though the approximated terminal condition $g(\hat{X}_T)$ is satisfied for the backward process. The Deep C-FBSDE method, in contrast, approximates all processes well, both in terms of average values but also specific sample paths.

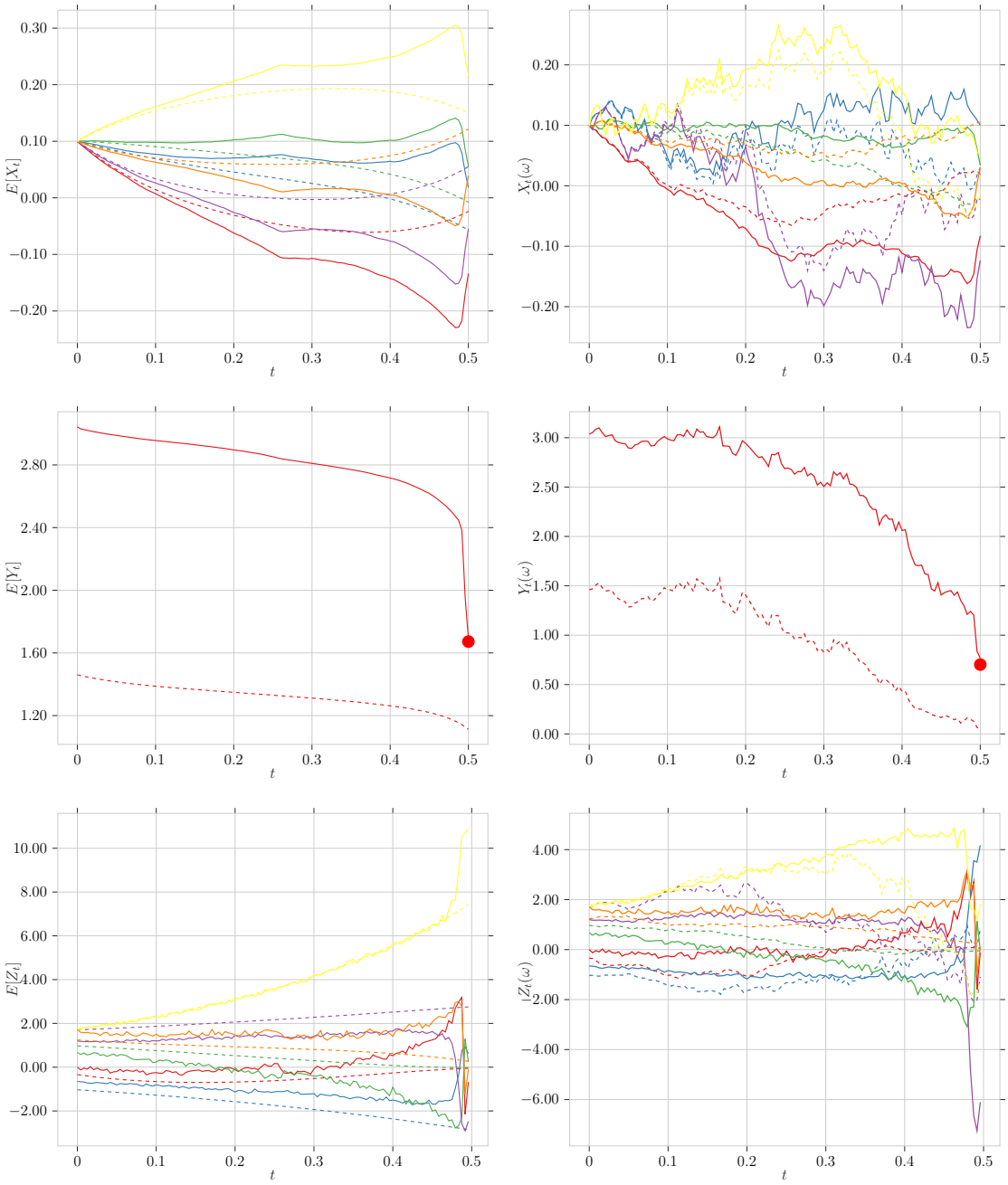

Figure 4.5: Results for the Deep FBSDE method with $d = 6$, $\ell = 2$. All six components of the (approximated and analytical) state process and control process are shown in the top and last row, respectively. The legend for these are omitted to avoid clutter. The middle row illustrates the scalar backward process. The left column shows the expectation and the right column illustrates a single sample path solution.

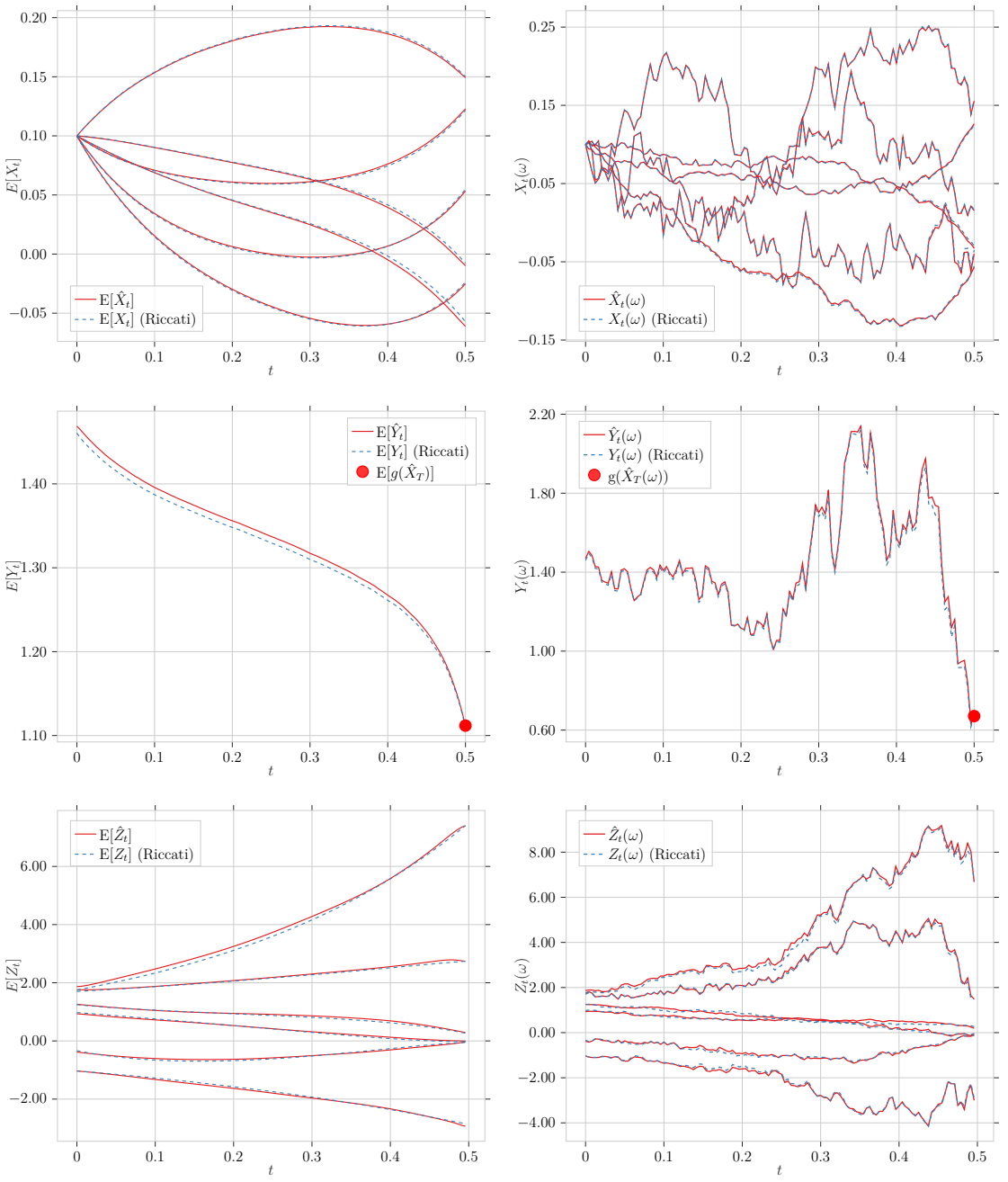

Figure 4.6: Results for the Deep C-FBSDE method with $d = 6$, $\ell = 2$. The top and bottom rows show all six components of the approximated and analytical state and control processes, respectively. The legend for these are omitted to avoid clutter. The middle row illustrates the scalar backward process. The left column shows the expectation, and the right an example sample path.

5

Multi agent setting

In this section, we present the methods used to solve the multi agent system derived in Section 3.2.3. This includes the Deep fictitious play algorithm [5], as well as the novel Robust deep fictitious play. The Deep fictitious play algorithm is based on the Deep FBSDE method, which is generalized to account for N agents. Thus, the Deep fictitious play algorithm also suffers from the flaws described in Section 4.3. The novel algorithm is based on the Deep C-FBSDE method, a more robust approach for solving stochastic control problems. The novel algorithm extends this concept to accommodate N agents.

This chapter is outlined such that we begin by presenting the Deep fictitious play method, the variational formulation for a single batch and player, and the implementation of it. This includes a pseudo code algorithm that outlines the steps needed to solve such a problem. Then, we present the novel algorithm Robust deep fictitious play, the variational formulation, and its algorithm. Finally, a numerical example is presented to quantify the performance of both algorithms. The goal of which, is to see that the novel algorithm successfully solves problems that the Deep fictitious play algorithm does not. For the numerical example, we derive an analytical solution of coupled Riccati equations for LQG differential games.

5.1 Deep fictitious play

The Deep fictitious play method is based on the Deep FBSDE method. It decouples each player's FBSDE into an individual decision problem. Therefore, the solution iterates over each player and uses the Deep FBSDE method to compute the entire strategy profile. To formulate this, we let $(\Omega, \mathcal{F}, (\mathcal{F}_t)_{t\in[0,T]}, \mathbb{P})$ denote the complete filtered probability space. To present the variational formulation for a single player, we let player i apply the strategy $\beta = (\beta_t)_{t\in[0,T]}$ and the rest $\mathbf{u}^{-i}$. To ease the notation, we let $\mathbf{X} \coloneqq \mathbf{X}^{\beta,\mathbf{u}^{-i}}$ and instead attach (y_0^i, ζ^i). This denotes the parameterized initial condition for the backward equation and the parameterized Markov map $\zeta^i : [0,T] \times \mathbb{R}^d \to \mathbb{R}^k$ for player i, to the notation as in Section 4.1. This is to underline that the minimization for player i is performed with respect to these free

variables. Then, for player i, we get the variational formulation

$$\underset{y_0^i,\,\zeta^i}{\text{minimize}} \quad \mathbb{E}\Big[|g^i(\mathbf{X}_T^{y_0^i,\zeta^i}) - Y_T^{i,y_0^i,\zeta^i}|^2\Big], \tag{5.1a}$$

$$\text{subject to} \quad \mathbf{X}_t^{y_0^i,\zeta^i} = \mathbf{x}_0 + \int_0^t \bar{\mathbf{b}}(s, \mathbf{X}_s^{y_0^i,\zeta^i}, Z_s^{i,y_0^i,\zeta^i}, \mathbf{Z}_s^{-i})\,\mathrm{d}s + \int_0^t \Sigma(s, \mathbf{X}_s^{y_0^i,\zeta^i})\,\mathrm{d}W_s, \tag{5.1b}$$

$$Y_t^{i,y_0^i,\zeta^i} = y_0 - \int_0^t \bar{f}^i(s, \mathbf{X}_s^{y_0^i,\zeta^i}, Z_s^{i,y_0^i,\zeta^i}, \mathbf{Z}_s^{-i})\,\mathrm{d}s + \int_0^t (Z_s^{i,y_0^i,\zeta^i})^\top\,\mathrm{d}W_s, \tag{5.1c}$$

$$Z_t^{i,y_0^i,\zeta^i} = \zeta^i(t, \mathbf{X}_t^{y_0^i,\zeta^i}), \tag{5.1d}$$

$$\mathbf{Z}_t^{-i} = \zeta^{-i}(t, \mathbf{X}_t^{y_0^i,\zeta^i}), \tag{5.1e}$$

where $y_0 \in \mathbb{R}$ and $\zeta \in C^{1,1}([0,T] \times \mathbb{R}^{dN})$. Note that the forward equation (5.1b) describes all players while the backward equation (5.1c) only describes the backward process for player i. This is due to the problem being closed-loop. Changing player i strategy affects and influences all other players. Therefore, we need to track what happens with the remaining players $j \neq i$. The individual drift coefficient (3.23) takes the entire state process and control process for all players as an input, which is the reason for computing the forward equation for all players to solve player i's decision problem.

Once again, consider a time grid $0 = t_0 < t_1 < \cdots < t_{\tilde{N}} = T$, which is a partition of the interval $[0,T]$. As the method is based upon the Deep FBSDE method, the free variables of player i are the initial value of the backward equation y_0^i, and the Markov map ζ^i. These are approximated with the neural networks ψ^{i,y_0^i} and ψ^{i,ζ^i} with parameters $\theta^{y_0^i}$ and θ^{ζ^i}, respectively. Since player i depends on the Markov maps of the other players, the idea is to consider these as given. Thus, the algorithm is iterated over several stages, denoted as N_{stage}. For each stage α, the training data is divided into batches of size M_{batch}, where the neural network of player i is updated for each batch. Meanwhile, to approximate the strategy of the remaining players, the networks $\Psi^{-i,y_0} = (\psi^{y_0^1}, \ldots, \psi^{y_0^{i-1}}, \psi^{y_0^{i+1}}, \ldots, \psi^{y_0^N})$ and $\Psi^{-i,\zeta} = (\psi^{\zeta^1}, \ldots \psi^{\zeta^{i-1}}, \psi^{\zeta^{i+1}}, \ldots, \psi^{\zeta^N})$ in the previous stage $\alpha - 1$ are used. Intuitively, the players are playing the game N_{stage} times. Player i chooses the strategy at stage α based on how the remaining players choose to act in the previous round, $\alpha - 1$. Further, the networks for the initial condition are defined as (2.3), while the network approximating the Markov maps has time embedding incorporated in them. Then, we let $\Delta W(m)$ be a vector of $\tilde{N} - 1$ Brownian increment realizations, for $m = 1, \ldots, M_{\text{train}}$. To simplify the notation, we define $\theta^{i,\alpha} \coloneqq (\theta^{y_0^i,\alpha}, \theta^{\zeta^i,\alpha})$. Then, the fully discretized and implementable scheme for player i at stage α, for a single

batch, is given by

$$\underset{\theta^{i,\alpha}}{\text{minimize}} \quad \mathcal{L}(\theta^{i,\alpha}) = \frac{1}{M_{\text{batch}}} \sum_{m=1}^{M_{\text{batch}}} |g^i(\hat{\mathbf{X}}_{\tilde{N}}^{\theta^{i,\alpha}}(m)) - \hat{Y}_{\tilde{N}}^{i,\theta^{i,\alpha}}(m)|^2, \tag{5.2a}$$

$$\begin{aligned}\text{subject to} \quad \hat{\mathbf{X}}_n^{\theta^{i,\alpha}}(m) &= \mathbf{x}_0 + \sum_{k=0}^{n-1} \bar{\mathbf{b}}(t_k, \hat{\mathbf{X}}_k^{\theta^{i,\alpha}}(m), \hat{Z}_k^{i,\theta^{i,\alpha}}(m), \hat{\mathbf{Z}}_k^{-i,\theta^{i,\alpha}}(m))\Delta t_k \\ &\quad + \sum_{k=0}^{n-1} \Sigma(t_k, \hat{\mathbf{X}}_k^{\theta^{i,\alpha}}(m))\Delta W_k(m), \end{aligned} \tag{5.2b}$$

$$\begin{aligned}\hat{Y}_n^{i,\theta^{i,\alpha}}(m) &= y_0 - \sum_{k=0}^{n-1} \bar{f}^i(t_k, \hat{\mathbf{X}}_k^{\theta^{i,\alpha}}(m), \hat{Z}_k^{i,\theta^{i,\alpha}}(m), \hat{\mathbf{Z}}_k^{-i,\theta^{i,\alpha}}(m))\, \Delta t_k \\ &\quad + \sum_{k=0}^{n-1} (\hat{Z}_k^{i,\theta^{y_0},\theta^{\zeta}}(m))^{\top}\, \Delta W_k, \end{aligned} \tag{5.2c}$$

$$\hat{Z}_k^{i,\theta^{i,\alpha}}(m) = \psi^{\zeta^i}(t_k, \hat{\mathbf{X}}_k^{\theta^{i,\alpha}}(m) \mid \theta^{\zeta^i,\alpha}), \tag{5.2d}$$

$$\hat{\mathbf{Z}}_k^{-i,\theta^{i,\alpha}}(m) = \Psi^{-i,\zeta}(t_k, \hat{\mathbf{X}}_k^{\theta^{i,\alpha}}(m) \mid \theta^{\zeta^{-i},\alpha-1}), \tag{5.2e}$$

$$y_0^i(m) = \psi^{y_0^i}(\hat{X}_0^{i,\theta^{i,\alpha}}(m) | \theta^{y_0^i,\alpha}). \tag{5.2f}$$

The algorithm that solves this problem is presented below. It iterates through N_{stages} where for each stage it iterates through all of the players. For each of these iterations the i: the player is the so-called active player whose network we are optimizing. For this player, we divide the Brownian increments into batches and solve the discretized variational formulation (5.2). We are now ready to present the Deep fictitious play algorithm, Algorithm 3.

Algorithm 3: Deep fictitious play

Input : Initialized neural networks $\psi^{y_0^i}$ and ψ^{ζ^i} with paramters $\theta^{y_0^i}$ and θ^{ζ^i} for all $i \in \mathcal{I}$.
Data set of M_{train} Brownian motion paths $\Delta W = \{\Delta W_n\}_{n=1}^{N-1}$.
Batch size M_{batch}, which gives $N_{\text{batch}} = M_{\text{train}}/M_{\text{batch}}$ batches.
Number of stages N_{stage}.

Output: Trained neural networks Ψ^{y_0} and Ψ^{ζ}.

for $\alpha \leftarrow 1, \dots, N_{\text{stage}}$ **do**
  for $i \leftarrow 1, \dots, N$ **do**
    Set $\theta^{y_0^i,\alpha} \leftarrow \theta^{y_0^i,\alpha-1}$.
    Set $\theta^{\zeta^i,\alpha} \leftarrow \theta^{\zeta^i,\alpha-1}$.
    for $k \leftarrow 1, \cdots, N_{\text{batch}}$ **do**
      for $m \leftarrow 1, \dots, M_{\text{batch}}$ **do**
        Set $\hat{\mathbf{X}}_0^{\theta^{i,\alpha}}(m) = \mathbf{x}_0$.
        Approximate initial condition y_0^i using the neural network ψ^{i,y_0}, $y_0^i(m) = \psi^{y_0^i}(\hat{X}_0^{i,\theta^{i,\alpha}}(m) \mid \theta^{y_0^i,\alpha})$.
        for $n \leftarrow 0, \dots, N-1$ **do**
          Approximate the Markov map $Z_n^{i,\theta^{i,\alpha}}(m)$ for player i using the neural network ψ^{ζ^i}, $\hat{Z}_n^{i,\theta^{i,\alpha}}(m) = \psi^{\zeta^i}(t_n, \hat{\mathbf{X}}_0^{\theta^{i,\alpha}}(m) \mid \theta^{\zeta^i,\alpha})$.
          Approximate the remaining $\mathbf{Z}_n^{-i,\theta}(m)$ using the networks $\Psi^{-i,\zeta}$ at the previous stage $\alpha - 1$, i.e. $\hat{\mathbf{Z}}_n^{-i,\theta^{i,\alpha}}(m) = \Psi^{-i,\zeta}(t_n, \hat{\mathbf{X}}_0^{\theta^{i,\alpha}}(m) \mid \boldsymbol{\theta}^{\zeta^{-i},\alpha-1})$.
          Compute $\hat{\mathbf{X}}_n^{\theta^{i,\alpha}}(m)$ according to (5.2b).
          Compute $\hat{Y}_n^{i,\theta^{i,\alpha}}(m)$ according to (5.2c).
        end
      end
      Compute loss $\mathcal{L}(\theta^{i,\alpha})$ for batch k for player i according to (5.2a) and take optimization step according to chosen optimization algorithm, $\theta^{i,\alpha} \leftarrow \arg\min_{\theta^{i,\alpha}} \mathcal{L}(\theta^{i,\alpha})$.
    end
  end
end

return Trained neural networks, $\Psi^{y_0} = (\psi^{y_0^1,N_{\text{Stage}}} \dots, \psi^{y_0^N,N_{\text{stage}}})$ and $\Psi^{\zeta} = (\psi^{\zeta^1,N_{\text{stage}}}, \dots, \psi^{\zeta^N,N_{\text{stage}}})$.

5.2 Robust Deep fictitious play

The novel solution methodology Robust deep fictitious play is, in contrast to Deep fictitious play, based on the Deep C-FBSDE method, which is a robust version of the Deep FBSDE method. This implies that each player has only the Markov map as a free parameter, compared to the additional parametrization of the initial value of the backward equation in Deep fictitious play. The fundamental idea of fictitious play are still conserved in the Robust Deep fictitious play algorithm. The N player problem are decoupled to a single-agent optimization problem, by considering the strategy of the inactive players as given. The variational formulation for player i, defined on the complete filtered probability space $(\Omega, \mathcal{F}, (\mathcal{F}_t)_{t\in[0,T]}, \mathbb{P})$, is given by

$$
\begin{aligned}
\underset{\zeta}{\text{minimize}} \quad & \mathbb{E}\left[\mathcal{Y}_0^{i,\zeta^i}\right] + \lambda \mathrm{Var}\left[\mathcal{Y}_0^{i,\zeta^i}\right], \\
\text{subject to} \quad \mathbf{X}_t^{\zeta^i} &= \mathbf{x}_0 + \int_0^t \bar{\mathbf{b}}(s, \mathbf{X}_s^{\zeta^i}, Z_s^{i,\zeta}, \mathbf{Z}_s^{-i,\zeta^i})\,\mathrm{d}s + \int_0^t \Sigma(s, \mathbf{X}_s^{\zeta^i})\,\mathrm{d}W_s, \\
Y_t^{i,\zeta^i} &= \mathbb{E}\left[\mathcal{Y}_0^{i,\zeta^i}\right] - \int_0^t \bar{f}^i(s, \mathbf{X}_s^{i,\zeta^i}, Z_s^{i,\zeta^i}, \mathbf{Z}_s^{-i,\zeta^i})\,\mathrm{d}s + \int_0^t (Z_s^{i,\zeta^i})^\top\,\mathrm{d}W_s, \\
\mathcal{Y}_0^{i,\zeta^i} &= g(\mathbf{X}_T^{\zeta^i}) + \int_0^T \bar{f}^i(s, \mathbf{X}_s^{i,\zeta^i}, Z_s^{i,\zeta^i}, \mathbf{Z}_s^{-i,\zeta^i})\,\mathrm{d}s - \int_0^T (Z_s^{i,\zeta^i})^\top\,\mathrm{d}W_s, \\
Z_t^{i,\zeta^i} &= \zeta^i(t, \mathbf{X}_t^{\zeta^i}), \\
\mathbf{Z}_t^{-i,\zeta^i} &= \zeta^{-i}(t, \mathbf{X}_t^{\zeta^i}).
\end{aligned}
$$

To formulate fully discrete scheme of the Deep fictitious play algorithm, we consider a partition of the interval $[0, T]$, that is $0 = t_0 < t_1 < \cdots < t_{\tilde{N}} = T$, with $\Delta t_k = t_{k+1} - t_k$. The expectation are approximated using Monte-Carlo estimations, and neural networks are used for parametrization of Markov maps. This gives

$$
\begin{aligned}
\underset{\theta^{i,\zeta}}{\text{minimize}} \quad \mathcal{L}(\theta^{i,\alpha}) \quad &= \frac{1}{\hat{M}_{\text{batch}}} \sum_{m=1}^{\hat{M}_{\text{batch}}} \hat{\mathcal{Y}}_0^{i,\theta^{i,\alpha}}(m) \\
&\quad + \frac{1}{\hat{M}_{\text{batch}}} \sum_{m=\hat{M}_{\text{batch}}+1}^{2\hat{M}_{\text{batch}}} \lambda \left| g^i\left(\hat{\mathbf{X}}_N^{\theta^{i,\alpha}}(m)\right) - \hat{Y}_N^{i,\theta^{i,\alpha}}(m) \right|^2 && \text{(5.4a)} \\
\text{subject to} \quad \hat{\mathbf{X}}_n^{\theta^{i,\alpha}}(m) \quad &= \mathbf{x}_0 + \sum_{k=0}^{n-1} \bar{\mathbf{b}}\left(t_k, \hat{\mathbf{X}}_k^{\theta^{i,\alpha}}(m), \hat{Z}_k^{i,\theta^{i,\alpha}}(m), \hat{\mathbf{Z}}_k^{-i,\theta^{i,\alpha}}(m)\right) \Delta t_k \\
&\quad + \sum_{k=0}^{n-1} \Sigma\left(t_k, \hat{\mathbf{X}}_k^{\theta^{i,\alpha}}(m)\right) \Delta W_k(m), && \text{(5.4b)} \\
\hat{Y}_n^{i,\theta^{i,\alpha}}(m) \quad &= \frac{1}{\hat{M}_{\text{batch}}} \sum_{r=1}^{\hat{M}_{\text{batch}}} \hat{\mathcal{Y}}_0^{i,\theta^{i,\alpha}}(r) \\
&\quad - \sum_{k=0}^{n-1} \bar{f}^i\left(t_k, \hat{\mathbf{X}}_t^{\theta^{i,\alpha}}(m), \hat{Z}_k^{i,\theta^{i,\alpha}}(m), \hat{\mathbf{Z}}_k^{-i,\theta^{i,\alpha}}(m)\right) \Delta t_k \\
&\quad + \sum_{k=0}^{n-1} (\hat{Z}_k^{i,\theta^{i,\alpha}}(m))^\top \Delta W_k(m), && \text{(5.4c)} \\
\hat{\mathcal{Y}}_0^{i,\theta^{i,\alpha}}(m) \quad &= g^i(\hat{\mathbf{X}}_N^{\theta^{i,\alpha}}(m)) \\
&\quad + \sum_{k=0}^{N-1} \bar{f}^i\left(t_k, \hat{\mathbf{X}}_k^{\theta^{i,\alpha}}(m), \hat{Z}_k^{i,\theta^{i,\alpha}}(m), \hat{\mathbf{Z}}_k^{-i,\theta^{i,\alpha}}(m)\right) \Delta t_k \\
&\quad - \sum_{k=0}^{N-1} (\hat{Z}_k^{\theta^{i,\alpha}}(m))^\top \Delta W_k(m), && \text{(5.4d)} \\
\hat{Z}_k^{i,\theta^{i,\alpha}}(m) \quad &= \psi^i(t_k, \hat{\mathbf{X}}_k^{\theta^{i,\alpha}}(m) \mid \theta^{i,\alpha}), && \text{(5.4e)} \\
\hat{\mathbf{Z}}_k^{-i,\theta^{i,\alpha}}(m) \quad &= \Psi^{-i}(t_k, \hat{\mathbf{X}}_k^{\theta^{i,\alpha}}(m) \mid \boldsymbol{\theta}^{-i,\alpha-1}), && \text{(5.4f)}
\end{aligned}
$$

where Ψ^ζ is a family of neural networks $\Psi^\zeta := (\psi^{1,\zeta}, \ldots, \psi^{N,\zeta})$, each defined as (2.3) with time embedding. Next, we present the algorithm that describes how the system (5.4). In short, the discretized problem is solved using the Deep C-FBSDE method. This process is then iterated for each batch and each player, since only *active player* is optimized. The entire process is then iterated for several stages until convergence. With this system in mind, we present the novel Robust Deep fictitious play algorithm, Algorithm 4.

Algorithm 4: Robust deep fictitious play

Input : Initialized neural networks ψ^i with parameters $\theta^{i,0}$ for all $i \in \mathcal{I}$.
Data set of M_{train} Brownian motion paths $\Delta W = \{\Delta W_{t_i}\}_{i=1}^n$.
Batch size M_{batch}, which gives $N_{\text{batch}} = M_{\text{train}}/M_{\text{batch}}$ batches.
Number of stages N_{stage}.

Output: Trained neural network Ψ^ζ.

for $\alpha \leftarrow 1, \dots, N_{\text{stage}}$ **do**
  for $i \leftarrow 1, \dots, N$ **do**
    Set $\theta^{i,\alpha} \leftarrow \theta^{i,\alpha-1}$
    for $k \leftarrow 1, \cdots, N_{\text{batch}}$ **do**
      for $m \leftarrow 1, \dots, \hat{M}_{\text{batch}}$ **do**
        Set $\hat{\mathbf{X}}_0^{\theta^{i,\alpha}}(m) = \mathbf{x}_0$.
        for $n \leftarrow 0, \dots, \tilde{N} - 1$ **do**
          Approximate the Markov map $Z_n^{i,\theta^{i,\alpha}}(m)$ for player i using the neural network ψ^i, i.e. $\hat{Z}_n^{i,\theta^\zeta}(m) = \psi^{i,\zeta}(t_n, \hat{\mathbf{X}}_n^{i,\theta^\zeta}(m) \mid \theta^{i,\zeta,\alpha})$.
          Approximate the remaining $\mathbf{Z}_n^{-i,\theta^{i,\alpha}}(m)$ using the networks Ψ^{-i} at the previous stage $\alpha - 1$, i.e. $\hat{\mathbf{Z}}_n^{-i,\theta^{i,\alpha}}(m) = \Psi^{-i}(t_n, \hat{\mathbf{X}}_n^{\theta^{i,\alpha}}(m) \mid \boldsymbol{\theta}^{-i,\alpha-1})$.
          Compute $\hat{\mathbf{X}}_{n+1}^{\theta^{i,\alpha}}(m)$ according to (5.4b).
        end
        Compute $\hat{\mathcal{Y}}_0^{\theta^{i,\alpha}}(m)$ according to (5.4d).
      end
      for $m \leftarrow \hat{M}_{\text{Batch}} + 1, \dots, 2\hat{M}_{\text{Batch}}$ **do**
        Set $\hat{\mathbf{X}}_0^{\theta^{i,\alpha}}(m) = \mathbf{x}_0$.
        Set $\hat{Y}_0^{\theta^{i,\alpha}}(m) = \frac{1}{\hat{M}_{\text{Batch}}} \sum_{m=1}^{\hat{M}_{\text{Batch}}} \hat{\mathcal{Y}}_0^{\theta^{i,\alpha}}$.
        for $n \leftarrow 0, \dots, \tilde{N} - 1$ **do**
          Approximate the Markov map $Z_n^{i,\theta^{i,\alpha}}(m)$ for player i using the neural network ψ^i, i.e. $\hat{Z}_n^{i,\theta^{i,\alpha}}(m) = \psi^i(t_n, \hat{\mathbf{X}}_n^{i,\theta^{i,\alpha}}(m) \mid \theta^{i,\alpha})$.
          Approximate the remaining $\hat{\mathbf{Z}}_n^{-i,\theta^{i,\alpha}}(m)$ using the networks Ψ^{-i} at the previous stage $\alpha - 1$, i.e. $\hat{\mathbf{Z}}_n^{-i,\theta^{i,\alpha}}(m) = \Psi^{-i}(t_n, \hat{\mathbf{X}}_n^{-i,\theta^{i,\alpha}}(m) \mid \boldsymbol{\theta}^{-i,\alpha-1})$.
          Compute $\hat{\mathbf{X}}_{n+1}^{\theta^{i,\alpha}}(m)$ according to (5.4b).
          Compute $\hat{Y}_{n+1}^{i,\theta^{i,\alpha}}(m)$ according to (5.4c).
        end
      end
      Compute loss $\mathcal{L}(\theta^{i,\alpha})$ for batch k according to (5.4a) and take optimization step according to chosen optimization algorithm, $\theta^{\zeta,\alpha} \leftarrow \arg\min_{\theta^\zeta} \mathcal{L}(\theta^{\zeta,\alpha})$.
    end
  end
end

return Trained neural networks $\Psi^\zeta = (\psi^{\theta^{1,N_{\text{Stage}}}}, \dots, \psi^{\theta^{N,N_{\text{Stage}}}})$.

6
Numerical examples

This section presents numerical examples and the results generated from both algorithms for two problems. That is, the Deep fictitious play algorithm as well as the novel Robust deep fictitious play algorithm. The first numerical example describes the inter-bank borrowing and lending of money. The forward equation for this numerical example is of mean-reverting type, meaning that the state process reverts to its mean. The second numerical example is a LQG game, a generalization of Section 4.4 for multiple agents. Finally, we design a numerical example dedicated to show the capabilities of the novel algorithm. This problem describes a swarm of autonomous drones and is designed to have an intuitive solution.

6.1 Inter-bank borrowing and lending

The inter-bank borrowing and lending stochastic differential game describes the cash flow between banks and is featured in a number of papers, see e.g. [5] and [23]. We let $(\Omega, \mathcal{F}, (\mathcal{F}_t)_{t\in[0,T]}, \mathbb{P})$ be a complete filtered probability space. Further, we let the process $\mathbf{X}^{\mathbf{u}} = (\mathbf{X}_t^{\mathbf{u}})_{t\in[0,T]}$ denote the states of all N players, and $\bar{\mathbf{X}}_t^{\mathbf{u}}$ denote the mean of all states at time t. For $a \in [0,\infty)$ and $\rho \in (-1,1)$, the system for player i is given by

$$\mathrm{d}X_t^{i,\mathbf{u}} = \left(a(\bar{\mathbf{X}}_t^{\mathbf{u}} - X_t^{\mathbf{u}}) + u^i\right)\,\mathrm{d}t + \left(\rho\,\mathrm{d}W_t^0 + \sqrt{1-\rho^2}\,\mathrm{d}W_t^i\right),\ \mathbf{X}^{\mathbf{u}} = \mathbf{x}_0, \tag{6.1a}$$

$$J^i(t,\mathbf{x}) = \mathbb{E}\left[\int_t^T f^i\left(s, \mathbf{X}_s^{\mathbf{u}}, u^i, \mathbf{u}_s^{-i}\right)\,\mathrm{d}s + g^i(\mathbf{X}_T^{\mathbf{u}})\middle|\mathbf{X}_t^{\mathbf{u}} = \mathbf{x}\right], \tag{6.1b}$$

where $W^0 = (W_t^0)_{t\in[0,T]}$ is a Brownian motion representing the so-called common noise, experienced by all players. In contrast, $W^i = (W_t^i)_{t\in[0,T]}$ represents the individual noise for player i. Furthermore, $a \geq 0$ describes the strength of the mean-reverting property of the system. The running cost $f^i : [0,T]\times\mathbb{R}^N\times\mathbb{U}^i\times\mathbb{U}\to\mathbb{R}$ and terminal cost $g^i : \mathbb{R}^N \to \mathbb{R}$ are both part of the families $\mathbf{f} = (f^1,\ldots,f^N)$ and $\mathbf{g} = (g^1,\ldots,g^N)$, respectively. Further, the running and terminal cost for player i are defined by

$$f^i(t,\mathbf{x}) = \frac{1}{2}(u^i)^2 - qu^i(\bar{\mathbf{x}} - x^i) + \frac{\varepsilon}{2}(\bar{\mathbf{x}} - x^i)^2,$$
$$g^i(\mathbf{x}) = \frac{c}{2}(\bar{\mathbf{x}} - x^i)^2,$$

where $q, \varepsilon, c > 0$. To ensure connectivity of the running cost, we assume that $q^2 \leq \varepsilon$. The control $u^i = (u_t^i)_{t\in[0,T]}$ controls the borrowing and lending to a central bank for

bank i. Thus, the parameter q can be seen as an incentive to borrow or lend for bank i since if the state is larger than the mean the bank will want to lend money out and vice versa. Further, we note that both ε and c ensure that the state does not stray away from the mean.

The system (6.1) is reformulated from an algorithm point of view. Thus, let $\beta = (\beta_t)_{t\in[0,T]}$ be the control applied by the player i. Then, let $\mathbf{X}^{\beta,\mathbf{u}^{-i}}$ be the state of all players, where player i applies control β, and the remaining players $\mathbf{u}^{-i}$. The system of player i is given by

$$\begin{aligned}
\mathrm{d}\mathbf{X}_t^{\beta,\mathbf{u}^{-i}} &= \Big(a(\bar{\mathbf{X}}_t^{\beta,\mathbf{u}^{-i}} - X_t^{i,\beta,\mathbf{u}^{-i}}) + \beta\Big)\ \mathrm{d}t + \Sigma\,\mathrm{d}\mathbf{W}_t,\ \mathbf{X}_0^{\beta,\mathbf{u}^{-i}} = \mathbf{x}_0,\\
J^i(t,\mathbf{x}) &= \mathbb{E}\left[\int_t^T f^i\left(s, \mathbf{X}_s^{\beta,\mathbf{u}^{-i}}, \beta_s, \mathbf{u}_s^{-i}\right)\,\mathrm{d}s + g^i(\mathbf{X}_T^{\beta,\mathbf{u}^{-i}})\middle|\mathbf{X}_t^{\beta,\mathbf{u}^{-i}} = \mathbf{x}\right],
\end{aligned}$$

where

$$\mathbf{W}_t = (W_t^0, W_t^1, \dots, W_t^N)^\top, \qquad \Sigma = \begin{pmatrix} \rho & \sqrt{1-\rho^2} & 0 & \dots & 0\\ \rho & 0 & \sqrt{1-\rho^2} & \dots & 0\\ \vdots & \vdots & \vdots & \ddots & \vdots\\ \rho & 0 & 0 & \dots & \sqrt{1-\rho^2}\end{pmatrix}.$$

As in Section 4.4, we omit the derivation of the corresponding HJB equation and only state the final formulation

$$\begin{aligned}
&\partial_t V^i + \inf_{\beta\in\mathcal{U}}\left(\Big(a(\bar{\mathbf{x}} - x^i) + \beta_t\Big)\nabla_x V^i + \frac{1}{2}\beta^2 - q\beta(\bar{\mathbf{x}} - \mathbf{x}^i) + \frac{\varepsilon}{2}(\bar{\mathbf{x}} - \mathbf{x}^i)^2\right) \qquad (6.3)\\
&\qquad + \frac{1}{2}\mathrm{Tr}\left(\Sigma\Sigma^\top \Delta_x V^i\right) = 0,\\
&V^i(T,\mathbf{x}) = \frac{c}{2}(\bar{\mathbf{x}} - x^i)^2,
\end{aligned}$$

where, again, the spacetime dependency $(t,x)\in[0,T]\times\mathbb{R}^d$ is omitted unless otherwise specified. In (6.3), the infimum gives $\beta = \kappa(t,x,\nabla_x V^i) = q(\bar{\mathbf{x}} - x^i) - \partial_{x^i} V^i$. Moving on, we let the triple $(\mathbf{X}_t^{\beta,\mathbf{u}^{-i}}, Y_t, Z_t) \in \mathbb{S}_T^2(\mathbb{R}^{dN})\times\mathbb{S}_T^2(\mathbb{R})\times\mathbb{H}_T^2(\mathbb{R}^{dN})$ solve the system of FBSDEs given by

$$\begin{aligned}
\mathrm{d}\mathbf{X}_t^{i,\beta,\mathbf{u}^{-i}} &= \Big(a(\bar{\mathbf{X}}_t^{\beta,\mathbf{u}^{-i}} - X_t^{i,\beta,\mathbf{u}^{-i}}) + \beta_t\Big)\ \mathrm{d}t + \Sigma\,\mathrm{d}W_t,\\
Y_t^i &= g^i(\mathbf{X}_T^{\beta,\mathbf{u}^{-i}}) - \int_t^T \bar{f}^i(s, \mathbf{X}_s^{\beta,\mathbf{u}^{-i}}, \beta_s, \mathbf{u})\,\mathrm{d}s + \int_t^T (Z_s)^\top\,\mathrm{d}W_s,
\end{aligned}$$

which is well-posed with the initial condition $\mathbf{X}_0^{\beta,\mathbf{u}^{-i}} = \mathbf{x}_0$ $\mathbb{P}$-a.s. The analytical solution to the problem is given by the function $\eta : [0,T]\to\mathbb{R}$ which satisfies the Riccati equation, for the derivation see [23]

$$\eta_t = \frac{-(\varepsilon - q^2)\left(e^{(\delta^+ - \delta^-)(T-t)} - 1\right) - c\left(\delta^+ e^{(\delta^+ - \delta^-)(T-t)} - \delta^-\right)}{(\delta^- e^{(\delta^+ - \delta^-)(T-t)} - \delta^+) - c(1 - \frac{1}{N^2})(e^{(\delta^+ - \delta^-)(T-t)} - 1)},$$

with $\eta_T = c$ and δ^+, δ^- defined as

$$\delta^{\pm} = -(a+q) \pm \sqrt{(a+q)^2 + \left(1 - \frac{1}{N^2}\right)(\varepsilon - q^2)}.$$

The above follows from the anzats

$$V^i(t, \mathbf{x}) = \frac{\eta(t)}{2}(\bar{\mathbf{x}} - x^i)^2 + \mu(t),$$

for all $i \in \mathcal{I}$, and where

$$\dot{\mu}(t) \coloneqq -\frac{1}{2}(1-\rho^2)(1-\frac{1}{N})\eta(t), \quad \mu(T) = 0.$$

6.1.1 Numerical results

The studied example in this thesis, regarding the Inter-bank borrowing game, aligns with the example presented in [5]. Thus, this is a known example where the Deep fictitious play approach does not fail. The parameters for the forward equation are given by

$$a = 0.1, \qquad \rho = 0.2,$$

and for the backward equation, the used parameters are

$$q = 0.1, \qquad \epsilon = 0.5, \qquad c = 0.5.$$

Figures 6.1 and 6.2 show that both the Deep fictitious play and the robust version solve the system well.

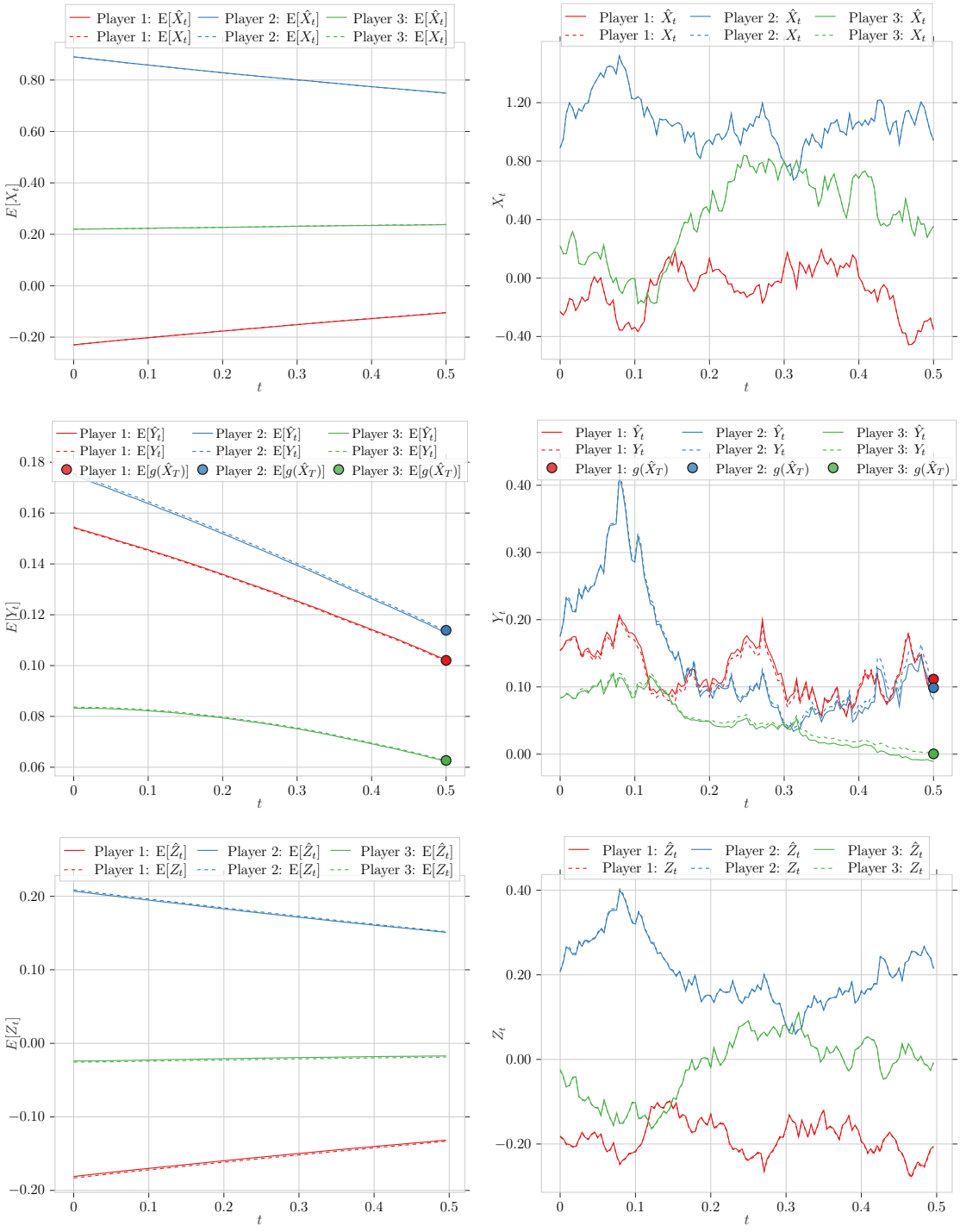

Figure 6.1: Numerical results for the Deep fictitious play method for $d = \ell = 1$ with three players. Columns represent the expectation and one representative trajectory, respectively. The rows represent the different processes. Each color represents a different player. The solid line represents the approximation of the process, and the dashed its analytical counterpart.

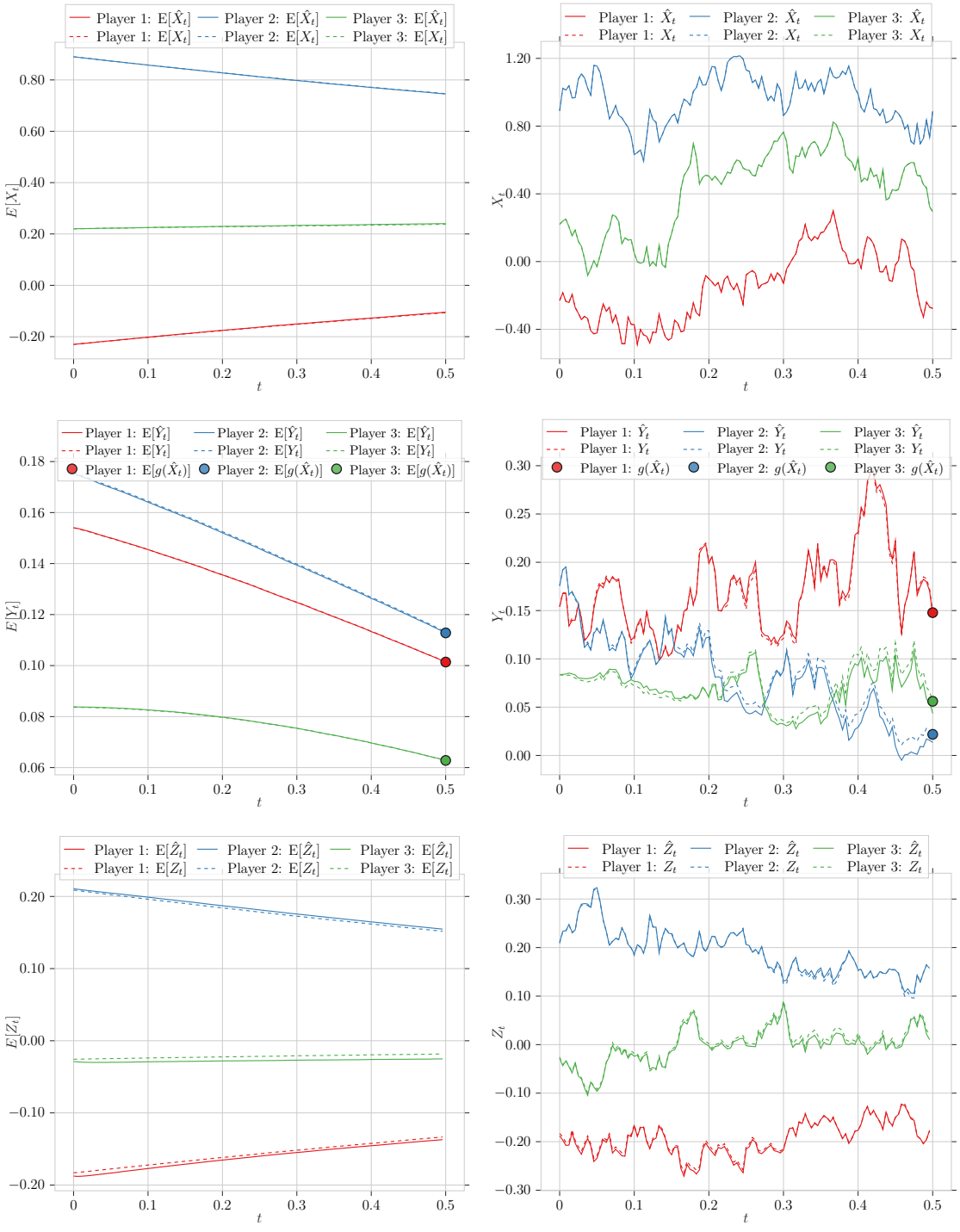

Figure 6.2: Numerical results for the Robust deep fictitious play method for $d = \ell = 1$ with three players. Columns represent the expectation and one representative trajectory, respectively. The rows represent the different processes. Each color represents a different player. The solid line represents the approximation of the process, and the dashed its analytical counterpart.

6.2 Linear–quadratic–Gaussian control game

This section presents the extension of Section 4.4. It is the same type of differential game, i.e. control problem, however, in the case of multiple players. We present numerical results for a two-dimensional case and an example with a six-dimensional state and two-dimensional control for each player. Firstly, we present the problem formulation. Then, we present the corresponding HJB equation, we refer to Appendix A for the derivation. Then we state the corresponding system of FBSDEs and refer to Section 3.2.3 for the derivation. Finally, we derive an analytical solution to quantify the performance of the Deep fictitious play algorithm and its robust counterpart.

Consider an non-cooperative LQG stochastic differential game, defined on the complete filtered probability space $(\Omega, \mathcal{F}, (\mathcal{F}_t)_{t\in[0,T]}, \mathbb{P})$ with the filtration $(\mathcal{F}_t)_{t\in[0,T]}$ generated by the k-dimensional Brownian motion $W = (W_t)_{t\in[0,T]}$, involving $N-$players, each associated with a state process $X^i = (X^i_t)_{t\in[0,T]} \in \mathbb{R}^d$. The full state is the family of each players individual state process, i.e., $\mathbf{X} = (X^1, \dots, X^N)^\top \in \mathbb{R}^{dN}$. The problem for player $i \in \{1, \dots, N\} = \mathcal{I}$ is then defined by

$$\mathrm{d}\mathbf{X}_t^{\beta,\mathbf{u}^{-i}} = \left(\mathcal{A}(\mathcal{C} - \mathbf{X}_t^{\beta,\mathbf{u}^{-i}}) + \mathcal{B}\mathbf{u}_t\right)\mathrm{d}t + \Sigma(t, \mathbf{X}_t^{\beta,\mathbf{u}^{-i}})\,\mathrm{d}W_t, \quad \mathbf{X}_0^{\beta,\mathbf{u}^{-i}} = \mathbf{x}_0, \tag{6.5a}$$

$$\begin{aligned} J^i(t,\mathbf{x}) = \mathbb{E}\Bigg[\int_t^T \left(\langle R_x \mathbf{X}_s^{\beta,\mathbf{u}^{-i}}, \mathbf{X}_s^{\beta,\mathbf{u}^{-i}}\rangle + \langle R^i_{\mathbf{u}}\beta_s, \beta_s\rangle\right)\mathrm{d}s \\ + \langle G^i \mathbf{X}_T^{\beta,\mathbf{u}^{-i}}, \mathbf{X}_T^{\beta,\mathbf{u}^{-i}}\rangle \Big| \mathbf{X}_t^{\beta,\mathbf{u}^{-i}} = \mathbf{x}\Bigg], \end{aligned} \tag{6.5b}$$

where $\mathcal{A} := \operatorname{diag}(A^1, \dots, A^N) \in \mathbb{R}^{dN\times dN}$, $\mathcal{C} := (C^1, \dots, C^N)^\top \in \mathbb{R}^{dN}$ and $\Sigma := \operatorname{diag}(\sigma^1, \dots, \sigma^N) \in \mathbb{R}^{dN\times dN}$ and $\mathcal{B} := (B^1, \dots, B^N) \in \mathbb{R}^{dN\times \ell N}$. In turn the individual matrices are defined as

$$B^i \in \mathbb{R}^{d\times\ell}, \quad C^i \in \mathbb{R}^d, \qquad R_x,\, G^i \in \mathbb{S}^{dN}_+, \qquad R^i_\alpha \in \mathbb{S}^\ell_+, \qquad A^i,\, \sigma^i \in \mathbb{R}^{d\times d},$$

where in turn we again abuse the notation and let $\mathbb{S}^k_+$ be defined as (4.7) for some size $k \in \mathbb{N}$. Moreover, we let $\mathbf{X}_0^{\beta,\mathbf{u}^{-i}} = \mathbf{x}_0$ $\mathbb{P}$-a.s.

For some player $i \in \mathcal{I}$, the corresponding HJB equation of (6.5) is given by

$$\begin{aligned} \partial_t V^i + \inf_{\beta\in\mathcal{U}} \Big(\left(\mathcal{A}(\mathcal{C} - x) + \mathcal{B}^{-i}\mathbf{u}_t^{-i} + \bar{\mathcal{B}}^i\beta_t\right)^\top \nabla_x V^i + \langle R^i_x x, x\rangle + \langle R^i_{\mathbf{u}}\beta_t, \beta_t\rangle\Big) \\ + \frac{1}{2}\operatorname{Tr}(\Sigma\Delta_x V^i \Sigma^\top) = 0, \end{aligned} \tag{6.6a}$$

$$V^i(T,\mathbf{x}) = \langle G^i\mathbf{x}, \mathbf{x}\rangle, \tag{6.6b}$$

where the spacetime dependence $(t,\mathbf{x}) \in [0,T]\times\mathbb{R}^{dN}$ is omitted for brevity unless otherwise specified. The optimal control β^* is derived from the infimum in (6.6a) with respect to β. This yields

$$\beta^* = -\frac{1}{2}(R^i_{\mathbf{u}})^{-1}(\bar{\mathcal{B}}^i)^\top \nabla_x V^i(t,\mathbf{x}), \tag{6.7}$$

which substituted in to (6.6a) yields

$$\begin{aligned}\partial_t V^i + \left(\mathcal{A}(\mathcal{C}-x) + \mathcal{B}^{-i}\mathbf{u}_t^{-i} - \frac{1}{2}\bar{\mathcal{B}}^i(R^i_{\mathbf{u}})^{-1}(\bar{\mathcal{B}}^i)^\top \nabla_x V^i(t,\mathbf{x})\right)^\top \nabla_x V^i \\ + \langle R^i_x x, x\rangle + \frac{1}{4}\nabla_x V^{i,\top}\bar{\mathcal{B}}^i((R^i_{\mathbf{u}})^{-1})^\top \bar{\mathcal{B}}^{i,\top}\nabla_x V^i \\ + \frac{1}{2}\mathrm{Tr}(\Sigma\Delta_x V^i \Sigma^\top) = 0.\end{aligned} \tag{6.8}$$

The corresponding FBSDE for player i is given by

$$\begin{aligned}\mathrm{d}\mathbf{X}_t^{\beta,\mathbf{u}^{-i}} &= \left(\left(\mathcal{A}(C - \mathbf{X}_t^{\beta,\mathbf{u}^{-i}}\right) + \mathcal{B}\mathbf{u}_t\right)\mathrm{d}t + \Sigma(t,\mathbf{X}_t^{\beta,\mathbf{u}^{-i}})\,\mathrm{d}W_t, \qquad \mathbf{X}_0^{\beta,\mathbf{u}^{-i}} = \mathbf{x}_0, \\ \mathrm{d}Y_t^i &= -\left(\langle R^i_x \mathbf{X}_t^{\beta,\mathbf{u}^{-i}}, \mathbf{X}_t^{\beta,\mathbf{u}^{-i}}\rangle + \langle R^i_{\mathbf{u}}\beta_t, \beta_t\rangle\right)\mathrm{d}t + Z_t^i\,\mathrm{d}W_t,\end{aligned}$$

where $Y_T^i = \langle G^i \mathbf{X}_t^{\beta,\mathbf{u}^{-i}}, \mathbf{X}_t^{\beta,\mathbf{u}^{-i}}\rangle$ is the terminal condition for the backward equation. To solve (6.8) analytically, for sufficiently smooth functions $P^i(t) : [0,T] \to \mathbb{R}^{dN\times dN}$, $Q^i(t) : [0,T] \to \mathbb{R}^{dN}$ and $R^i(t) : [0,T] \to \mathbb{R}$, we make the following ansatz

$$V^i(t,x) = x^\top P^i(t)x + x^\top Q^i(t) + R^i(t), \tag{6.10}$$

where we assume that $P^i \in \mathbb{S}^{dN}$ for $\mathbb{S}^k$ as in (4.7). Setting $t = T$ and inserting (6.10) in (6.6b) gives the terminal conditions for each matrix-valued function as

$$P^i(T) = G^i; \quad Q^i(T) = \mathbf{0}_{dN}; \quad R^i(T) = 0.$$

Differentiating the expression yields

$$\begin{aligned}\partial V_t^i(t,\mathbf{x}) &= \mathbf{x}^\top \dot{P}^i(t)\mathbf{x} + \mathbf{x}^\top \dot{Q}^i(t) + \dot{R}^i(t), \\ \nabla_x V^i(t,\mathbf{x}) &= (P^i(t) + P^i(t)^\top)\mathbf{x} + Q^i(t), \\ \Delta_x V^i(t,\mathbf{x}) &= P^i(t) + P^i(t)^\top.\end{aligned}$$

Substituting the ansatz for $V^i(t,x)$, and its derivatives, into the HJB-equation (6.8) with (6.7) yields

$$\begin{aligned}&\mathbf{x}^\top\left(\dot{P}^i - \mathcal{A}^\top\tilde{P}^i + R^i_{\mathbf{x}} - \frac{1}{4}\tilde{P}^{i,\top}\bar{\mathcal{B}}^i((R^i_{\mathbf{u}})^{-1})^\top\bar{\mathcal{B}}^{i,\top}\tilde{P}^i\right)\mathbf{x} \\ &\quad + \left(\dot{Q}^i + \mathcal{C}^\top\mathcal{A}^\top\tilde{P}^i - Q^i\mathcal{A} + \mathbf{u}_t^{-i,\top}\mathcal{B}^{-i,\top}\tilde{P}^i - \frac{1}{2}Q^{i,\top}\bar{\mathcal{B}}^i((R^i_{\mathbf{u}})^{-1})^\top\bar{\mathcal{B}}^{i,\top}\tilde{P}^i\right)\mathbf{x} \\ &\quad + \dot{R}^i + \mathcal{C}^\top\mathcal{A}^\top Q^i + \mathbf{u}_t^{-i,\top}\mathcal{B}^{-i,\top}Q^i - \frac{1}{4}Q^{i,\top}\bar{\mathcal{B}}^i((R^i_{\mathbf{u}})^{-1})^\top\bar{\mathcal{B}}^{i,\top}Q^i + \frac{1}{2}\mathrm{Tr}(\Sigma\tilde{P}^i\Sigma^\top) = 0,\end{aligned} \tag{6.11}$$

where we have eased the clutter by using the defining

$$\tilde{P}^i \coloneqq P^i + P^{i,\top}.$$

Furthermore, we have omitted time dependency for the same reason. Then, to deal with the term containing $\mathbf{u}^{-i}$ we recognize that

$$u^j = -\frac{1}{2}(R^j_{\mathbf{u}})^{-1}\bar{\mathcal{B}}^{j,\top}(\tilde{P}^j\mathbf{x} + Q^j),$$

and therefore, in turn we get

$$\mathbf{u}_t^{-i,\top}\mathcal{B}^{-i,\top}\tilde{P}^i\mathbf{x} = -\frac{1}{2}\mathbf{x}^\top \sum_{j\in\mathcal{I}:\, j\neq i} \tilde{P}^{j\top}\bar{\mathcal{B}}^j((R^j_{\mathbf{u}})^{-1})^\top\bar{\mathcal{B}}^{j\top}\tilde{P}^i\mathbf{x} - \frac{1}{2}\sum_{j\in\mathcal{I},\, j\neq i} Q^{j\top}\bar{\mathcal{B}}^j((R^i_{\mathbf{u}})^{-1})^\top\bar{\mathcal{B}}^{j\top}\tilde{P}^i\mathbf{x}, \tag{6.12}$$

$$\mathbf{u}_t^{-i,\top}\mathcal{B}^{-i,\top}Q^i = -\frac{1}{2}\mathbf{x}^\top \sum_{j\in\mathcal{I},\, j\neq i} \tilde{P}^{j\top}\bar{\mathcal{B}}^j((R^j_{\mathbf{u}})^{-1})^\top\bar{\mathcal{B}}^{j\top}Q^i - \frac{1}{2}\sum_{j\in\mathcal{I},\, j\neq i} Q^{j\top}\bar{\mathcal{B}}^j((R^j_{\mathbf{u}})^{-1})^\top Q^i. \tag{6.13}$$

Inserting the identities (6.12) and (6.13) in (6.11) we get the formulation

$$\begin{aligned}
&\mathbf{x}^\top\Bigg(\dot{P}^i - \mathcal{A}^\top\tilde{P}^i + R^i_{\mathbf{x}} - \frac{1}{4}\tilde{P}^{i,\top}\bar{\mathcal{B}}^i((R^i_{\mathbf{u}})^{-1})^\top\bar{\mathcal{B}}^{i,\top}\tilde{P}^i \\
&\quad -\frac{1}{2}\sum_{j\in\mathcal{I},\, j\neq i}\tilde{P}^{j\top}\bar{\mathcal{B}}^j((R^j_{\mathbf{u}})^{-1})^\top\bar{\mathcal{B}}^{j\top}\tilde{P}^i\Bigg)\mathbf{x} \\
&+\Bigg(\dot{Q}^i + \mathcal{C}^\top\mathcal{A}^\top\tilde{P}^i - Q^i\mathcal{A} - \frac{1}{2}Q^{i,\top}\bar{\mathcal{B}}^i((R^i_{\mathbf{u}})^{-1})^\top\bar{\mathcal{B}}^{i,\top}\tilde{P}^i \\
&\quad -\frac{1}{2}Q^{i\top}\sum_{j\in\mathcal{I},\, j\neq i}\bar{\mathcal{B}}^j((R^j_{\mathbf{u}})^{-1})\bar{\mathcal{B}}^{j\top}\tilde{P}^j - \frac{1}{2}\sum_{j\in\mathcal{I},\, j\neq i}Q^{j\top}\bar{\mathcal{B}}^j((R^j_{\mathbf{u}})^{-1})^\top\bar{\mathcal{B}}^{j\top}\tilde{P}^i\Bigg)\mathbf{x} \\
&+\dot{R}^i + \mathcal{C}^\top\mathcal{A}^\top Q^i - \frac{1}{4}Q^{i,\top}\bar{\mathcal{B}}^i((R^i_{\mathbf{u}})^{-1})^\top\bar{\mathcal{B}}^{i,\top}Q^i \\
&\quad -\frac{1}{2}\sum_{j\in\mathcal{I},\, j\neq i}Q^{j\top}\bar{\mathcal{B}}^j((R^j_{\mathbf{u}})^{-1})^\top\bar{\mathcal{B}}^{j,\top}Q^i + \frac{1}{2}\mathrm{Tr}(\Sigma\tilde{P}^i\Sigma^\top) = 0,
\end{aligned}$$

Extracting terms of equal order with respect to x, which we omit, yields the coupled matrix-valued Riccati ODE

$$\begin{aligned}
&\dot{P}^i - \mathcal{A}^\top\tilde{P}^i + R^i_{\mathbf{x}} - \frac{1}{4}\tilde{P}^{i,\top}\bar{\mathcal{B}}^i((R^i_{\mathbf{u}})^{-1})^\top\bar{\mathcal{B}}^{i,\top}\tilde{P}^i \\
&\quad -\frac{1}{2}\sum_{j\in\mathcal{I},\, j\neq i}\tilde{P}^{j\top}\bar{\mathcal{B}}^j((R^j_{\mathbf{u}})^{-1})^\top\bar{\mathcal{B}}^{j\top}\tilde{P}^i = \mathbf{0}_{dN\times dN}, \\
&\dot{Q}^i + \mathcal{C}^\top\mathcal{A}^\top\tilde{P}^i - Q^i\mathcal{A} - \frac{1}{2}Q^{i,\top}\bar{\mathcal{B}}^i((R^i_{\mathbf{u}})^{-1})^\top\bar{\mathcal{B}}^{i,\top}\tilde{P}^i \\
&\quad -\frac{1}{2}Q^{i\top}\sum_{j\in\mathcal{I},\, j\neq i}\bar{\mathcal{B}}^j((R^j_{\mathbf{u}})^{-1})\bar{\mathcal{B}}^{j\top}\tilde{P}^j - \frac{1}{2}\sum_{j\in\mathcal{I},\, j\neq i}Q^{j\top}\bar{\mathcal{B}}^j((R^j_{\mathbf{u}})^{-1})^\top\bar{\mathcal{B}}^{j\top}\tilde{P}^i = \mathbf{0}_{dN}, \\
&\dot{R}^i + \mathcal{C}^\top\mathcal{A}^\top Q^i - \frac{1}{4}Q^{i,\top}\bar{\mathcal{B}}^i((R^i_{\mathbf{u}})^{-1})^\top\bar{\mathcal{B}}^{i,\top}Q^i \\
&\quad -\frac{1}{2}\sum_{j\in\mathcal{I},\, j\neq i}Q^{j\top}\bar{\mathcal{B}}^j((R^j_{\mathbf{u}})^{-1})^\top\bar{\mathcal{B}}^{j,\top}Q^i + \frac{1}{2}\mathrm{Tr}(\Sigma\tilde{P}^i\Sigma^\top) = 0
\end{aligned}$$

As stated in Section 4.4, the Riccati system is usually solved by common ODE techniques, such as the explicit Euler numerical scheme.

6.2.1 Two-dimensional example

In this section, we consider an LQG example example with a two-dimensional state and two-dimensional control. The game consists of five players, $N = 5$, where the parameters are given by

$$A^1 = A^2 = A^3 = \begin{bmatrix} 1 & 0 \\ 0 & 2 \end{bmatrix}, \qquad A^4 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \qquad A^5 = \begin{bmatrix} 2 & 0 \\ 0 & 1 \end{bmatrix}.$$

Further, for all players $i \in \mathcal{I}$, we have that

$$C^i = \begin{bmatrix} 0.1 \\ 0.2 \end{bmatrix}, \qquad \Sigma^i = \begin{bmatrix} 0.05 & 0.20 \\ 0.05 & 0.25 \end{bmatrix}, \qquad R_u^i = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}.$$

The remaining parameters are different for each player. We have

$$\begin{aligned}
B^1 &= \begin{bmatrix} 1 & 0.5 & 0 & 0.1 & 0 & 0.1 & 0 & 0.1 & 0 & 0.1 \\ -0.5 & 1 & 0.1 & 0 & 0.1 & 0 & 0.1 & 0 & 0.1 & 0 \end{bmatrix}, \\
B^2 &= \begin{bmatrix} 0.1 & 0 & 1 & 0.5 & 0.1 & 0 & 0.1 & 0 & 0.1 & 0 \\ 0 & 0.1 & -0.5 & 1 & 0 & 0.1 & 0 & 0.1 & 0 & 0.1 \end{bmatrix}, \\
B^3 &= \begin{bmatrix} 0.1 & 0 & 0.1 & 0 & 1 & 0.5 & 0.1 & 0 & 0.1 & 0 \\ 0 & 0.1 & 0 & 0.1 & -0.5 & 1 & 0 & 0.1 & 0 & 0.1 \end{bmatrix}, \\
B^4 &= \begin{bmatrix} 0.1 & 0 & 0.1 & 0 & 0.1 & 0 & 1 & 0.5 & 0.1 & 0 \\ 0 & 0.1 & 0 & 0.1 & 0 & 0.1 & -0.5 & 1 & 0 & 0.1 \end{bmatrix}, \\
B^5 &= \begin{bmatrix} 0.1 & 0 & 0.1 & 0 & 0.1 & 0 & 0.1 & 0 & 1 & 0.5 \\ 0 & 0.1 & 0 & 0.1 & 0 & 0.1 & 0 & 0.1 & -0.5 & 1 \end{bmatrix}.
\end{aligned}$$

The penalty matrices in the backward equation are defined as

$$\begin{aligned}
R_x^1 &= \operatorname{diag}(10, 1, 0, 0, 0, 0, 0, 0, 0, 0), & R_x^2 &= \operatorname{diag}(0, 0, 5, 1, 0, 0, 0, 0, 0, 0), \\
R_x^3 &= \operatorname{diag}(0, 0, 0, 0, 1, 1, 0, 0, 0, 0), & R_x^4 &= \operatorname{diag}(0, 0, 0, 0, 0, 0, 3, 1, 0, 0), \\
R_x^5 &= \operatorname{diag}(0, 0, 0, 0, 0, 0, 0, 0, 2, 1),
\end{aligned}$$

and

$$\begin{aligned}
G^1 &= \operatorname{diag}(1, 10, 0, 0, 0, 0, 0, 0, 0, 0), & G^2 &= \operatorname{diag}(0, 0, 1, 5, 0, 0, 0, 0, 0, 0), \\
G^3 &= \operatorname{diag}(0, 0, 0, 0, 1, 1, 0, 0, 0, 0), & G^4 &= \operatorname{diag}(0, 0, 0, 0, 0, 0, 1, 3, 0, 0), \\
G^5 &= \operatorname{diag}(0, 0, 0, 0, 0, 0, 0, 0, 1, 2).
\end{aligned}$$

6.2.1.1 Numerical results

Figure 6.3 illustrates the X-process for the two-dimensional LQG game, and it is evident that the Deep fictitious play algorithm fails to approximate it accurately. The same behavior is observed in Figures 6.4 and 6.5.

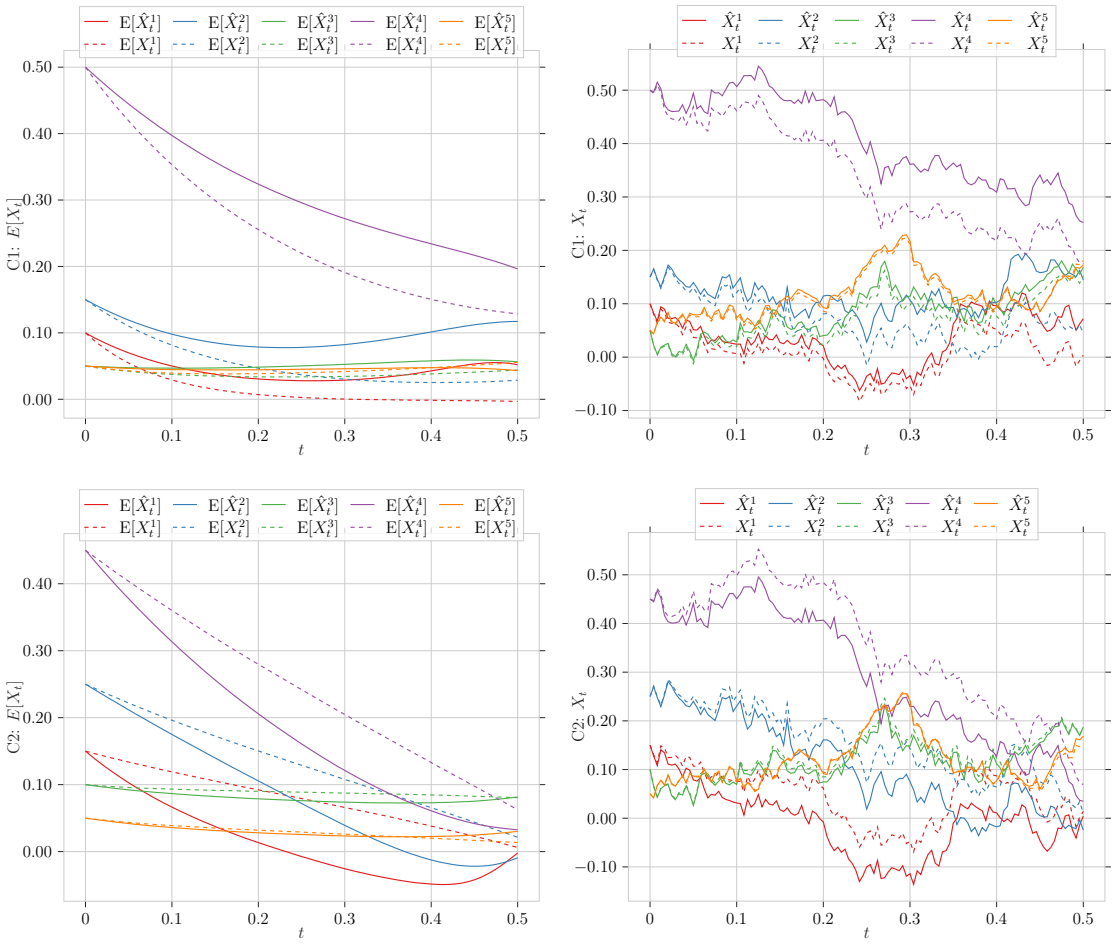

Figure 6.3: Results using the Deep fictitious play method with $d = \ell = 2$ with five players. The top row represents the first component of the state process X^i and its approximation $\hat{X}^i$ for each player $i = 1, \ldots, 5$. The bottom row represents the second component. The left component shows the expectation, and the right is one representative trajectory.

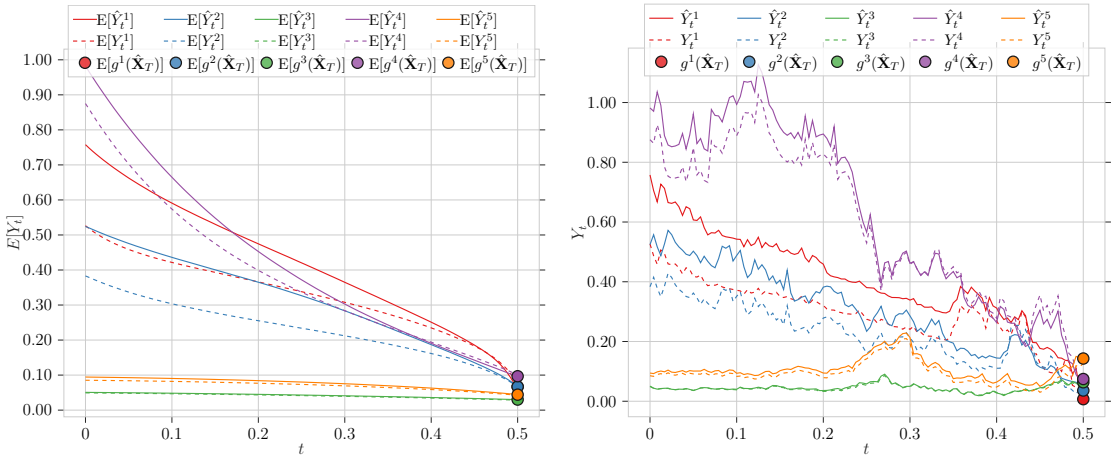

Figure 6.4: Results using the Deep fictitious play method with $d = \ell = 2$ with five players. The left figure shows the expectation of the backward process Y^i and its approximation $\hat{Y}^i$ as well as the approximated terminal condition $g^i(\hat{\mathbf{X}}_T)$ for $i = 1, \ldots, 5$. The right shows one representative trajectory of the same scenario.

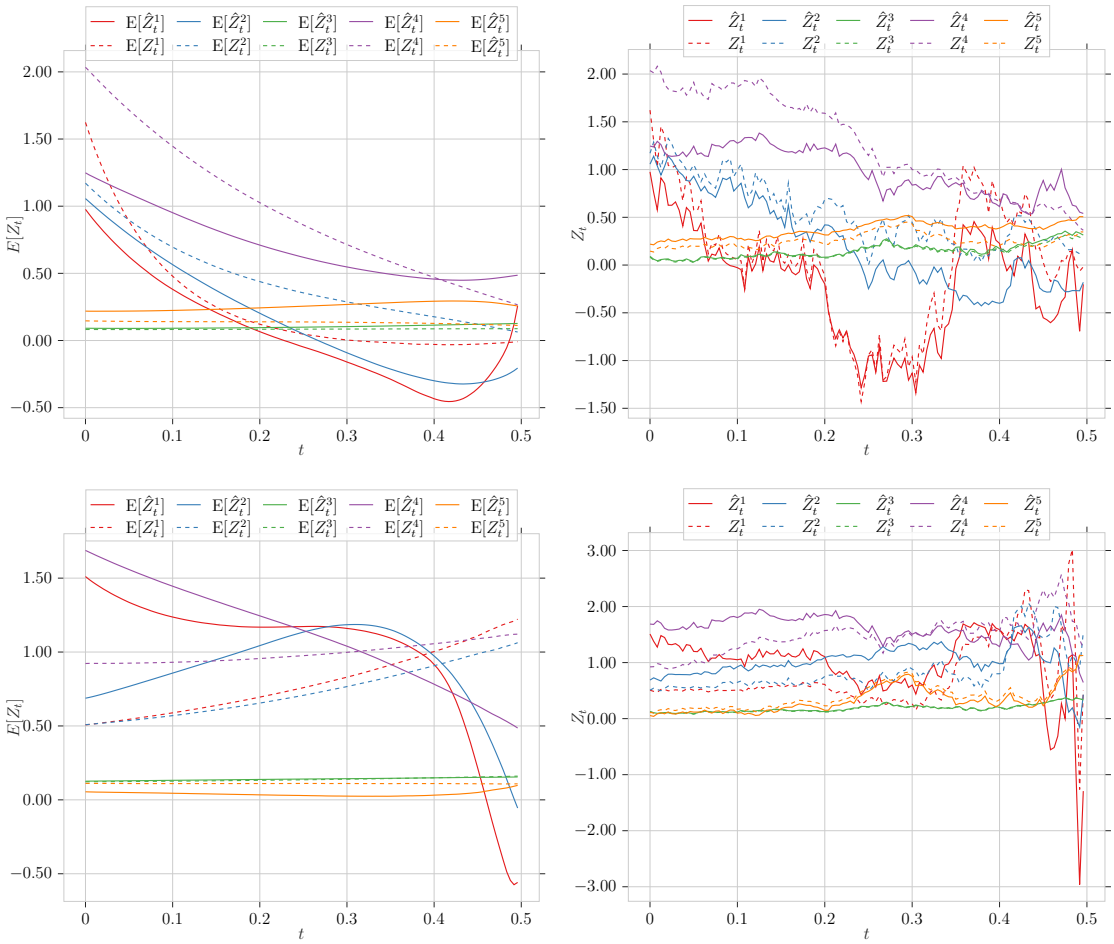

Figure 6.5: Results for the two-dimensional LQG game with five players, using the Deep fictitious play method. The top row shows the expectation and one representative trajectory of the first component of the control process Z^i and its approximation $\hat{Z}^i$ for each player, respectively. The bottom row instead shows the second component.

We present the numerical results for the two-dimensional LQG game with five players, for the Robust deep fictitious play method. Figure 6.6 shows that the method successfully approximates the state process. The same patterns appears in Figures 6.7 and 6.8.

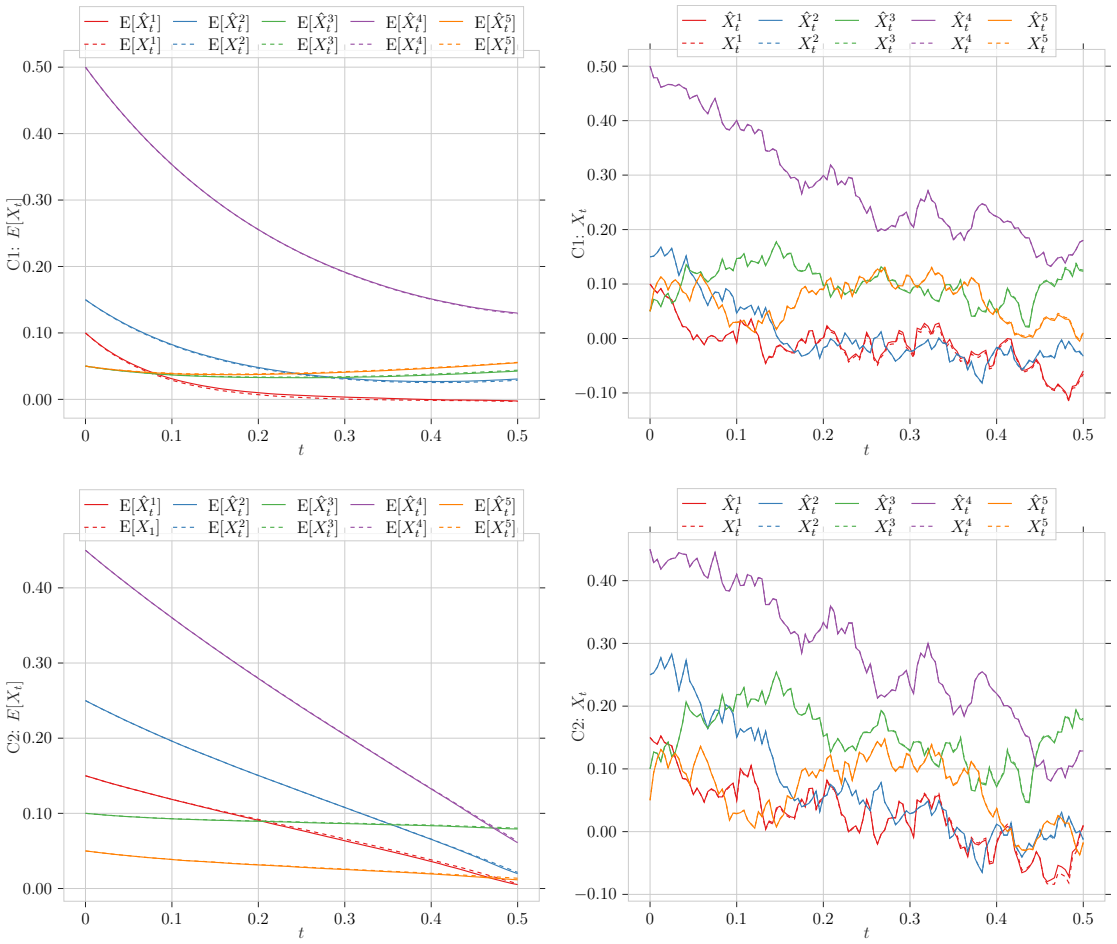

Figure 6.6: Numerical results for the Robust deep fictitious play method for the two-dimensional LQG game with five players. The expectation and one representative trajectory of the first component of the state process X^i and its approximation $\hat{X}^i$ for each player are shown in the first row. The second row illustrates the second component instead.

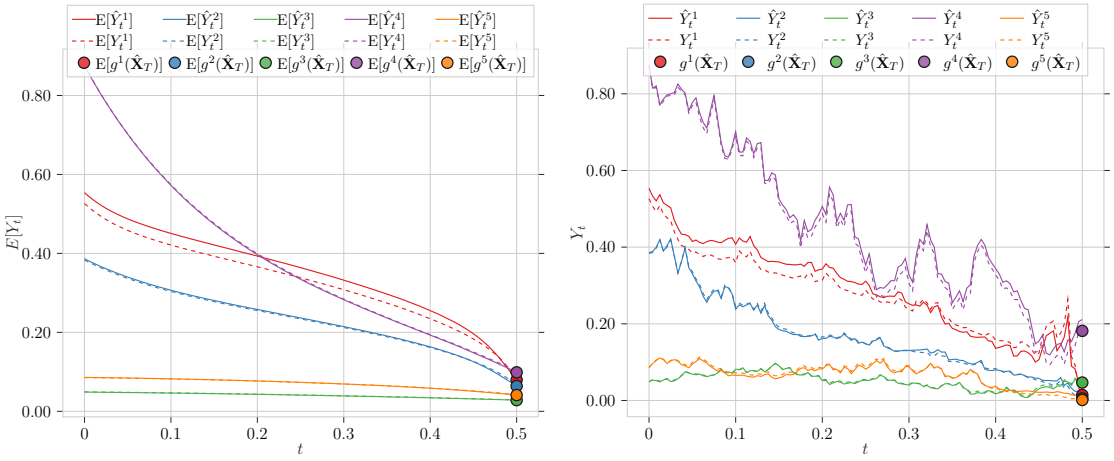

Figure 6.7: Numerical results for five-player LQG game. The left column illustrates the expectation of the backward process Y^i and the approximation $\hat{Y}^i$ for each player. The right column, instead, shows one representative trajectory.

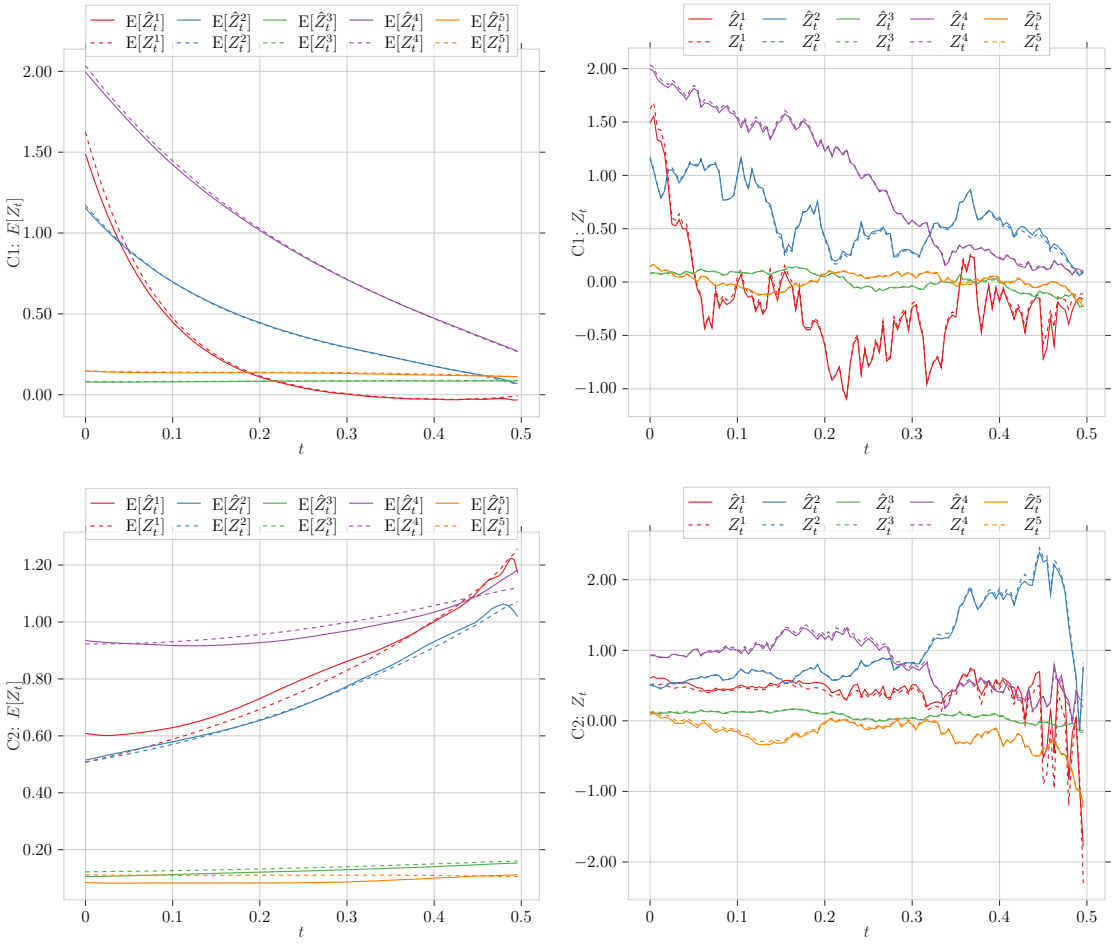

Figure 6.8: Numerical results for five-player LQG game using the Robust deep fictitious play method. The first row illustrates the expectation and one representative trajectory of the first component of the control process Z^i and its approximation $\hat{Z}^i$. The second row illustrates the second component.

6.2.2 High-dimensional example

In this section, we will consider an LQG game with a six-dimensional state and two-dimensional control. The game involves three players, and the parameters are given, for all players $i \in \mathcal{I}$, by

$$A^i = \operatorname{diag}(1,2,3,1,2,3), \quad C^i = (-0.2,-0.1,0,0,0.1,0.2)^\top,$$
$$\Sigma^i = \operatorname{diag}(0.05,0.25,0.05,0.25,0.05,0.25), \quad R_u^i = \operatorname{diag}(1,1).$$

Then, the remaining parameter for the dynamics of the players, which is individual for each player is

$$B^1 = \begin{bmatrix} 1 & -1 & 0 & 0.1 & 0 & 0.1 \\ 1 & 1 & 0.1 & 0 & 0.1 & 0 \\ 1 & -1 & 1 & 0 & 0.1 & 0 \\ 0.5 & 1 & 0.1 & 0 & 0.1 & 0 \\ 1 & -1 & 0 & 0.1 & 1 & 0.1 \\ 0 & -1 & 0.1 & -1 & 0 & 0.1 \end{bmatrix},$$
$$B^2 = \begin{bmatrix} 0 & 0.1 & 1 & -1 & 0 & 0.1 \\ 0.1 & 0 & 1 & 1 & 0.1 & 0 \\ 1 & 0 & 1 & -1 & 0.1 & 0 \\ 0.1 & 0 & 0.5 & 1 & 0.1 & 0 \\ 0 & 0.1 & 1 & -1 & 0 & 0.1 \\ 0.1 & 0 & 0 & -1 & 0.1 & 0 \end{bmatrix},$$
$$B^3 = \begin{bmatrix} 0 & 0.1 & 0 & 0.1 & 1 & -1 \\ 0.1 & 0 & 0.1 & 0 & 1 & 1 \\ 1 & 0 & 0.1 & 0 & 1 & -1 \\ 0.1 & 0 & 0.1 & 0 & 0.5 & 1 \\ 0 & 0.1 & 0 & 0.1 & 1 & -1 \\ 0.1 & 0 & 0.1 & 0 & 0 & -1 \end{bmatrix}.$$

The penalty matrices in the backward equation are defined as

$$R_x^1 = \operatorname{diag}(5,1,5,1,5,1,0,0,0,0,0,0,0,0,0,0,0,0),$$
$$R_x^2 = \operatorname{diag}(0,0,0,0,0,0,3,1,5,1,5,1,0,0,0,0,0,0),$$
$$R_x^3 = \operatorname{diag}(0,0,0,0,0,0,0,0,0,0,0,0,5,1,2,1,5,1),$$

and

$$G^1 = \operatorname{diag}(1,5,1,5,1,5,0,0,0,0,0,0,0,0,0,0,0,0),$$
$$G^2 = \operatorname{diag}(0,0,0,0,0,0,1,5,1,5,1,5,0,0,0,0,0,0),$$
$$G^3 = \operatorname{diag}(0,0,0,0,0,0,0,0,0,0,0,0,1,5,1,5,1,5).$$

6.2.2.1 Numerical results

As in the two-dimensional case, the Deep fictitious play algorithm fails to approximate the processes. This is evident in Figures 6.9, 6.10 and 6.11.

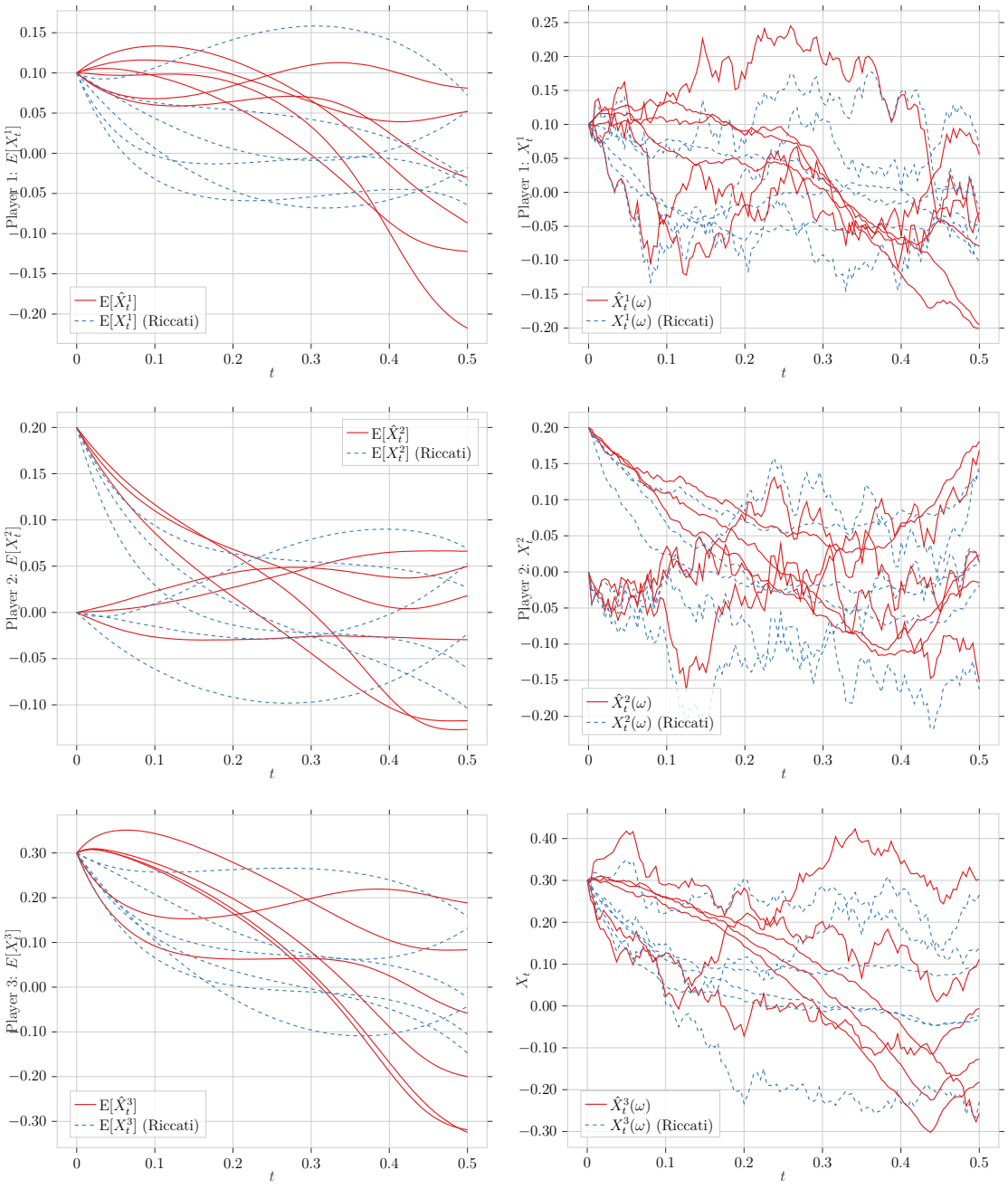

Figure 6.9: Numerical results, for the state process X^i and its approximation $\hat{X}^i$ for each player $i = 1, 2, 3$, using the Deep fictitious play method for $d = 6$, $\ell = 2$. Each row represents a player. The columns illustrate the expectations of all components and provide one representative trajectory for each component, respectively.

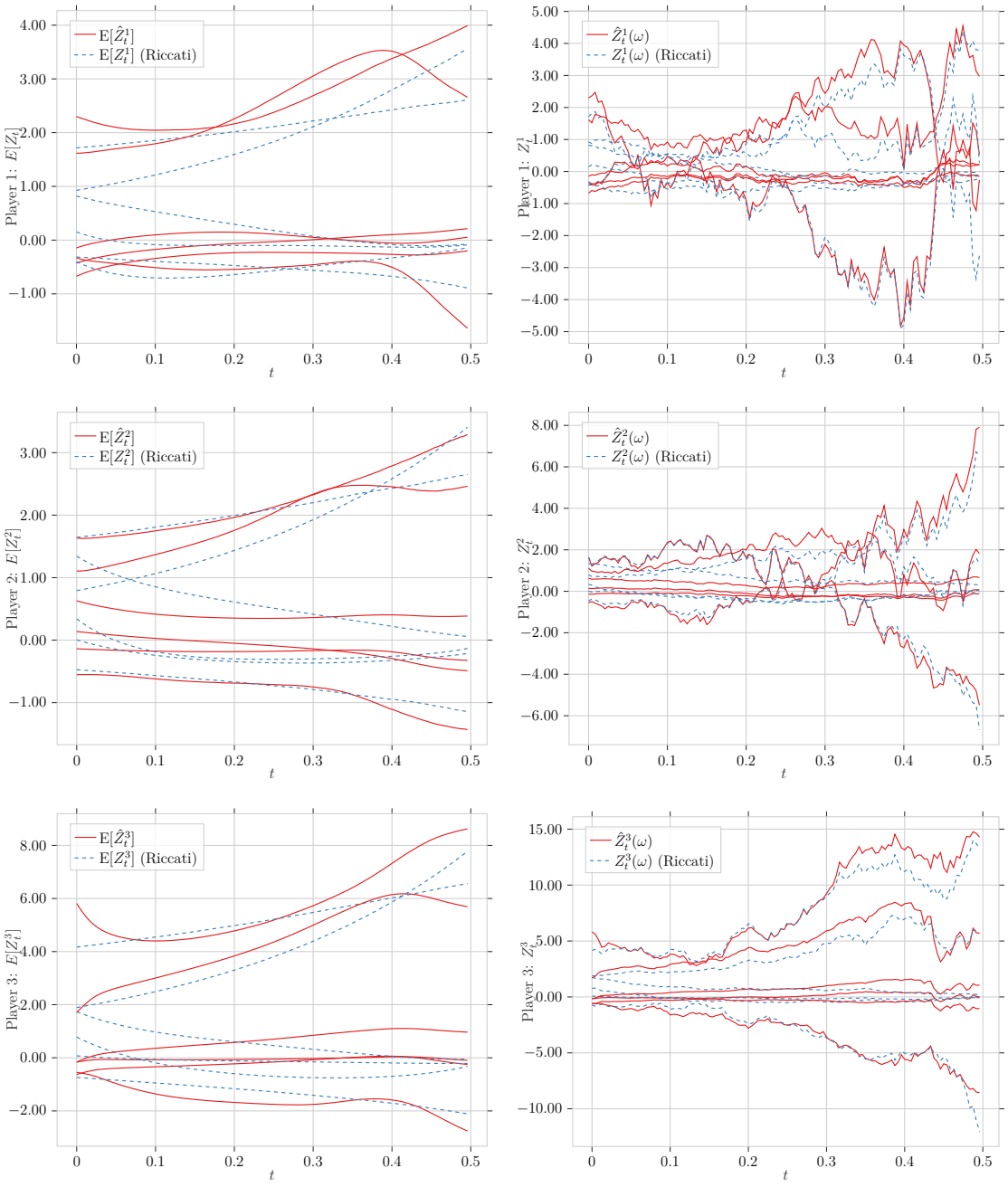

Figure 6.10: Numerical results for the high-dimensional LQG game, using the Deep fictitious play algorithm, with three players. The Z^i and its approximation $\hat{Z}^i$ for each player $i = 1, 2, 3$ is illustrated. Each row represents all components of a player i. The columns show the expectations for all components and illustrate one representative trajectory for each component.

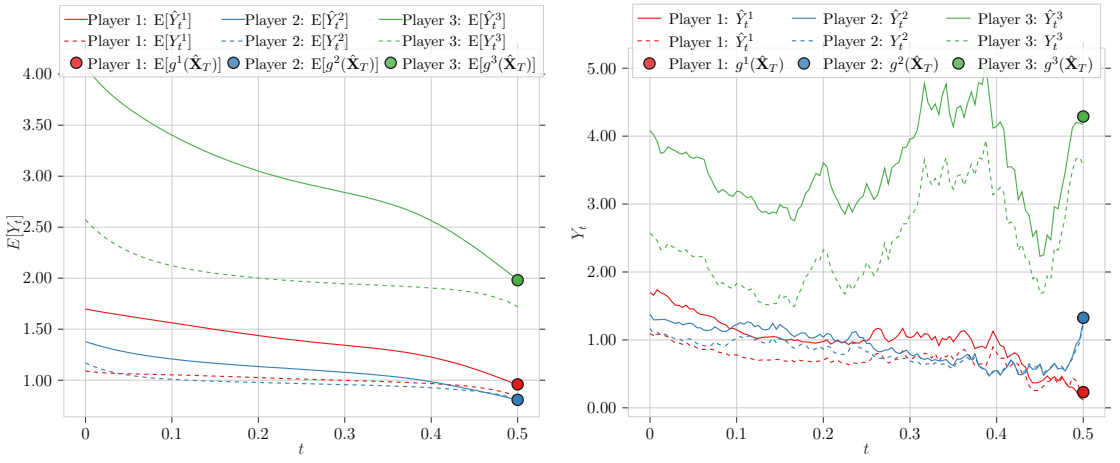

Figure 6.11: Results using the Deep fictitious play method for $d = 6$, $\ell = 2$ with three players. The left figure illustrates the backward process Y^i and its approximation $\hat{Y}^i$ as well as the approximated terminal condition $g^i(\hat{\mathbf{X}}_T)$ for $i = 1, 2, 3$. The right column illustrates one representative trajectory for each player.

Next, we show the results generated using the Robust Deep fictitious play algorithm. Figures 6.12, 6.13 and 6.14 indicate that the robust algorithm effectively approximates the different processes. For the Z process we some deviation from the analytical solution even for the robust algorithm. Since numerically, we will never find the true optimal solution, we instead find a solution which approximately minimizes the cost. These solutions are not necessarily unique and therefore different solutions might differ for some specific component or player in such a way as to not affecting the overall cost enough to not be approximately optimal.

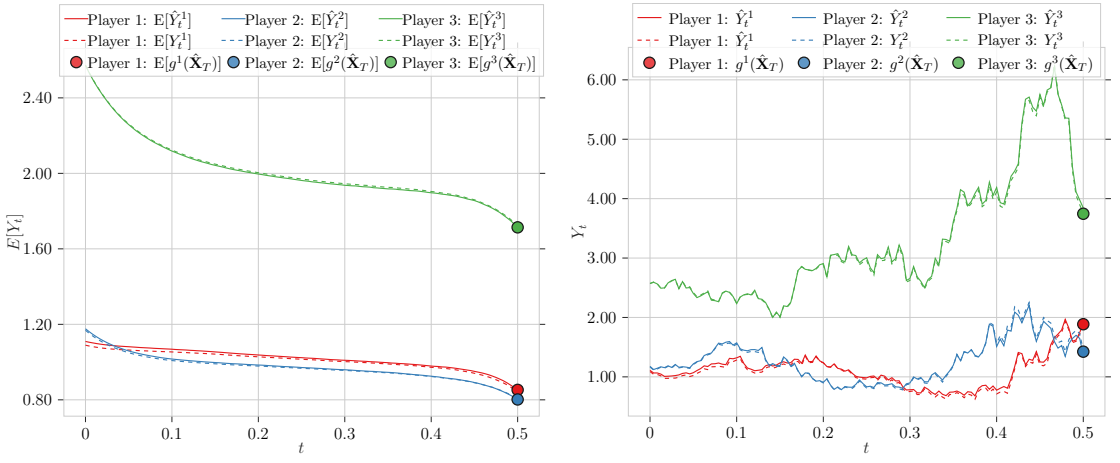

Figure 6.12: Numerical results, for the backward process Y^i and its approximation $\hat{Y}^i$ as well as the approximated terminal condition $g^i(\hat{\mathbf{X}}_T)$ for each player $i = 1, 2, 3$, using the Robust deep fictitious play method with $d = 6$, $\ell = 2$. The left figure shows the expectation, and the right is one representative trajectory.

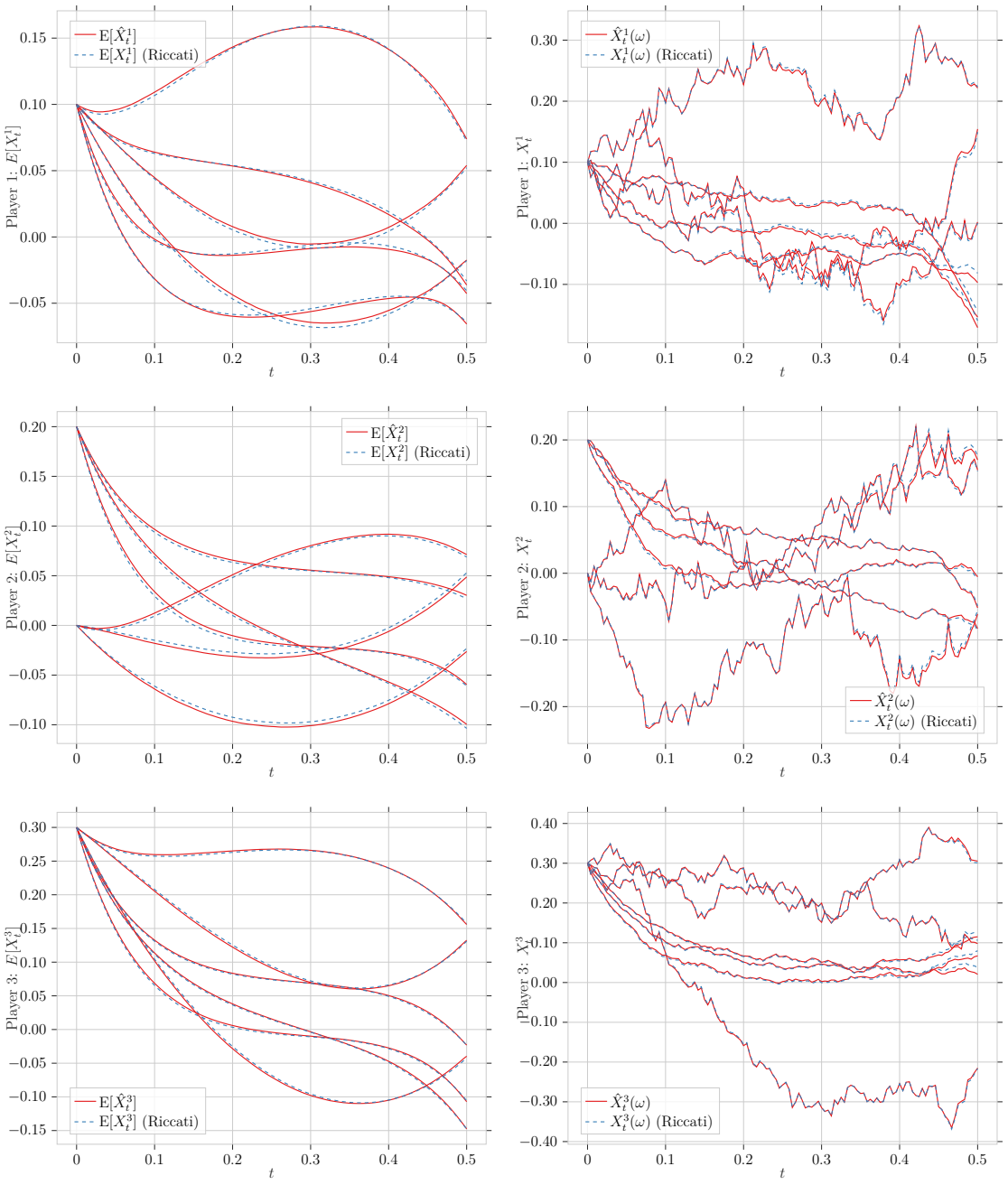

Figure 6.13: Numerical results for the Robust deep fictitious play method with $d = 6$, $\ell = 2$ for three players $i = 1, 2, 3$. Each row represents a different player, and the columns show the expectation of all components and one representative trajectory for all components, respectively.

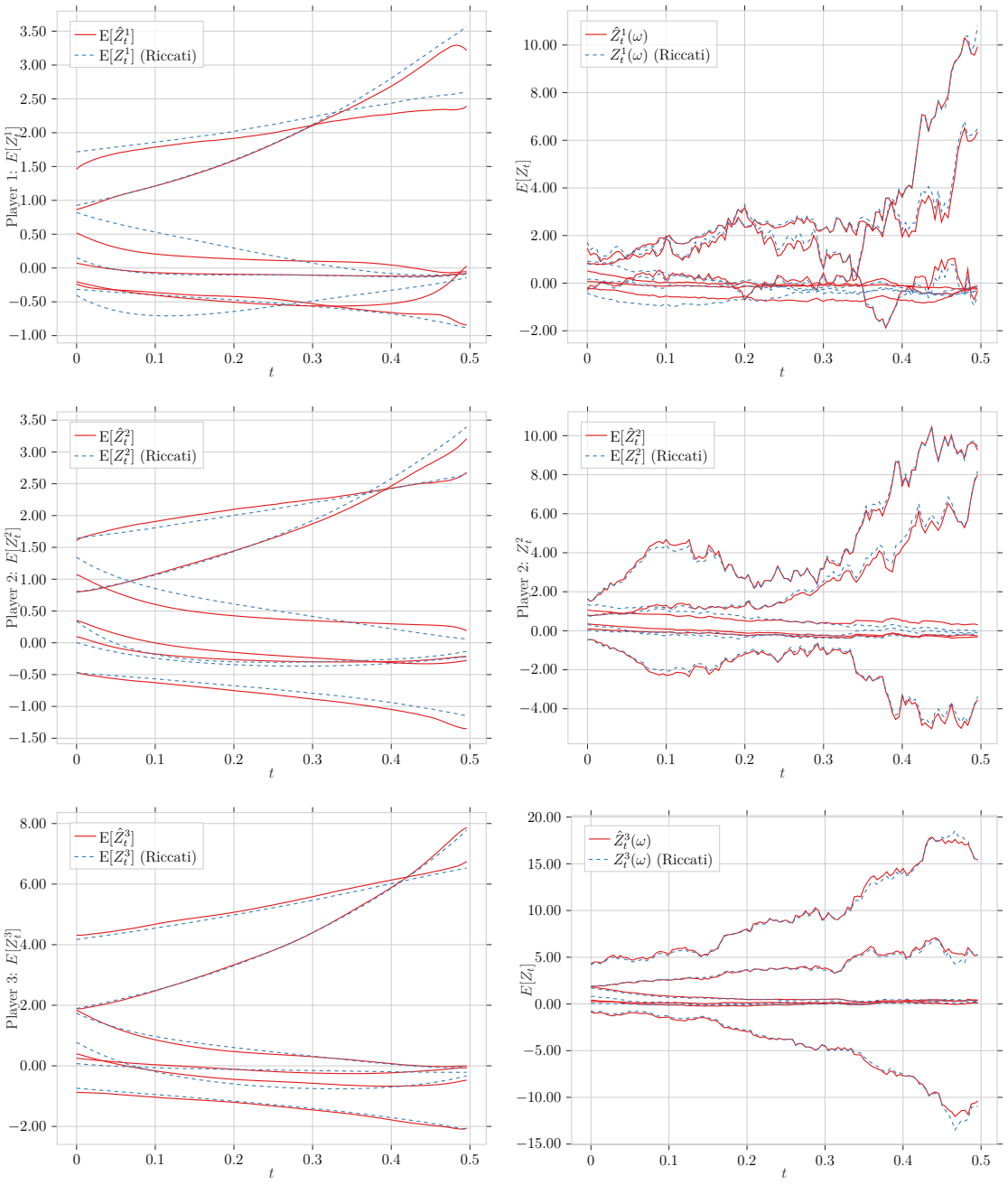

Figure 6.14: Numerical results, for the control process Z^i and its approximation $\hat{Z}^i$ for each player $i = 1, 2, 3$, using the Robust deep fictitious play method with $d = 6$, $\ell = 2$. Each row represents all components of a different player. The column shows the expectation and one representative trajectory, respectively.

6.3 Leader–Follower Drone Game

This chapter introduces a further numerical example designed to show the width of the possibilities of the novel solution algorithm, Robust deep fictitious play. The numerical example is a cooperative heterogeneous differential game wherein a swarm of drones is tasked to follow a path through a possibly complex environment. The cooperative nature of the game is due to the drones having complete knowledge of each other's strategies as well as positions, a noncooperative variant could involve two teams of drones competing to achieve some goal. Moreover, the differential game is heterogeneous since all drones are not of the same kind, there is a designated leader drone and the rest are followers. The leader has the known path given, while the followers only know the leader's position. This, along with the use of common and individual noise makes the problem incredibly complex.

The stochastic differential game for the leader is defined differently than that of the followers. Therefore, to underline the difference in problem formulations for the leader and the followers we let $\mathcal{X}^{\mathfrak{u}} = (\mathcal{X}_t^{\mathfrak{u}})_{t\in[0,T]}$ be the state process of the leader while $\mathbf{X}_t^{\mathbf{u}} = (X_t^{1,\mathbf{u}}, \dots, X_t^{N-1,\mathbf{u}})$ denotes the state process for all of the follower drones where both are defined on the complete filtered probability space $(\Omega, \mathcal{F}, (\mathcal{F}_t)_{t\in[0,T]}, \mathbb{P})$ where the filtration $(\mathcal{F}_t)_{t\in[0,T]}$ is generated by the common noise as well as the individual noise k-dimensional Brownian motions $W^0 = (W_t^0)_{t\in[0,T]}$ and $W^i = (W_t^i)_{t\in[0,T]}$ respectively, i.e $(\mathcal{F}_t)_{t\in[0,T]} = \sigma(\{W_s^0,\ W_s^i,\ 0 \le s \le t,\ 0 \le i \le N\})$. Then, both state processes include the two-dimensional position and velocity of the drone. Further, we differentiate between the control processes for the different types of drones, thus we let $\mathfrak{u} = (\mathfrak{u}_t)_{t\in[0,T]} \in \mathcal{U}$ be the admissible control process of the leader drone which takes values in the set $\mathbb{U} \subseteq \mathbb{R}^2$. We let the collection $\mathbf{u} = ((u_t^1)_{t\in[0,T]}, \dots, (u_t^{N-1})_{t\in[0,T]}) \in \mathcal{U}$ be the admissible control processes of the follower drones that takes values in the product space $\mathbb{U} \coloneqq \mathbb{U}^1 \times \cdots \times \mathbb{U}^{N-1}$. Finally, we differentiate between the types of drones in terms of the cost functional as well. That is, the leader drones' cost functional is the mapping $\mathcal{J} : \mathbb{U} \times [0,T] \times \mathbb{R}^4 \to \mathbb{R}$ and the cost functional for the follower drones are the mapping $J : \mathbb{U} \times \mathcal{I} \times [0,T] \times \mathbb{R}^{4N} \to \mathbb{R}$. We now present the problem formulation for the leader drone

$$
\begin{aligned}
\mathrm{d}\mathcal{X}_t^{\mathfrak{u}} &= (A\mathcal{X}_t^{\mathfrak{u}} + B\mathfrak{u}_t)\,\mathrm{d}t + \sigma(\rho\,\mathrm{d}W_t^0 + \sqrt{1-\rho^2}\,\mathrm{d}W_t^i), \quad \mathcal{X}_0^{\mathfrak{u}} = \hat{x}_0, \\
\mathcal{J}^{\mathfrak{u}}(t,x) &= \mathbb{E}\left[\int_t^T \mathbb{w}^1 \|\mathcal{X}_s^{\mathfrak{u}} - \hat{x}_s\|_{L^2}^2 + \mathbb{w}^2 \|\mathfrak{u}_s\|_{L^2}^2\,\mathrm{d}s + \|\mathcal{X}_T^{\mathfrak{u}} - \hat{x}_T\|_{L^2}^2 \middle| \mathcal{X}_t^{\mathfrak{u}} = x\right],
\end{aligned}
$$

where the matrices A and B are defined as

$$
A \coloneqq \begin{pmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}, \qquad B \coloneqq \begin{pmatrix} 0 & 0 \\ 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{pmatrix}.
$$

Here, the matrix A is formulated in such a way as to extract the velocity of the state and B in such a way as to only control the velocity of the drone. Moreover, the initial condition is set such that the leader is initialized at the start of the

given path $\hat{x} : [0,T] \to \mathbb{R}^4$, i.e. $\mathcal{X}_0^{\mathfrak{u}} = \hat{x}_0$ $\mathbb{P}$-a.s. which is explicitly defined as $\hat{x}_t = (v_0 t,\ C\sin(wt),\ v_0, C\omega\cos(\omega t))^\top$ with $v_0, C, \omega \in \mathbb{R}$. The running cost is defined as the norm between the state process and the given path at a time $s \in [t,T]$. In addition, there is a cost of applying control, which is accumulated over all times in the interval $[t,T]$. The running cost is the norm between the state process and the given path at the terminal time, thus the drone should follow the path at all times, as well as end up at the final position of the given path. Furthermore, $\mathbb{w}^1$, $\mathbb{w}^2$are two constants used to weight the cost of deviating from the given path and applying the control $\mathfrak{u}_t$. Then, the stochastic control problem for follower i is given by

$$\begin{aligned}
\mathrm{d}X_t^{i,\mathbf{u}} &= (A\mathbf{X}_t^i + B\mathbf{u}_t^i)\,\mathrm{d}t + \sigma^i(\rho\,\mathrm{d}W_t^0 + \sqrt{1-\rho^2}\,\mathrm{d}W_t^i), \quad X_0^{i,\mathbf{u}} = x_0^i,\\
J^{i,\mathbf{u}}(t,\mathbf{x}) &= \mathbb{E}\left[\int_t^T \left(\|\mathfrak{P}_s - P_s^i\|_{L^2} - d\right)^2 + w^1 \sum_{j\in\mathcal{I},\, j\neq i} \max\left(0, d - \|P_s^i - P_s^j\|_{L^2}\right)^2 \right.\\
&\qquad \left. + w^2\|\mathbf{u}_t^i\|_{L^2}^2\,\mathrm{d}s + \left(\|\mathfrak{P}_T - P_T^i\|_{L^2} - d\right)^2 \,\middle|\, P_t^i = (\mathbf{x}_i)_{i\in\{1,2\}}\right],
\end{aligned}$$

where A, B are defined as for the leader drone and $\mathbf{X}_0^{i,\mathbf{u}} = \mathbf{x}_0^i$ $\mathbb{P}$-a.s. is a given initial condition for each follower drone. Moreover, $\mathfrak{P} \coloneqq (\mathcal{X}_i^{\mathfrak{u}})_{i\in\{1,2\}}$ and $P^i \coloneqq (\mathbf{X}_j^{i,\mathbf{u}})_{j\in\{1,2\}}$ denote the position of the leader drone and follower drone i respectively. Therefore, the running cost is the norm of the difference between the position of the follower drone i and the leader drone minus a set distance $d \in \mathbb{R}$ length units, plus the sum of the max of zero or the norm of the positional difference between each pair of follower drone i and $j \neq i$ minus the set distance d. Thus the optimal position of the follower drone i is at a distance d from the leader as well as from all other follower drones. We apply a cost of applying control at each timestep. Finally, the terminal cost is the normed distance between the leader and follower i minus the set distance d. Thus, the terminal condition is for the follower drone i to end at a distance d from the leader. Moreover, the diffusion term, for player i as well as for the leader, can be expressed as the matrix

$$\Sigma^i \coloneqq \begin{bmatrix}
\sigma\rho & 0 & 0 & 0 & \sigma\sqrt{1-\rho^2} & 0 & 0 & 0\\
0 & \sigma\rho & 0 & 0 & 0 & \sigma\sqrt{1-\rho^2} & 0 & 0\\
0 & 0 & \sigma\rho & 0 & 0 & 0 & \sigma\sqrt{1-\rho^2} & 0\\
0 & 0 & 0 & \sigma\rho & 0 & 0 & 0 & \sigma\sqrt{1-\rho^2}
\end{bmatrix},$$

multiplied with a vector of Brownian motion increments $\mathrm{d}\widetilde{W_t^i} \coloneqq (\mathrm{d}W_t^0,\ \mathrm{d}W_t^i)^\top$, where $\sigma \in \mathbb{R}$ and $\rho \in (-1,1)$.

We are now ready to present the HJB equations for each drone type. We omit the derivation and instead refer to Section 3.1.2 and Section 3.2.2, respectively. We state the HJB equation for the leader drone

$$\partial_t \mathcal{V} + \mathbb{w}^1\|x - \hat{x}_s\|_{L^2}^2 + \inf_{\mathfrak{u}\in\mathcal{U}}\left((Ax + B\mathfrak{u}_t)^\top \nabla_{\mathbf{x}}\mathcal{V} + \mathbb{w}^2\|\mathfrak{u}_t\|_{L^2}^2\right) + \frac{1}{2}\mathrm{Tr}(\sigma\sigma^\top\Delta_x\mathcal{V}) = 0,$$

$$\mathcal{V}(T,x) = \|x - \hat{x}_T\|^2,$$

where $\mathcal{V}$ denotes the value function for the leader drone, and $(t,x) \in [0,T] \times \mathbb{R}^4$ is omitted unless otherwise specified. For the follower i the HJB equation is instead given by

$$\begin{aligned}
&\partial V^i + w^1\|\mathbf{x} - \hat{\mathbf{x}}_s\|^2 + \inf_{\mathbf{u}^i \in \mathcal{U}} \Big(\left(A\mathbf{x} + B\mathbf{u}_t^i\right)^\top \nabla_x V^i + \left(\|\mathfrak{P}_s - P_s^i\|_{L^2} - d\right)^2 \\
&\qquad + w^1 \sum_{j \in \mathcal{I}, j \neq i} \max\left(0, \|P_s^i - P_s^j\|_{L^2}\right)^2 + w^2 \|\mathbf{u}_t^i\|_{L^2}^2 \Big) + \frac{1}{2}\mathrm{Tr}\left(\sigma^i \sigma^{i,\top} \Delta_x V^i\right) = 0, \\
&V^i(T,\mathbf{x}) = \left(\|\mathfrak{P}_T - P_T^i\|_{L^2} - d\right)^2.
\end{aligned}$$

Further, the optimal strategies for both of these systems are given by minimizing the Hamiltonian with respect to the control process and solving the minimization problem. This yields

$$\begin{aligned}
\mathfrak{u}^* &= -\frac{1}{2\mathbb{w}^2} B^\top \nabla_x \mathcal{V}, \\
\mathbf{u}^{i,*} &= -\frac{1}{2w^2} B^\top \nabla_x V^i.
\end{aligned}$$

The FBSDE and system of FBSDEs for the leader and followers respectively is thus given by

$$\begin{aligned}
\mathrm{d}\mathcal{X}_t^{\mathfrak{u}} &= \left(A\mathcal{X}_t^{\mathfrak{u}} + B\mathfrak{u}_t\right)\mathrm{d}t + \Sigma\,\mathrm{d}\widetilde{W}_t^i, \quad \mathcal{X}_0^{\mathfrak{u}} = \hat{x}_0, \\
\mathcal{Y}_t &= \|\mathcal{X}_T^{\mathfrak{u}} - \hat{x}\|_{L^2}^2 + \int_t^T \mathbb{w}^1 \|\mathcal{X}_s^{\mathfrak{u}} - \hat{x}_s\|_{L^2}^2 + \mathbb{w}^2 \|\mathfrak{u}_s\|_{L^2}^2 \,\mathrm{d}s - \int_t^T (\mathcal{Z}_s)^\top \,\mathrm{d}W_s,
\end{aligned}$$

and for the leader and for the followers by

$$\begin{aligned}
\mathrm{d}X_t^{i,\mathbf{u}} &= \left(AX_t^i + Bu_t^i\right)\mathrm{d}t + \Sigma^i\,\mathrm{d}\widetilde{W}_t^i, \quad \mathbf{X}_0^{i,\mathbf{u}} = \mathbf{x}_0^i, \\
Y_t &= \left(\|\mathfrak{P}_T - P_T^i\|_{L^2} - d\right)^2 + \int_t^T \left(\|\mathfrak{P}_s - P_s^i\|_{L^2} - d\right)^2 \\
&\qquad + w^1 \sum_{j \in \mathcal{I}, j \neq i} \max\left(0, d - \|P_s^i - P_s^j\|_{L^2}\right)^2 + w^2 \|u_t^i\|_{L^2}^2 \,\mathrm{d}s + \int_t^T (Z_s)^\top \,\mathrm{d}W_s.
\end{aligned}$$

Both systems are then discretized in order to apply the Deep fictitious play algorithm and the Robust deep fictitious play algorithm to compute numerical results.

6.3.1 Numerical results

In this section we present the numerical results generated by the two algorithms, namely the Deep fictitious play algorithm and the novel Robust deep fictitious play algorithm. Since the numerical example is designed to have an intuitive solution we present a representative trajectory for the two algorithms. Results for the expectation of the processes are instead presented in Appendix B. Thus, Figure 6.15 and Figure 6.16 showcase a representative trajectory for the drones. It is evident from Figure 6.15 that the drones do not follow the path at all and instead simply move towards the final position, disregarding the desired way there. For the novel algorithm however, the drones follow the given path closely as well as keeping a safe distance in between each other. Thus, the novel algorithm is a clear improvement on the previous algorithms in the literature.

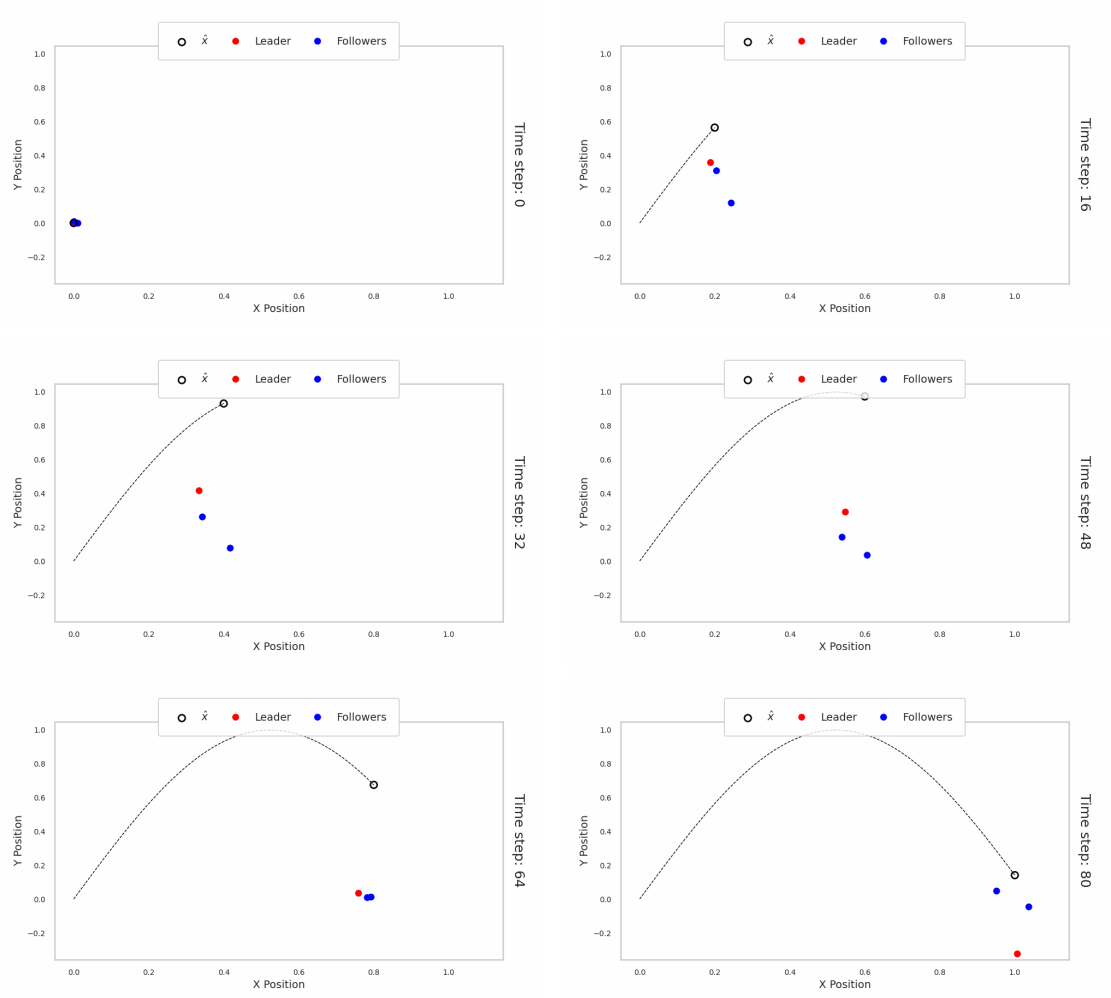

Figure 6.15: Representative trajectory for the drones generated with the Deep fictitious play algorithm. The position for drones is plotted for the different times $t_1 = 0\text{s}$, $t_2 = 0.1\text{s}$, $t_3 = 0.2\text{s}$, $t_4 = 0.3\text{s}$, $t_5 = 0.4\text{s}$, $t_6 = 0.5\text{s}$. The leader drone is colored red, the followers are colored blue and the given path is the black dotted line.

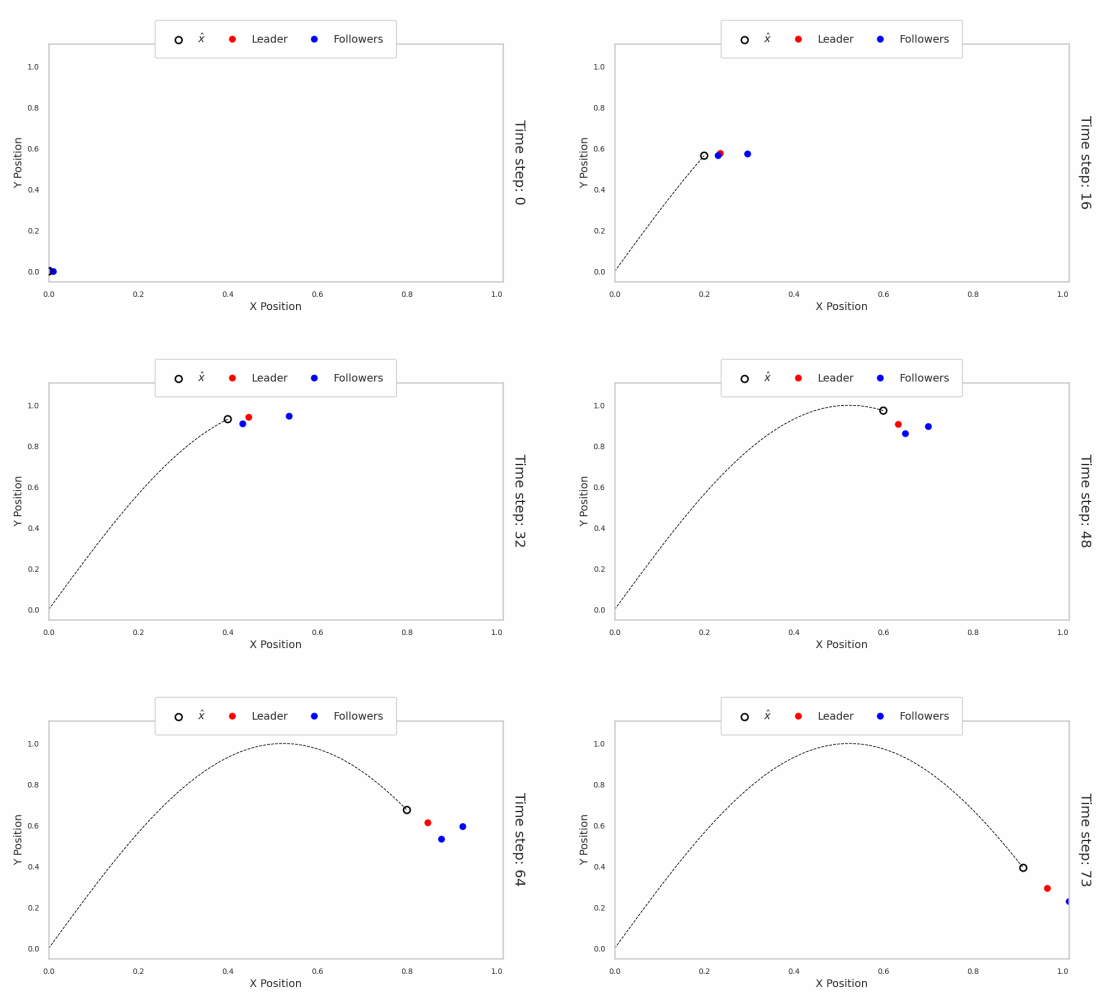

Figure 6.16: Representative trajectory for the drones generated with the Robust deep fictitious play algorithm. The position for drones is plotted for the different times $t_1 = 0$s, $t_2 = 0.1$s, $t_3 = 0.2$s, $t_4 = 0.3$s, $t_5 = 0.4$s, $t_6 = 0.456$s. The leader drone is colored red, the followers are colored blue and the given path is the black dotted line.

7
Conclusion

This thesis considers the stochastic optimal control of N-player systems such as swarms of drones. To consider such problems, measure theory and stochastic calculus are presented in order to understand the problem of stochastic optimal control. Then, we present the problem formulation of a stochastic control problem. The stochastic control problem is then reformulated in two steps, firstly, to the HJB semi-linear parabolic PDE and in turn to the FBSDE. The existence of a unique solution is proved for all formulations of the problem. Then, we generalize the system to N players and repeat the structure of presenting the problem formulation, the N coupled HJB equations, and finally the systems of FBSDEs. Further, the solution algorithm Deep FBSDE is discussed as well as its evolution, the Deep C-FBSDE algorithm which generalizes the method to handle problems stemming from stochastic control problems. A third algorithm from the literature is discussed, the Deep fictitious play algorithm which is a generalization of the Deep FBSDE method which handles N-player differential games. A clear research gap emerges for a robust N-player solution algorithm that handles stochastic control problems with N agents. Therefore, a novel algorithm is developed to solve such problems, which builds on the robust solution algorithm Deep C-FBSDE. To showcase the increased performance of the novel algorithm a numerical example is designed which is too complex for previous algorithms. The results show that the novel algorithm is a successful improvement on existing algorithms. The numerical example is a heterogeneous cooperative stochastic differential game that describes a swarm of drones navigating a complex environment. A further property of the numerical example that increases the complexity is the use of common noise among the players. The complexity of the numerical example showcases the capabilities of the novel robust algorithm.

Future research should further increase the possibilities of the robust algorithms. This could include many things, e.g., including so-called Lévy processes which would allow for impulse-modeling. This would translate to the drones in the numerical example colliding, thus more realistic game scenarios can be modeled. Further, the possibilities in terms of non-cooperative games increase tremendously since colliding in to an enemy could be, in certain scenarios, a viable strategy to further the goals of your team. In technical terms a Lévy process includes a Poisson process in addition to the traditional Brownian motion of an SDE. The Poisson process introduces random discontinuous jumps. These jumps is what could model the impulse of the drones. Other applications include finance where jump processes are commonplace for modeling financial derivatives and portfolios with underlying financial assets.

Other efforts could include using control policies that do not fulfill the Markov condition. Thus, the system would at all times be dependent on its entire history. Further research is not limited to solely expanding the horizon of the novel robust solution algorithm. Efforts could investigate the a priori as well as a posteriori error bounds of the algorithm as well as its convergence. Another topic to explore is the use of the so-called BMO spaces to seek the control process $Z = (Z_t)_{t\in[0,T]}$ in. This space is used in order to limit the oscillating nature of the control process which would increase the robustness of the solution algorithm.

Bibliography

[1] H. Zhao, S. Wu, Y. Wen, W. Liu, and X. Wu, "Modeling and flight experiments for swarms of high dynamic uavs: A stochastic configuration control system with multiplicative noises," *Sensors*, vol. 19, no. 15, 2019.

[2] M. Wang, J. G. Scott, and A. Vladimirsky, "Stochastic optimal control to guide adaptive cancer therapy," *bioRxiv*, 2022.

[3] C. A. Ball and A. Roma, "Stochastic volatility option pricing," *Journal of Financial and Quantitative Analysis*, vol. 29, no. 4, p. 589607, 1994.

[4] J. Han, A. Jentzen, and W. E, "Solving high-dimensional partial differential equations using deep learning," *Proceedings of the National Academy of Sciences*, vol. 115, no. 34, pp. 8505–8510, 2018.

[5] J. Han, R. Hu, and J. Long, "Convergence of deep fictitious play for stochastic differential games," *Frontiers of Mathematical Finance*, vol. 1, no. 2, pp. 287–319, 2022.

[6] K. Andersson, A. Andersson, and C. W. Oosterlee, "Convergence of a robust deep fbsde method for stochastic control," *SIAM Journal on Scientific Computing*, vol. 45, no. 1, pp. A226–A255, 2023.

[7] T. Tao, *Introduction to measure theory*. American Mathematical Soceity, 2011.

[8] S. Wagon, *The Banach-Tarski Paradox*. Encyclopedia of Mathematics and its Applications, Cambridge University Press, 1985.

[9] C. Heil, *Introduction to Real Analysis*. Springer Cham, 2019.

[10] F. C. Klebaner, *Introduction to stochastic calculus with applications*. Imperial college press, third ed., 2012.

[11] P. Morters and Y. Peres, *Brownian motion*, vol. 30 of *Cambridge Series in Statistical and Probabilistic Mathematics*. UK United Kingdom: Cambridge University Press, 2010.

[12] A. Friedman, "5 - stochastic differential equations," in *Stochastic Differential Equations and Applications* (A. Friedman, ed.), Probability and Mathematical Statistics: A Series of Monographs and Textbooks, pp. 98–127, Academic Press, 1975.

[13] P. E. Kloeden and E. Platen, *Numerical Solution of Stochastic Differential Equations*. Springer Berlin, Heidelberg, 1992.

[14] S. Park, C. Yun, J. Lee, and J. Shin, "Minimum width for universal approximation," *CoRR*, vol. abs/2006.08859, 2020.

[15] W. H. Fleming and H. Soner, *Controlled Markov Processes and Viscosity Solutions*. Springer New York, NY, 2005.

[16] A. S. Poznyak, "Chapter 22 - variational calculus and optimal control," in *Advanced Mathematical Tools for Automatic Control Engineers: Deterministic Techniques* (A. S. Poznyak, ed.), pp. 647–711, Oxford: Elsevier, 2008.

[17] N. Krylov, *Nonlinear elliptic & parabolic equations of the second order*. Mathematics & its applications 7, Dordrecht: D.Reidel, 1987.

[18] B. S. D. Equations, *Jianfeng Zhang*. Springer New York, NY, 2017.

[19] K. Andersson, A. Andersson, and C. W. Oosterlee, "The deep multi-fbsde method: a robust deep learning method for coupled fbsdes," *arXiv preprint arXiv:2503.13193*, 2025.

[20] R. Carmona and F. Delarue, *Probabilistic Theory of Mean Field Games with Applications I Mean Field FBSDEs, Control, and Games*, pp. 1–695. Probability Theory and Stochastic Modelling, United States: Springer Nature, 2018.

[21] J. Han and J. Long, "Convergence of the deep bsde method for coupled fbsdes," *Probability, Uncertainty and Quantitative Risk*, vol. 5, 2020.

[22] D. Becherer and S. Hesse, "Common noise by random measures: Mean-field equilibria for competitive investment and hedging," 2024.

[23] R. Carmona, J. Fouque, and L. Sun, "Mean field games and systemic risk," *Communications in Mathematical Sciences*, vol. 13, no. 4, pp. 911–933, 2015. Publisher Copyright: © 2015 International Press.

A

Derivation of coupled system of N HJB equations

The so-called value function is central in understanding the derivation of the HJB equation. It follows naturally that a value function for N-players is central in the derivation for the N-coupled HJB equations. Therefore we define

$$V^i(t, \mathbf{x}) \coloneqq \inf_{u_t^i \in \mathcal{U}^i} J^i(t, \mathbf{x}; \mathbf{u}_t),$$

where $i \in \mathcal{I}$ and J^i denotes the cost functional for player i. Furthermore, both $\mathbf{V} \coloneqq (V^1, \dots, V^N)^\top$ and $\mathbf{J} \coloneqq (J^1, \dots, J^N)^\top$ are vector concatenations of the individual contributions. We are now ready to state Bellman's dynamic programming principle for a multi-agent system.

Theorem 15 (N-player Bellman's dynamic programming principle)**.** *Let $t \in [0, T]$ and $\tau \in (t, T]$, then for $i \in \mathcal{I}$ it holds*

$$V^i(t, \mathbf{x}) = \inf_{u_t^i \in \mathcal{U}^i} \mathbb{E}\left[\int_t^\tau f^i(s, \mathbf{X}_s^{\mathbf{u}}, u_s^i, \mathbf{u}_s^{-i}) \, \mathrm{d}s + V^i(\tau, \mathbf{X}_\tau^{\mathbf{u}}) \middle| \mathbf{X}_t^{\mathbf{u}} = \mathbf{x} \right], \tag{A.1}$$

for $\mathbf{x} \in \mathbb{R}^{dN}$. Here we have introduced the notation

$$\mathbf{u}_t^{-i} \coloneqq (u_t^1, \dots, u_t^{i-1}, u_t^{i+1}, \dots, u_t^N),$$

for which it follows that $\mathbf{u}_t = (u_t^i, \mathbf{u}_t^{-i})$.

Proof. The claim follows from the proof of Bellman's dynamic programming principle for a single agent. □

Then, to derive the HJB equation, we begin with Bellman's dynamic programming principle (A.1). Let $\mathbf{v}_t \in \mathcal{A}$ be an arbitrary admissible control and $\tau = t + h$ for some small $h > 0$. This gives

$$V^i(t, \mathbf{x}) \leq \mathbb{E}\left[\int_t^{t+h} f^i(s, \mathbf{X}_s^{\mathbf{v}}, v_s^i, \mathbf{v}_s^{-i}) \, \mathrm{d}s + V^i(t + h, \mathbf{X}_{t+h}^{\mathbf{v}}) \middle| \mathbf{X}_t^{\mathbf{v}} = \mathbf{x} \right]. \tag{A.2}$$

Next, subtract $V^i(t,\mathbf{x})$ from both sides of (3.4) and divide by h, giving

$$\begin{aligned}
0 &\leq \frac{1}{h}\mathbb{E}\left[\int_t^{t+h} f^i(s,\mathbf{X}_s^{\mathbf{v}},v_s^i,\mathbf{v}_s^{-i})\,\mathrm{d}s + V^i(t+h,\mathbf{X}_{t+h}^{\mathbf{v}}) - V^i(t,\mathbf{x})\Big|\mathbf{X}_t^{\mathbf{v}}=\mathbf{x}\right]\\
&= \frac{1}{h}\mathbb{E}\left[\int_t^{t+h} f^i(s,\mathbf{X}_s^{\mathbf{v}},v_s,\mathbf{v}_s^{-i})\,\mathrm{d}s\Big|\mathbf{X}_t^{\mathbf{v}}=\mathbf{x}\right]\\
&\qquad + \frac{1}{h}\mathbb{E}\left[V^i(t+h,\mathbf{X}_{t+h}^{\mathbf{v}}) - V^i(t,\mathbf{x})\big|\mathbf{X}_t^{\mathbf{v}}=\mathbf{x}\right]\\
&= I_1 + I_2, \qquad \text{(A.3)}
\end{aligned}$$

where the linearity of the expectation operator is used in the first equality. We now study the limits of I_1 and I_2 as $h\downarrow 0$. For I_1, assume that the map

$$s\mapsto \mathbb{E}\left[f^i(s,\mathbf{X}_s^{\mathbf{v}},v_s^i,\mathbf{v}_s^{-i})|\mathbf{X}_t^{\mathbf{v}}=\mathbf{x}\right]$$

is locally integrable in s. Fubini's theorem then allows to interchange expectation and integration, and togheter with the Lebesgue differentiation theorem, it follows that

$$\lim_{h\downarrow 0} I_1 = \lim_{h\downarrow 0}\frac{1}{h}\int_t^{t+h}\mathbb{E}\left[f^i(s,\mathbf{X}_s^{\mathbf{v}},v_s^i,\mathbf{v}_s^{-i})\Big|\mathbf{X}_t^{\mathbf{v}}=\mathbf{x}\right]\,\mathrm{d}s = f^i(t,\mathbf{x},v_t^i,\mathbf{v}_t^{-i}). \quad \text{(A.4)}$$

For the second term I_2, applying Itô's formula (see Theorem 5), gives

$$\begin{aligned}
&\lim_{h\downarrow 0} I_2 \qquad \text{(A.5)}\\
&= \lim_{h\downarrow 0}\frac{1}{h}\mathbb{E}\left[V^i(t+h,\mathbf{X}_{t+h}^{\mathbf{v}}) - V^i(t,\mathbf{X}_t^{\mathbf{v}})\,\mathrm{d}s\Big|\mathbf{X}_t^{\mathbf{v}}=\mathbf{x}\right]\\
&= \lim_{h\downarrow 0}\frac{1}{h}\mathbb{E}\left[\int_t^{t+h}(\partial_s + L^{\mathbf{v}})\,V^i(s,\mathbf{X}_s^{\mathbf{v}})\,\mathrm{d}s + \int_t^{t+h}\nabla_x V^i(s,\mathbf{X}_s^{\mathbf{v}})\Sigma(s,\mathbf{X}_s^{\mathbf{v}})\,\mathrm{d}W_s\Big|\mathbf{X}_t^{\mathbf{v}}=\mathbf{x}\right]\\
&= \lim_{h\downarrow 0}\frac{1}{h}\mathbb{E}\left[\int_t^{t+h}(\partial_s + L^{\mathbf{v}})\,V^i(s,\mathbf{X}_s^{\mathbf{v}})\,\mathrm{d}s\Big|\mathbf{X}_t^{\mathbf{v}}=\mathbf{x}\right],
\end{aligned}$$

where the zero-mean property of the Itô integral is used in the last equality. Once again, by assuming the conditions for Fubini's theorem and Lebesgue's theorem of differentiation are satisfied, it follows that

$$\lim_{h\downarrow 0} I_2 = \lim_{h\downarrow 0}\frac{1}{h}\int_t^{t+h}\mathbb{E}\left[(\partial_s + L^{\mathbf{v}})\,V^i(s,\mathbf{X}_s^{\mathbf{v}})\Big|\mathbf{X}_t^{\mathbf{v}}=\mathbf{x}\right]\,\mathrm{d}s = (\partial_t + L^{\mathbf{v}})V^i(t,\mathbf{x}).$$

Insterting the limits obtained in (A.4), (A.5) into (A.3) we find, for every admissible control $\mathbf{v}_t\in\mathcal{A}$, that

$$-\partial_t V^i(t,\mathbf{x}) \leq L^{\mathbf{v}}V^i(t,\mathbf{x}) + f^i(t,\mathbf{x},v_t^i,\mathbf{v}_t^{-i}), \qquad \text{(A.6)}$$

Recall that in (3.3) the value function is defined as a infimum over all admissible conotrls. In (A.2), an arbritary control $\mathbf{v}_t\in\mathcal{A}$ was considered, and the infimum is

therefore removed. Thus, the equality was replaced by an inquality for the value function. By once again considering the infimum, (A.6) becomes

$$-\partial_t V^i(t,\mathbf{x}) = \inf_{v_t^i \in \mathcal{U}^i} \Big\{ L^{\mathbf{v}} V^i(t,\mathbf{X}_t^{\mathbf{v}}) + f^i(t,\mathbf{x},v_t^i,\mathbf{v}_t^{-i}) \Big\}. \tag{A.7}$$

The optimal policy is then obtained as

$$\pi^i(t,\mathbf{x}) \in \arg\min_{v_t^i \in \mathcal{U}^i} \Big\{ L^{\mathbf{v}} V^i(t,\mathbf{X}_t^{\mathbf{v}}) + f^i(t,\mathbf{x},v_t^i,\mathbf{v}_t^{-i}) \Big\}, \qquad (t,\mathbf{x}) \in [0,T] \times \mathbb{R}^{dN},$$

where $\pi^i : [0,T] \times \mathbb{R}^{dN} \to \mathcal{U}^i$ is an individual deterministic map. The deterministic map for the system as a whole is the vector concatenation of all invidiual contributions, i.e., $\boldsymbol{\pi} \coloneqq (\pi^1, \dots, \pi^N)$. At each time t the control is obtained as $v_t^i = \pi^i(t, \mathbf{X}_t^{\mathbf{v}})$, i.e., the decision only depends on the current time and current state. Hence, when this is valid, the control is called a *Markov control policy*. We assume the set of minimizers above is nonempty. Since the diffusion term of the forward SDE, in this thesis, does not depend on the control $\mathbf{u}_t$, then (A.7) is a second-order semi-linear HJB. By explicitly extracting the diffusion term from the operator L^v that does not depend on the control, the HJB equation is expressed as

$$\partial_t V^i + \frac{1}{2} \operatorname{Tr}(\Sigma\Sigma^\top \Delta_x V^i) + \mathcal{H}^i(t,\mathbf{x},\nabla_x V^i) = 0, \quad (t,\mathbf{x}) \in [0,T) \times \mathbb{R}^{dN}, \tag{A.8a}$$

$$V^i(T,\mathbf{x}) = g(\mathbf{x}), \quad \mathbf{x} \in \mathbb{R}^{dN}, \tag{A.8b}$$

For $i \in \mathcal{I}$. Here the space-time dependence (t,x) has been excluded for brevity unless otherwise specified. Furthermore, the Hamiltonian $\mathcal{H}^i$ of (A.8a) is defined as

$$\mathcal{H}^i(t,\mathbf{x},p) \coloneqq \inf_{v_t^i \in \mathcal{U}^i} \Big\{ \big\langle \mathbf{b}(t,\mathbf{x},v_t^i,\mathbf{v}_t^{-i}), p \big\rangle + f^i(t,\mathbf{x},v_t^i,\mathbf{v}_t^{-i}) \Big\}, \qquad (t,\mathbf{x}) \in [0,T] \times \mathbb{R}^{dN}.$$

Here, $p \in \mathbb{R}^{dN}$ is a placeholder for the spatial derivative of the value function. The terminal condition (A.8b) is derived by considering the value function (3.2) at the terminal time T, that is,

$$V^i(T,\mathbf{x}) = \inf_{u_t^i \in \mathcal{U}^i} J^i(T,\mathbf{x};\mathbf{u}_t) = \inf_{u_t^i \in \mathcal{U}^i} \mathbb{E}\left[g^i(\mathbf{X}_T^{\mathbf{u}}) | \mathbf{X}_T^{\mathbf{u}} = \mathbf{x} \right] = g^i(\mathbf{x}).$$

B

Drone game results

Here we present further results for the drone game. The section follows the structure used throughout the thesis. That is, we begin with presenting the results for state process $X = (X_t^u)_{t\in[0,T]}$ for the Deep fictitious play method and then the results for the Robust deep fictitious play algorithm. This structure is then repeated for the backward process $Y = (Y_t)_{t\in[0,T]}$ and finally the control process $Z = (Z_t)_{t\in[0,T]}$.

Since no analytic solution is available for this problem it is difficult to draw any conclusions from these figures. For that we rely on the results presented in Section 6.3.1. What one can say is that, from Figure B.3, the Deep fictitious play algorithm has converged to fulfill the terminal condition. However, since the state process and the backward process are approximated simultaneously, the algorithm has converged to an incorrect terminal condition. Thus, we get the incorrect solution presented in Section 6.3.1.

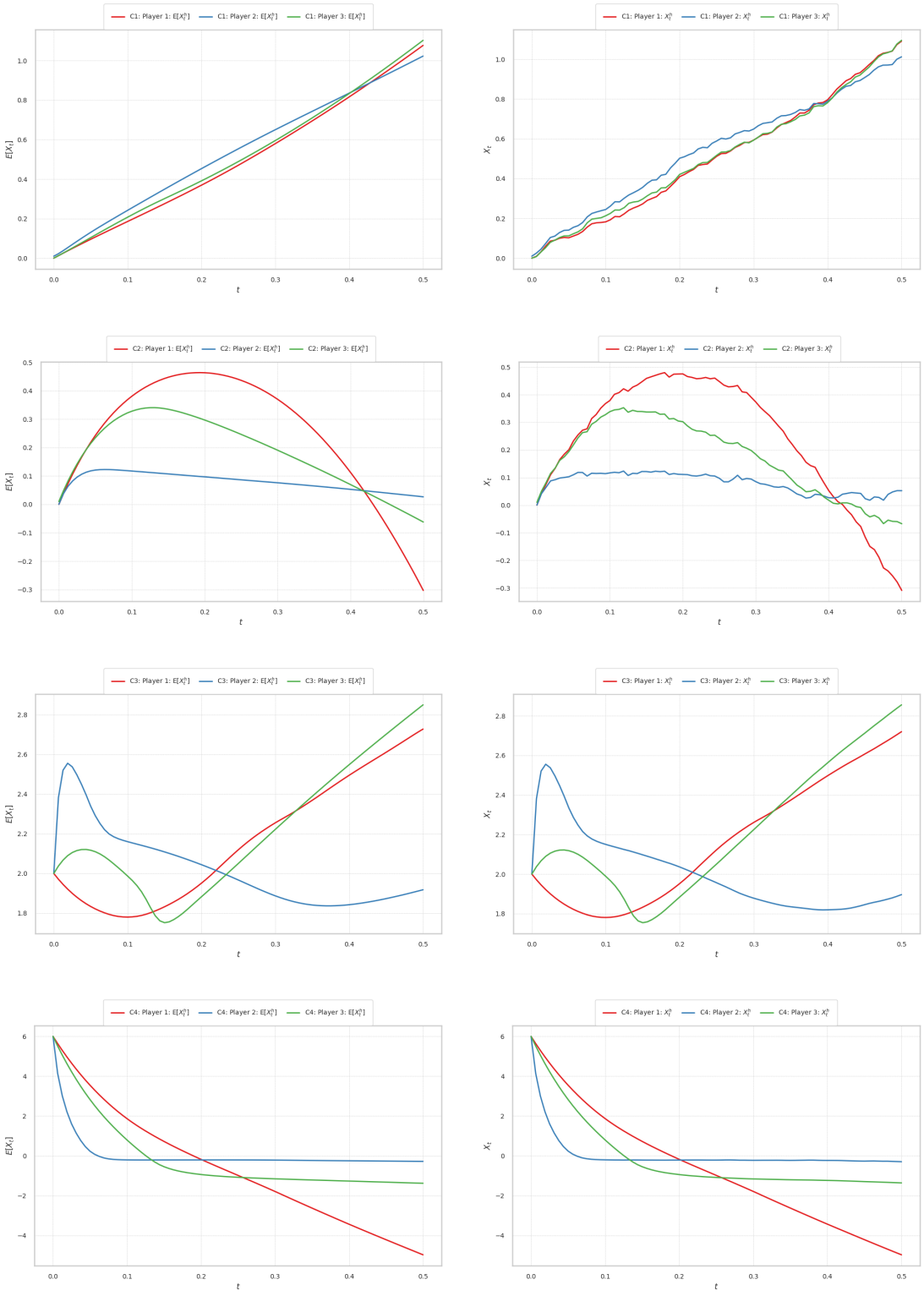

Figure B.1: The figure illustrates the state process generated by the Deep fictitious play algorithm. The rows show each component of the process and the columns represent the expectation of that component and a representative trajectory respectively. The different colors represent the different players, player one in red, player two in blue and player 3 in green.

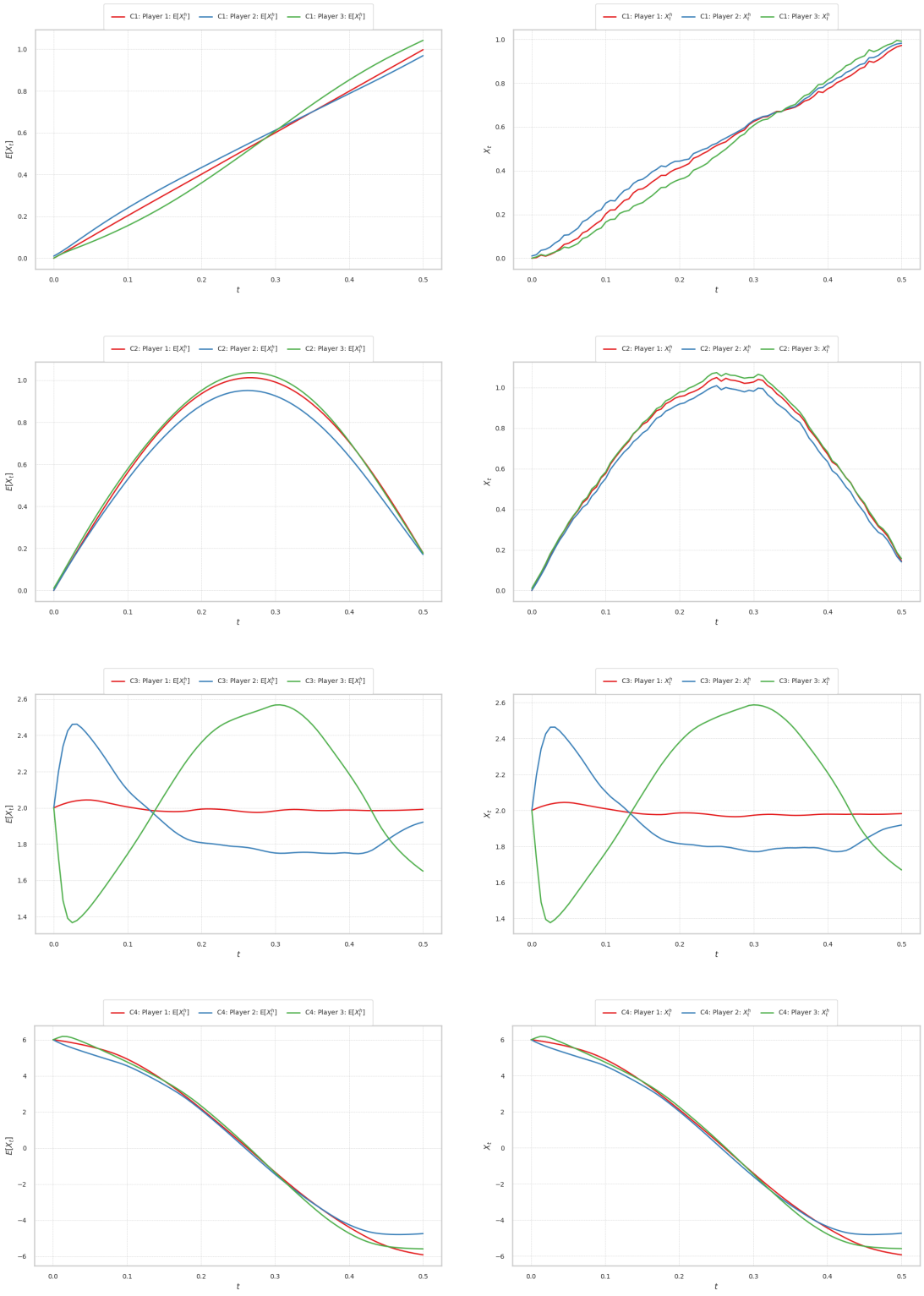

Figure B.2: The figure illustrates the state process generated by the Robust deep fictitious play algorithm. The rows show each component of the process and the columns represent the expectation of that component and a representative trajectory respectively. The different colors represent the different players, player one in red, player two in blue and player 3 in green.

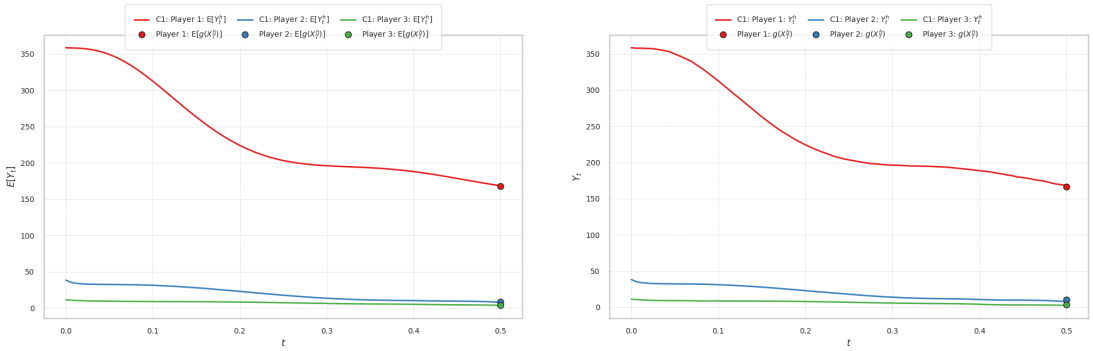

Figure B.3: The figure illustrates the backward process generated by the Deep fictitious play algorithm. The columns represent the expectation of the process and a representative trajectory respectively. The different colors represent the different players, player one in red, player two in blue and player 3 in green. Finally, the circle denotes the approximated terminal condition.

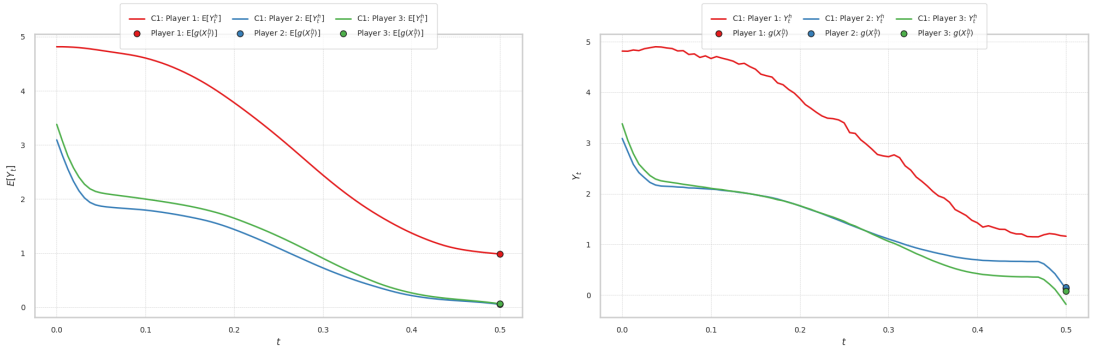

Figure B.4: The figure illustrates the backward process generated by the Robust deep fictitious play algorithm. The columns represent the expectation of the process and a representative trajectory respectively. The different colors represent the different players, player one in red, player two in blue and player 3 in green. Finally, the circle denotes the approximated terminal condition.

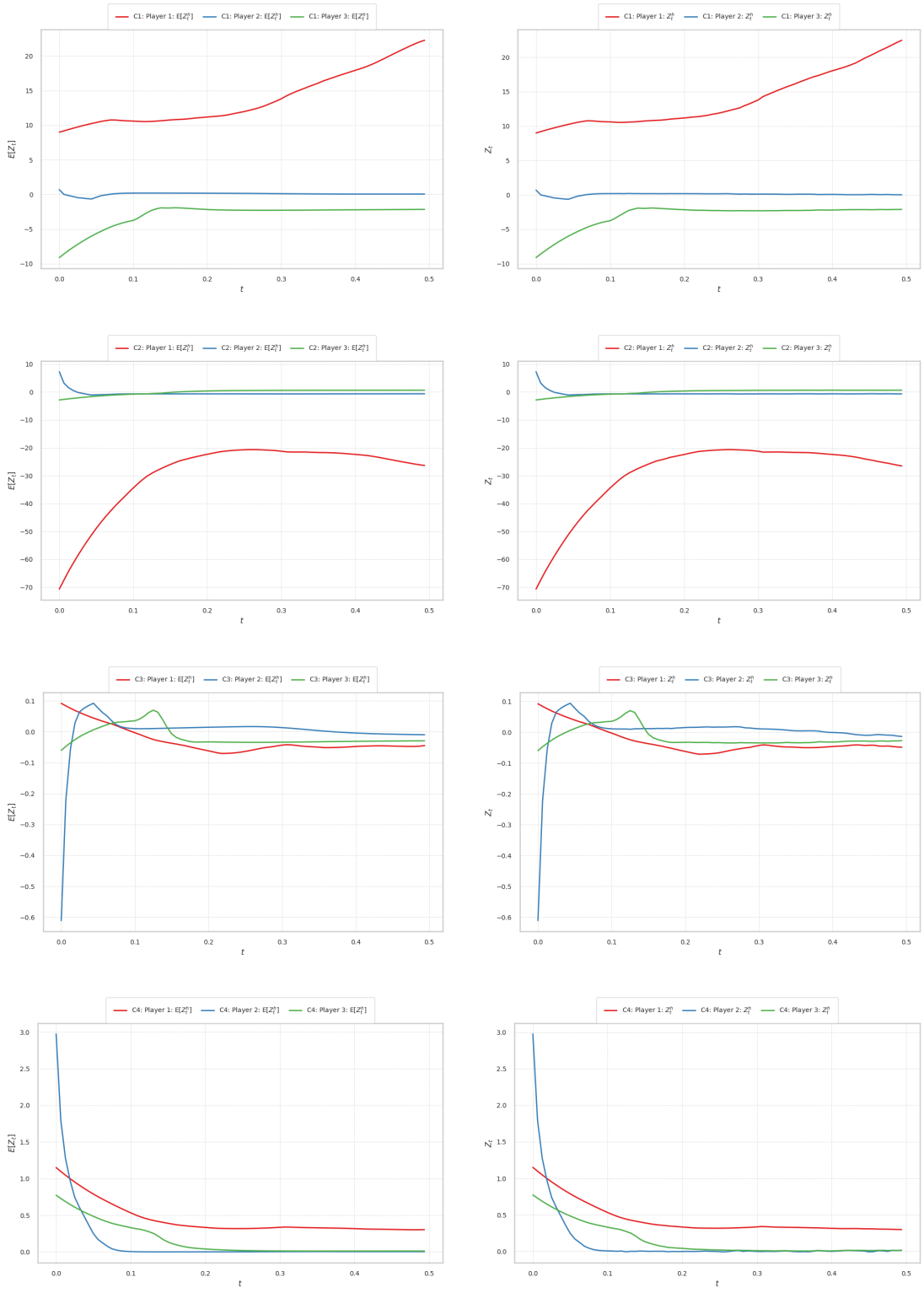

Figure B.5: The figure illustrates the Z-process generated by the Deep fictitious play algorithm. The rows correspond to the different components of the process while the columns represent the expectation of that component and a representative trajectory respectively. Further, the colors denote the different players, player one in red, player two in blue and finally player three in green.

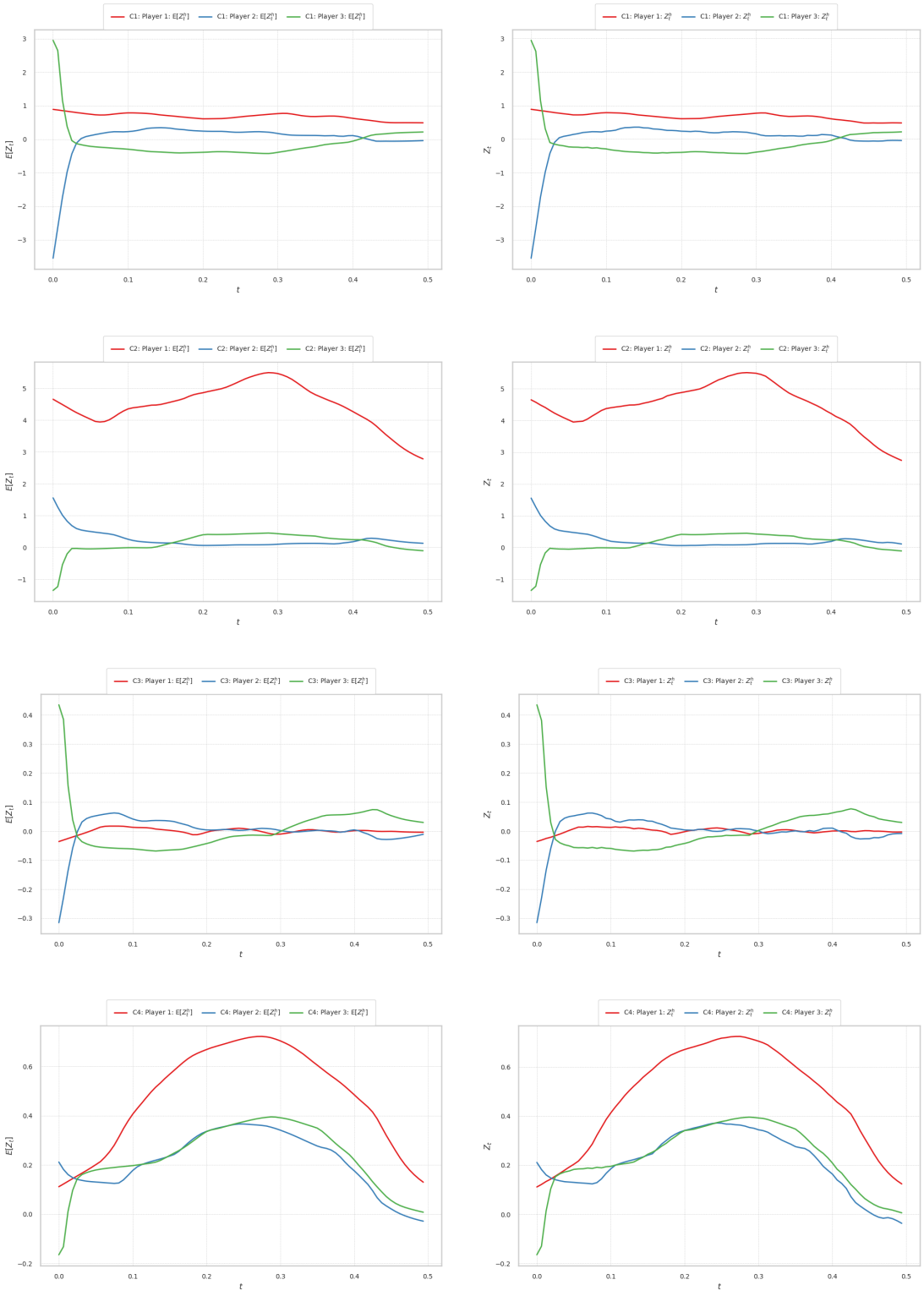

Figure B.6: The figure illustrates the Z-process generated by the Robust deep fictitious play algorithm. The rows correspond to the different components of the process while the columns represent the expectation of that component and a representative trajectory respectively. Further, the colors denote the different players, player one in red, player two in blue and finally player three in green.

DEPARTMENT OF MATHEMATICAL SCIENCES
CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden
www.chalmers.se