Neural networks for stochastic control and decision making in mathematical finance

(met een samenvatting in het Nederlands)

Proefschrift

ter verkrijging van de graad van doctor aan de
Universiteit Utrecht
op gezag van de
rector magnificus, prof. dr. H.R.B.M. Kummeling,
ingevolge het besluit van het college voor promoties
in het openbaar te verdedigen op

maandag 18 december 2023 des middags te 4.15 uur

door

Kristoffer Herbert Andersson

geboren op 26 maart 1989
te Gothenburg, Zweden

Promotor:

Prof. dr. ir. C.W. Oosterlee

Copromotor:

Dr. Ir. L.A. Grzelak

Beoordelingscommissie:

Prof. dr. ir. J.E. Frank

Prof. dr. P.A. Forsyth

Prof. dr. R.J.A. Laeven

Prof. dr. A. Papapantoleon

Prof. dr. C. Reisinger

Summary

As the title suggests, this thesis is devoted to machine learning algorithms for stochastic control with applications in mathematical finance. More specifically, three classes of stochastic control problems are considered. Namely, problems in which:

1. The *dynamic programming principle* (DPP) holds and the state process is decoupled. With a decoupled state process, we mean a state process that can be sampled from independently of the control process, *i.e.,* the dynamics (or update rules) do not take the control process as an input. Chapters 2 and 3 treat problems from this class;

2. The DPP holds and the state process is coupled. Contrary to decoupled state processes, a coupled state process depends on the control process and we cannot sample from it without access to the control of the system. In Chapter 4, problems from this class are considered;

3. The DPP does not hold and the state process is coupled. This is the most general problem class in this thesis and Chapter 5 is devoted to these kinds of problems.

In this thesis, we propose neural network-based approximation methods for each of the classes introduced above. Although, each strategy needs to be adjusted to the specific problem at hand, our main belief is to use as much mathematical structure as possible. For instance, when the DPP holds, we aim to incorporate this structural knowledge in order to increase explainability as well as to construct efficient and accurate approximation methods. This opens up for problem-specific algorithms, constructed to take advantage of much of the available mathematical structure, see Chapters 2-3. In Chapter 4, we have the DPP satisfied but lose the advantage of having a decoupled state process. However, we can still benefit from the DPP and several possible mathematical reformulations of the problem to end up with an algorithm that is appropriate for the specific problem considered. Finally, in Chapter 5, most of the structure, and also theoretical results (existence and uniqueness of a solution) are at best hard to verify and at worst do not hold. Therefore, we take on a more pragmatic strategy and present data-driven methods which approximate the relevant problem in a more direct way, without much mathematical manipulation.

Chapters 2 and 3: (based on [1, 2])
In these two chapters, we consider some risk management aspects of high-dimensional options with early-exercise features, more precisely Bermudan (and American) options. The proposed algorithm is divided into two phases. In the first phase, we use neural networks to approximate the optimal exercise strategy and in the second phase we employ neural networks

to approximate pathwise option values. By focusing directly on the exercise strategy, instead of the option (or continuation) values, we achieve a high accuracy. This accuracy is then carried on to the second phase in which the pathwise option values are approximated, based on the (close to) optimal exercise strategy. It is shown that this method is highly accurate in comparison to classical methods, especially in high dimensions when the latter often run into difficulties.

In Chapter 3, the algorithm is extended to a portfolio of high-dimensional options with early-exercise features, which implies high-dimensional problems, both in the sense of the number of underlying assets and in the sense of the number financial products in the portfolio. Moreover, we introduce *counterparty credit risk* (CCR) and approximate the *credit valuation adjustment* (CVA). An important conclusion made in Chapter 3 is that in the presence of default risk of the counterparty, the optimal exercise strategy should be decided at the level of the entire portfolio. This implies that the market practice of valuating options one by one, independent of the counterparty as well as the entire outstanding portfolio, leads to a sub-optimal exercise strategy.

Chapter 4: (based on [3])
In Chapter 4, we consider a standard stochastic optimal control problem with the state process given by a controlled diffusion-type SDE and a cost functional satisfying the DPP. We use the well-known connection between a stochastic control problem and a *forward backward stochastic differential equation* (FBSDE) to formulate the associate FBSDE. To find an approximate solution to the FBSDE, we propose a neural network-based approximation method, in which the loss function is constructed such that properties from both the stochastic control problem and the FBSDE are incorporated. In this way, we include more mathematical structure into the methodology than similar FBSDE solution methods such as the *deep BSDE method* proposed in [4]. The price we have to pay is that the usage of our method is limited to FBSDEs stemming from stochastic control, whereas the deep BSDE method, at least conceptually, is applicable to general FBSDEs. On the other hand, we show by a numerical example that a direct extension of the classical deep BSDE method to coupled FBSDEs, fails for a simple linear-quadratic control problem, and we motivate why our method converges, both with mathematical insights and numerical experiments.

Chapter 5: (based on [5])
As already stated, in the final research chapter, we consider a general type of stochastic control problems in which the DPP does not hold. More specifically, we are concerned with time-inconsistent portfolio optimization problems in which the investor is allowed to trade in a bond, stocks and options. The motivation for going beyond the classical portfolio optimization theory and allowing for trades also in options is twofold: *i)* with options in the portfolio we add flexibility in shaping a desirable terminal wealth distribution since options offer a significantly less symmetrical terminal wealth distribution than stocks. *ii)* many funds are prohibited from using leverage in their portfolios and trading in options can be seen as a way to achieve a leveraged position. The price to pay for the leverage here is only the option premium and, hence, the large downside risk connected to a leveraged position can be avoided.

With higher flexibility to shape the terminal wealth distribution, it is rational to consider a less symmetrical objective function than the classical mean-variance optimization target. We formulate the problem with a general objective function, taking the probability law of the terminal wealth as input. More specifically, we explore some objectives which aim to control the tail distribution of the terminal wealth.

With the general setting as described above, most mathematical properties of classical portfolio optimization or stochastic control theory are no longer available. For instance, uniqueness of an optimal allocation strategy is not guaranteed, and, in some special cases, we can even show that there are infinitely many optimal allocation strategies. With this in mind, our neural network-based approximation method, named Deep Time-Inconsistent Portfolio optimization with stocks and Options (D-TIPO), uses a direct, data-driven approach in which the loss function coincides with an empirical version of the objective function.

With the resulting novel algorithms in this thesis, we aim to demonstrate the applicability of neural networks as function approximators for challenging problems in mathematical finance. In particular, the additional value of the machine learning methodologies is found for high-dimension problems with a complicated structure. Moreover, we wish to show that it is worth striving for a high level of mathematical structure in the algorithms, in order to enhance explainability, not only explaining how an algorithm works, but also for the purpose of constructing better algorithms when one analyzes the details of the underlying components. With this said, it should be pointed out that the usage of machine learning in applied mathematics is still in its early stages. In order to reach the level of full explainability, as it is available for the more mature classical approximation methods, much more research needs to be conducted.

Samenvatting

Zoals de titel vam dit proefschrift al suggereert, is deze scriptie gewijd aan machine learning algoritmen voor stochastische regeltechniekproblemen met toepassingen in de financiële wiskunde. Meer specifiek worden drie klassen van stochastische regeltechniekproblemen bestudeerd. Namelijk, problemen waarin:

1. Het *dynamisch programmeer principe* (DPP) geldt en de stochastische processen ontkoppeld zijn. Met een ontkoppeld proces bedoelen we een proces dat onafhankelijk van het controleproces kan worden gesimuleerd, d.w.z. de dynamiek (of update-regels) heeft het controleproces niet als input. Hoofdstukken 2 en 3 behandelen problemen uit deze klasse;

2. Het DPP geldt en de stochastische processen zijn gekoppeld. In tegenstelling tot ontkoppelde processen, hangt een gekoppeld stochastisch proces af van het controleproces en kunnen we het proces niet simuleren zonder de controleparameters van het systeem. In Hoofdstuk 4 wordt een probleem uit deze klasse bestudeerd;

3. Het DPP geldt niet en de stochastische processen zijn gekoppeld. Dit is de meest algemene probleemklasse, en in dit proefschrift is Hoofdstuk 5 gewijd aan dit soort problemen.

In dit proefschrift introduceren we neurale netwerk gebaseerde benaderingsmethoden, voor elk van de hierboven geïntroduceerde klassen. Hoewel elke methode moet worden aangepast aan het specifieke probleem, is onze insteek om zoveel mogelijk wiskundige structuur te gebruiken bij het oplossen van een probleem. Bijvoorbeeld, wanneer het DPP geldt, streven we ernaar deze kennis te gebruiken om de uitlegbaarheid van het algoritme te verbeteren, en ook om efficiënte en nauwkeurige oplosmethoden te ontwikkelen. Dit opent de deur naar meer probleemspecifieke algoritmen, geconstrueerd om optimaal gebruik te maken van de beschikbare structuur, zie de Hoofdstukken 2-3. In Hoofdstuk 4 behouden we het DPP, maar verliezen we het aspect van een ontkoppeld stochastisch proces. We kunnen echter nog steeds profiteren van het DPP en van een wiskundige formulering om een algoritme te ontwikkelen dat bijzonder geschikt is voor deze specifieke probleemklasse. Ten slotte, in Hoofdstuk 5, zijn de structuur, en ook theoretische resultaten (existentie en uniciteit van een oplossing), op zijn best moeilijk te verifiëren en soms zijn ze niet van toepassing. Daarom hanteren we een pragmatische strategie en presenteren we op data gebaseerde methoden die de relevante problemen op een directe manier benaderen, zonder veel wiskundige manipulatie.

Hoofdstukken 2 en 3: (gebaseerd op [1, 2])
In deze twee hoofdstukken bestuderen we aspecten van risicobeheer van hoog-dimensionale

financiële opties met vroegtijdige uitoefeningsmogelijkheden, meer specifiek Bermuda (en Amerikaanse) opties. Het voorgestelde algoritme bestaat in twee fasen. In de eerste fase gebruiken we neurale netwerken om de optimale uitoefeningsstrategie te benaderen en in de tweede fase gebruiken we neurale netwerken om de optiewaarden op basis van de gesimuleerde paden te benaderen. Door ons te richten op de uitoefeningsstrategie, in plaats van de optie- (of continuerings-) waarden, bereiken we een hoge nauwkeurigheid. Van deze nauwkeurigheid profiteren we in de tweede fase, waarin de optiewaarden op basis van de (bijna) optimale uitoefeningsstrategie worden berekend. We tonen aan dat deze methode zeer nauwkeurig is in vergelijking met klassieke methoden, vooral in hoge dimensies.

In Hoofdstuk 3 wordt het algoritme uitgebreid naar een portefeuille met hoogdimensionale Bermuda (en Amerikaanse) opties, wat leidt tot problemen met een hoge dimensie, zowel in termen van het aantal onderliggende activa als het aantal producten in de portefeuille. Bovendien introduceren we *tegenpartij kredietrisico* (CCR) en benaderen we de *kredietwaarde aanpassing* (CVA). Een belangrijke conclusie in Hoofdstuk 3 is dat bij het risico op faillissement van de tegenpartij, de optimale uitoefeningsstrategie op het niveau van de hele portefeuille moet worden gemaakt. Dit impliceert dat de marktpraktijk om opties afzonderlijk van elkaar te waarderen, onafhankelijk van de kredietwaardigheid van de tegenpartij en de resterende portefeuille, leidt tot een suboptimale uitoefeningsstrategie.

Hoofdstuk 4: (gebaseerd op [3])
In Hoofdstuk 4 bestuderen we een stochastisch optimaal regeltechniekprobleem met een stochastisch proces dat beschreven wordt door een gecontroleerde stochastische differentiaalvergelijking en een kostenfunctionaal die aan het DPP voldoet. We gebruiken de connectie tussen een stochastisch regeltechniekprobleem en een *voorwaarts terugwaartse stochastische differentiaalvergelijking* (FBSDE) om het bijbehorende FBSDE wiskundig te formuleren. Om een numerieke oplossing voor de FBSDE te vinden, definiëren we een neurale netwerk-gebaseerde benaderingsmethode, waarbij de verliesfunctie is geconstrueerd zodat de belangrijke eigenschappen van zowel het regelprobleem als van de FBSDE worden meegenomen. Op deze manier introduceren we meer wiskundige structuur dan vergelijkbare FBSDE-methoden zoals de *deep BSDE method* in [6]. De prijs die we moeten betalen, is dat de voorgestelde methode beperkt blijft tot FBSDEs die gerelateerd zijn aan stochastische regeltechniek, terwijl de deep BSDE-methode, conceptueel, gebruikt kan worden voor algemene FBSDEs. Tegelijkertijd laten we zien, aan de hand van een numeriek voorbeeld, dat de "klassieke" deep BSDE-methode divergeert voor eenvoudige gekoppelde lineair-kwadratische regelproblemen, en we motiveren waarom onze methode werkt, zowel met wiskundige inzichten als met numerieke experimenten.

Hoofdstuk 5: (gebaseerd op [5])
Zoals eerder vermeld, behandelen we in het laatste hoofdstuk een algemene vorm van stochastische regelproblemen waarvoor het DPP niet geldt. Meer specifiek gaan we in op tijd-inconsistente portefeuille-optimalisatieproblemen waarin de belegger in obligaties, aandelen en opties mag handelen. De motivatie om verder te gaan dan de klassieke portefeuille-optimalisatietheorie en ook handel in opties toe te staan, is tweeledig: *i)* met opties in de

portefeuille voegen we flexibiliteit toe bij het vormgeven van een gewenste verdeling van het eindvermogen, aangezien opties aanzienlijk minder symmetrische eindvermogensverdelingen bieden dan aandelen. *ii)* Voor veel fondsen is het verboden om de hefboomwerking van opties te gebruiken in hun portefeuilles om te speculeren. De prijs voor de hefboomwerking is hier echter slechts de optiepremie, en dus wordt het grote neerwaartse risico van hefboomwerking vermeden.

Met de flexibiliteit om de eindvermogensverdeling vorm te geven, is het zinvol om een minder symmetrische doelfunctie dan de klassieke gemiddelde-variantie doelfunctie te kiezen. We formuleren het probleem met een algemene doelfunctie, gebaseerd op de distributie van het eindvermogen, en onderzoeken enkele doelfuncties die gericht zijn op het beheersen van de staartverdeling van het eindvermogen.

In de hierboven beschreven setting zijn de meeste wiskundige eigenschappen van klassieke portefeuille-optimalisatie of stochastische regeltheorie niet langer van toepassing. Bijvoorbeeld, de uniciteit van een optimale strategie is niet gegarandeerd en in sommige gevallen kunnen we zelfs aantonen dat er oneindig veel optimale strategieën zijn. Om die reden gebruikt onze neurale netwerk-gebaseerde benaderingsmethode een directe, op data gebaseerde, aanpak waarbij de doelfunctie samenvalt met een empirische versie van de doelfunctie.

Met de resulterende nieuwe algoritmen in dit proefschrift willen we de toepasbaarheid van neurale netwerken als functiebenadering voor uitdagende problemen in de financiële wiskunde aantonen. In het bijzonder is er toegevoegde waarde van de machine learning-methodieken voor hoogdimensionale problemen met een gecompliceerde structuur. Bovendien willen we aantonen dat het streven naar een hoge mate van wiskundige structuur in de algoritmen belangrijk is, om de werking van de algoritmen te verklaren, niet alleen om uit te leggen hoe een algoritme werkt, maar ook om betere algoritmen te construeren wanneer men de details van de onderliggende componenten analyseert. Echter, het gebruik van machinaal leren in de toegepaste wiskunde bevindt zich nog in de beginfase. Om het niveau van volledige verklaarbaarheid te bereiken, zoals met klassieke benaderingsmethoden, moet nog veel meer onderzoek worden uitgevoerd.

Table of Contents

1

Introduction

The introduction is divided into three parts. Firstly, significant ideas from stochastic optimal control theory are presented. These concepts are then connected to financial mathematics in the following section. Here, the focus is on specific topics such as option valuation, risk management, and portfolio optimization. Lastly, the structure of the thesis is explained, highlighting how it is organized and what each section covers.

1.1 Stochastic control

Stochastic control is a central topic in all chapters in this thesis. More precisely, we are concerned with control problems on a finite time horizon, usually from $t = 0$ until $t = T \in \mathbb{R}^+$. In general, a stochastic control problem consists of three components, the *state process*, the *control process* and the *cost functional*. Below, these three components are described one by one in the context of this thesis:

- **State process**
  The state of the system evolves over time in an uncertain environment, described on a filtered probability space $(\Omega, \mathcal{F}, \mathbb{P})$. Given $\omega \in \Omega$, the state process, denoted $X = (X_t)_{t\in[0,T]}$ is a mapping $t \mapsto X_t(\omega)$. For simplicity, we assume that X is a *Markov process*.

- **Control process**
  The state process $t \mapsto X_t$, or an associated process $(t, X_t) \mapsto V_t$ is typically controlled by some $\alpha = (\alpha_t)_{t\in[0,T]}$. To emphasise that X is controlled by α, a superscript $X^\alpha = (X^\alpha_t)_{t\in[0,T]}$ is sometimes used. The set of *admissible*, or allowed controls, is denoted by $\mathcal{A}$. From a practical perspective, $\mathcal{A}$ can be interpreted as physical constraints on the control, for instance, the kind of trading strategies an investor is allowed to implement. These restrictions could be motivated from a regulatory viewpoint and/or trading directives from an investor's client. Beyond this, $\mathcal{A}$ has a more involved mathematical definition which is important for the framework to be well defined. The mathematical details are outlined in the chapters to follow.

- **Cost/objective functional**

  The cost functional or, equivalently, the objective function, denoted by J, aims to measure the performance of the control process when acting on the state process. The first type of cost functional assumes a control in the form of a *stopping time* which yields an *optimal stopping* problem. The cost functional is then given by

$$J(\tau; t, x) = \mathbb{E}\Big[\int_t^\tau L(s, X_s)\mathrm{d}s + g(X_\tau)\Big]. \tag{1.1}$$

  Here τ is an X-stopping time, which essentially means that the information required to determine whether or not to stop at some time t is completely contained in X_t. We denote the set of X-stopping times by $\mathcal{T}$, and the set of X-stopping times, greater than or equal to t, by $\mathcal{T}_t$. A typical example of a problem of this form is to find a fair value of an American option.

  The second kind of cost functional considered is perhaps the most common in stochastic optimal control and takes the form

$$J(\alpha; t, x) = \mathbb{E}\Big[\int_t^T L(s, X_s^\alpha, \alpha_s)\mathrm{d}s + g(X_T^\alpha)\,\big|X_t^\alpha = x\Big]. \tag{1.2}$$

  Here L is referred to as the *running cost* and aims to penalize deviation from an optimal state on the time interval $[t, T]$ and similarly, g is referred to as the *terminal cost* and aims to penalize deviation from the preferred state at the terminal time.

  The final cost functional considered in this thesis takes a more general form. Instead of an expected value as in (1.2) and (1.1), it is expressed as a function of the probability law $\mathcal{L}$ of the state process

$$J(\alpha) = U(\mathcal{L}[X^\alpha]). \tag{1.3}$$

  Note that while the cost functionals in (1.2) and (1.1) are parametrized with t, x, indicating that the cost is conditioned on $X_t = x$ and computed on the interval $[t, T]$, (1.3) is not. The reason for this is that the control problems which use this more general form of the cost functional are often only considered from the perspective of $t = 0$. Under which assumptions a time-dependent cost functional is used and what impact this has on the choice of approximation algorithm, are some of the main topics in this thesis. A typical problem of the form (1.3) is the classical *mean variance optimization problem.*

Regardless of whether (1.1)-(1.2) or (1.3) is considered, the aim is to find an admissible control such that the cost functional is minimized, which in each case is formulated as

$$\tau^* \in \inf_{\tau\in\mathcal{T}} J(\tau; 0, x_0), \quad \alpha^* \in \inf_{\alpha\in\mathcal{A}} J(\alpha; 0, x_0), \quad \text{and} \quad \alpha^* \in \inf_{\alpha\in\mathcal{A}} J(\alpha). \tag{1.4}$$

The above raises some important question about the strategy α^* (and τ^*). Is the infimum attainable? Is it unique? What properties can we expect from J, evaluated at α^* (or τ^*)? The answer to all these questions is that it depends on *i)* the underlying model, *ii)* the set of admissible strategies, and *iii)* the regularity of the functions L and g when (1.2) or (1.1) is considered and U when (1.3) is considered. In this thesis, we mostly assume that *i)* and *ii)* are

nice enough to satisfy the necessary relevant conditions, while *iii)* is explored in some more detail.

Assuming the infimums in (1.4) exist, the optimal costs at $(t = 0, x = x_0)$ are given by

$$J(\tau^*; 0, x_0), \quad J(\alpha^*; 0, x_0), \quad \text{and} \quad J(\alpha^*). \tag{1.5}$$

In the next section, we discuss whether or not a strategy α^* (or τ^*) is optimal at any $(t, X_t^{\alpha^*})$ or whether it is optimal to update the strategy along the random trajectory. Or, equivalently, for $\alpha^1, \alpha^2 \in \mathcal{A}$, with $\alpha^1 \in \inf_{\alpha\in\mathcal{A}} J(\alpha; 0, x_0)$ and $\alpha^2 \in \inf_{\alpha\in\mathcal{A}} J(\alpha; t, X_t^{\alpha^1})$, does it hold that $J(\alpha^1; t, X_t^{\alpha^1}) = J(\alpha^2; t, X_t^{\alpha^1})$. This is related to *Bellman's principle of dynamic programming*, which is treated in the section below.

1.1.1 Bellman's principle of dynamic programming and time consistency

In this section, we consider stochastic control problems without a stopping time component to keep the notation simple. The extension to include stopping times is straight forward and in the chapters to follow, this is outlined in detail when needed. Bellman's principle of dynamic programming states that for $h \in [0, T - t]$

$$J(\alpha^*; t, x) = \inf_{\alpha\in\mathcal{A}} \mathbb{E}\Big[\int_t^T L(s, X_s^\alpha, \alpha_s)\mathrm{d}s + J(\alpha; t + h, X_{t+h}^\alpha) \,|X_t = x\Big]. \tag{1.6}$$

In words, this means that finding an optimal strategy on $[t, T]$ is equivalent to first finding an optimal strategy on $[t, t + h]$ and then determine a new optimal strategy on $[t + h, T]$. Iteratively, this implies that the strategy which is optimal on $[t, T]$ is also optimal on each subinterval of $[t, T]$. This principle is used in many classical algorithms since one large global optimization problem can be transformed into multiple smaller local optimization problems. Under realistic standard assumptions, (1.6) holds for problems with cost functionals (1.1) and (1.2), however, in general, there is no version which holds for (1.3). To better understand why the *dynamic programming principle* (DPP) fails to hold for many cost functionals, we resort to a special form of (1.3) in which the cost functional is a function of the expected terminal state

$$J(\alpha) = \sum_{i=1}^N U_i(\mathbb{E}[u_i(X_T^\alpha)]), \tag{1.7}$$

where U_i and u_i are some suitable functions. A typical example which belongs to this class of cost functionals is the classical mean-variance optimization problem. To see this, consider $X_T \in \mathbb{R}$ and $\lambda > 0$. The cost functional can then be rewritten as $-\mathbb{E}[X_T^\alpha] + \lambda \mathrm{Var}[X_T^\alpha] = -\mathbb{E}[X_T^\alpha] + \lambda \cdot \Big(\mathbb{E}[(X_T^\alpha)^2] - \mathbb{E}[X_T^\alpha]^2\Big)$. Since the law of iterated expectations then does not hold for all terms in (1.7), *i.e.,*

$$\mathbb{E}\Big[U_i(\mathbb{E}[u_i(X_T^\alpha) \,|X_{t+h}^\alpha]) \,|X_t^\alpha\Big] \neq U_i(\mathbb{E}[u_i(X_T^\alpha) \,|X_t^\alpha = x]),$$

neither does the dynamic programming principle. The class of problems, for which the DPP does not hold is often referred to as *time-inconsistent* problems. As a consequence, for these problems the optimization procedure cannot be divided into multiple subproblems and a whole

different class of approximation algorithms needs to used. It should however be pointed out that in some special cases it is still possible to transform a time-inconsistent problem into a *time-consistent* problem, see *e.g.,* [7].

1.1.2 Forward and backward algorithms with local or global optimization

As mentioned in the previous sections, the DPP holds for problems with cost functionals of the form (1.1) and (1.2). This has a major impact in the choice of algorithm. Another important aspect when deciding between algorithms is the way the control is incorporated in the state process X. For the problems with cost functionals (1.2) and (1.3), the control component affects the dynamics of the state process, which is indicated by superscript α. This implies that the state process cannot be sampled independently of the control process. For optimal stopping problems, on the other hand, the control of the system is given by a stopping time, which does not affect the evolution in time of the state process. This opens up for algorithms in which the state process is sampled independently of the rest of the problem. In this thesis, we refer to the former as problems with *coupled state process* and the latter as problems with *decoupled state process.*

Another distinction made in this thesis is between *forward* and *backward* type algorithms. This refers to the temporal order in which a problem is solved. The temporal order often affects whether we consider a *local* or *global optimization* algorithm. This refers to whether the optimization is performed for one time step at a time or for multiple time steps simultaneously. Often, but not always, a forward algorithm uses global optimization and a backward algorithm resorts to local optimization. For instance, if an optimization problem is solved by dynamic programming in a backward induction scheme, starting from right before the terminal time and working its way back, step by step, to the initial time, then we are dealing with a backward type algorithm with local optimization.

Below, an example aiming to demonstrate the difference between local and global optimization is given.

Example 1.1.1. *Assume a time discretization, $0 = t_0 < t_1 < \cdots < t_N = T$, with a control at t_n denoted by θ_n. The entire control process, prior to the control at t_n, is denoted by $\theta_n = \{\theta_0, \theta_1, \ldots, \theta_{n-1}\}$. The control, which can be in the form of a regular control of the state process or in the form of a stopping time, is here given by the feedback map taking the current state as input, i.e., $\theta_n(X_{t_n}^{\boldsymbol{\theta}_n})$. the superscript $\boldsymbol{\theta}_n$ indicates that the state process is influenced by the historical control. We assume that the state process is stochastic by some influence of an $\mathcal{F}_n$-measurable random variable ξ_n and the update rule for the state is given by*

$$X_{t_n}^{\boldsymbol{\theta}_n} = F_x(t_n, X_{t_n}^{\boldsymbol{\theta}_n}, \theta_n, \xi_n).$$

The aim is then to find $\boldsymbol{\theta}_N$, such that some cost functional is minimized.

Global optimization:

Here, the optimization is performed only once

$$\min_{\boldsymbol{\theta}_N} J(\boldsymbol{\theta}_N), \quad \textit{yielding} \quad \boldsymbol{\theta}_N^* = \arg\min_{\boldsymbol{\theta}_N} J(\boldsymbol{\theta}_N).$$

Most algorithms that resort to global optimization are of forward type.

Local optimization:

Here, the optimization is performed N times, one per time step t_n

$$\inf_{\theta_n} J_n(\theta_n) \quad \textit{yielding} \quad \boldsymbol{\theta}_N^* = \{ \arg\min_{\theta_0} J_0(\theta_0), \dots, \arg\min_{\theta_{N-1}} J_{N-1}(\theta_{N-1}) \}.$$

The algorithms with local optimization considered in this thesis are all of backward type, however, there are also many examples of forward algorithms with local optimization, e.g., greedy reinforcement learning methods.

Below, we introduce the three main types of problems that are considered in this thesis.

Decoupled state equation with the DPP satisfied:

Arguably, the most common problem in this category is valuation of options with an early-exercise feature, in particular, American and Bermudan options. These problems are highly suitable for dynamic programming, since the entire state process is decoupled and can therefore be sampled without any insight into the behaviour of the control process. This procedure breaks the problem down into approximations of appearing conditional expectations, one time point at the time. If the conditional expectation of future "cost" (or usually reward in the shape of future cash-flows) is higher than the cost achieved by stopping at the current time point, then it is optimal to stop. In this way, the problem is approximated, step by step, backwards in time, all the way back to $t = 0$. There are several ways to approximate these conditional expectations. A few alternatives are Monte-Carlo methods, Fourier-based methods or by a reformulation to a PDE or a reflected BSDE. The associate PDE or BSDE can, in turn, be approximated with finite difference/element methods or by Monte–Carlo or Fourier-based methods, respectively. More recently, researchers have turned their attention to machine learning-based methods to solve PDEs and (reflected) BSDEs. This is explained in more detail with several sources in Chapter 2 and 3.

Instead of the approximation of conditional expectations, the algorithm used in this thesis relies on a direct approximation of binary stopping decisions. We resort to the Deep Optimal Stopping algorithm (DOS), in which neural networks are used to learn the optimal stopping rule directly from Monte-Carlo samples of the underlying asset process.

Problems in this category are treated in Chapters 2 and 3.

Coupled state equation with the DPP satisfied:

Compared to the previous problem class, the seemingly minor change with a coupled state process actually completely changes the conditions for a well-functioning approximation strategy. A common approach to solving problems in this class is to formulate a *value function*, which, assuming that the infimum in (1.4) exists, is defined as

$$V(t, x) = J(\alpha^*; t, x).$$

If we assume that the state process is modelled by a controlled diffusion type SDE, with sufficiently nice drift- and diffusion-coefficients as well as sufficiently nice running- and terminal-cost functions in the cost functional. Then, the value function is a so-called *viscosity solution*

to a parabolic PDE. Moreover, the control of the system is a function of the gradient of V as well as of time and state. In turn, this implies that by solving the PDE, we have access to the optimal control. In practice, the PDE is solved on a space-time grid, *e.g.,* with a finite difference/element method. Before the equation can be approximated, we need to determine a domain and appropriate boundary conditions. Once the optimal control is available in the entire domain, paths of the asset process can be dynamically controlled based on some interpolation procedure between grid points. In some instances, the domain and boundary conditions are clearly prescribed by the problem but in many practically relevant examples the real domain is unbounded and the best we can do is to come up with an artificial domain and corresponding boundary conditions. This implies that we have to approximate the PDE solution on a grid which is large to guarantee that all the controlled paths stay within the domain. For low-dimensional problems, this is usually not a problem since we may have a rough idea of where in the space-time grid the controlled paths tend to end up and choose the domain large enough so that we are almost certain that all samples stay within. This is visualized for two different asset dynamics, one with low volatility and one with large volatility, in Figure 1.1. It is clear that the chosen domain is not large enough for the process with higher volatility (blue) since some of the paths leave the domain. In turn, this means that we do not have access to an optimal control on the entire $[0, T]$ for some of the paths. On the other hand, for the process with lower volatility (red), we have access to the control for all samples in $[0, T]$. Note that this is by no means equivalent to stating that the probability of the controlled state process with lower volatility will stay in the domain with probability 1.

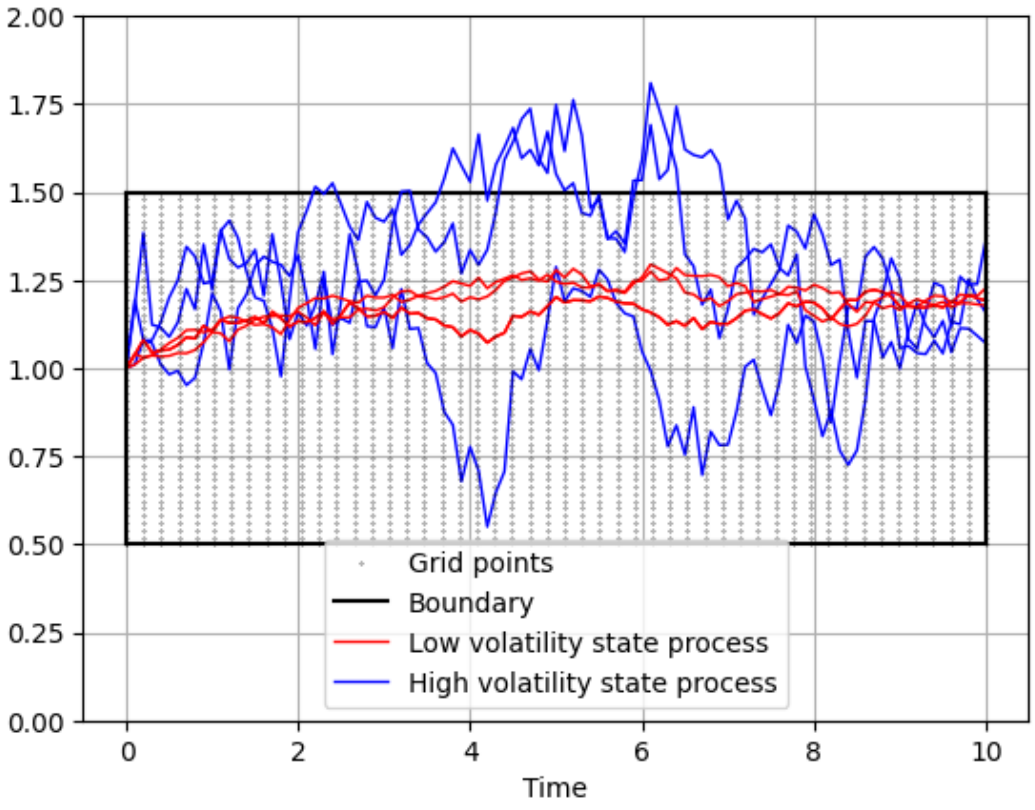

Figure 1.1: An example of a domain in which we have an optimal control and three representative sample paths of one controlled SDE, with low volatility (red) and high volatility (blue).

For higher-dimensional problems, and in particular when the dynamics are more complex than in the toy example from Figure 1.1, the task of choosing a domain can easily become intractable which makes the PDE approach infeasible (to approximate a PDE on a large domain in high dimensions is usually not possible with classical grid-based methods) from a practical perspective.

Another approach is to consider a grid-free approximation in the sense that the control is approximated on stochastic paths of the asset process. For $\omega \in \Omega$

$$\left(V\left(t, X_t^\alpha(\omega)\right)\right)_{t\in[0,T]}, \quad \text{and} \quad \left(\nabla_x V\left(t, X_t^\alpha(\omega)\right)\right)_{t\in[0,T]},$$

where the second term is the gradient of the value function, are required to obtain the optimal control. This would lower the computational burden since we would only have to approximate the control in the relevant domain. A natural way to achieve this is to reformulate the PDE (or directly the stochastic control problem) to a *forward backward stochastic differential equation* (FBSDE). More specifically, we would obtain a coupled FBSDE, meaning that the primary stochastic variables in the backward SDE (BSDE) impact the forward SDE and vice versa. To approximate decoupled FBSDEs, *i.e.,* when the forward SDE does not take the backward component as input, backward numerical schemes are natural. Assume that the terminal condition of the BSDE is of the form $Y_T = g(X_T)$, where X_T and Y_T are the terminal values of the forward SDE and BSDE, respectively. Then, it is clear that we have full access to the terminal value of the BSDE, if we are able to sample from X_T. In the decoupled case, a backward scheme can be employed, in which we start from Y_T, and recursively work our way backwards in time to obtain an approximation of $(Y_t)_{t\in[0,T]}$.

When there is a coupling, on the other hand, backward schemes run into problems because it is not clear in which spatial area the terminal value of the BSDE lies. The reason for this is that it is not possible to sample from X_T before we have access to a solution to the BSDE. Therefore, to approximate coupled FBSDEs is an inherently difficult task with classical and in particular backward methods.

In Chapter 4, a novel neural network-based method is proposed which combines the structure from both the original stochastic control problem and the associate FBSDE, to obtain an accurate approximation to a wide range of high-dimensional stochastic control problems. This method circumvents the problems faced by backward methods by considering an approximation forward in time and conducting the optimization globally.

Coupled state equation where the DPP is not satisfied:
The final class of problems considered in this thesis is based on cost functionals of the more general form (1.3), in which the cost is determined from a function which takes the probability law of the controlled state process as input. If one considers the state equation to be a controlled diffusion type SDE, possibly with a mean-field interaction (a McKean–Vlasov SDE), we are in the framework of mean-field control or control of McKean–Vlasov SDEs. Even though there are special cases that have been considered for many years, such as the mean-variance portfolio optimization problem, research into more general problems is relatively new with [8] as the corner stone reference works. One way to approach these kinds of problems is to use the stochastic maximum principle to reformulate a McKean–Vlasov type control problem

into a McKean–Vlasov type FBSDE. As the name suggests, this is an FBSDE in which the parameters take the probability law of the solutions to the forward SDE and the BSDE as inputs. A different approach to solving this problem is to transform it into a PDE, which is often referred to as the master equation of the McKean–Vlasov control problem. Both these approaches require restrictive assumptions on all parameters to guarantee existence of a solution and uniqueness requires an even more restrictive formulation. To approximately solve these equations is a highly nontrivial task for which most classical methods break down.

We therefore take on a strictly data-driven approach with the algorithms proposed in Chapter 5 in this thesis. We consider practical problems and take on a relaxed approach regarding mathematical technicalities such as existence and uniqueness of the sought solution. If the problem is to optimize the performance of an investment portfolio, we do not worry about whether or not an optimal strategy exists from a mathematical perspective. Also the uniqueness of such a strategy is of minor importance to an investor. As already pointed out, the algorithm is purely data-driven, and the probability law is approximated by its empirical counterpart. The control process is represented by a sequence of neural networks and the loss function of the neural networks is simply given by an empirical version of the cost functional. The algorithm operates forward in time and the optimization is carried out globally.

In Chapter 5, this algorithm is applied to a portfolio optimization problem in which allocation into a bond, stocks and options is allowed. To this, we consider a general form of objective function which aims to control the tail distribution of the terminal wealth. From a financial perspective, this could be seen as a way to mitigate rare events of large losses.

1.2 Mathematical finance

Financial mathematics encompasses various mathematical tools applied to understand and manage financial aspects. This thesis concentrates on three key domains: option valuation, risk management, and portfolio optimization, acknowledging their significance within the broader field of financial mathematics.

1. **Financial options:**
   This segment delves into determining the fair value of financial derivatives known as options. Mathematical models like the Black–Scholes model are utilized to estimate these values. However, it is essential to note that option valuation represents just one facet of financial mathematics.

2. **Risk Management:**
   Addressing uncertainties prevalent in financial markets, this section explores methodologies to measure and mitigate risks. Quantitative tools such as Value at Risk (VaR) and various mathematical approaches aid in managing risks. Nonetheless, this constitutes only a segment within the wider scope of risk management strategies. This area of financial mathematics has seen a drastic increase in attention from scientists and practitioners in the aftermath of the financial crisis in 2007-2008.

3. **Portfolio Optimization:**
   Focused on constructing investment portfolios, this thesis examines strategies for

optimizing returns while minimizing risks. It draws from Harry Markowitz's modern portfolio theory (MPT) to illustrate the balancing act between risk and return. However, this approach forms a part of the diverse landscape of portfolio optimization techniques within financial mathematics.

It is pivotal to recognize that financial mathematics extends beyond these specific areas, encompassing a rich array of mathematical methodologies and analytical tools employed to comprehend multifaceted financial phenomena. While this thesis delves deeply into these specific segments, it acknowledges the intricate and expansive nature of financial mathematics, which continually evolves and offers multifarious avenues for exploration and analysis.

By focusing on these specific domains, this thesis aims to contribute nuanced insights and in-depth analyses. Yet, it also underscores the vastness and continual evolution of financial mathematics, endeavoring to explore a few segments while acknowledging the broader expanse of this complex and dynamic field.

Below, these three concepts are presented in the context of this thesis.

1.2.1 Financial options

An option is a financial contract between two parties (holder and writer), which gives the *holder* the right (option), but not the obligation, to trade an asset in the future, according to predefined rules. The *writer*, on the other hand, is obliged to offer the holder to trade the asset, if such a trade is called for by the holder. There are many different types of options but in this chapter, we restrict the presentation to the contracts relevant for this thesis. The motivation to invest in an option can be to leverage a speculative position in the stock market, but more commonly it is meant to hedge an interest rate/FX risk.

The main mathematical problem with options is to find the fair value, which in short is given by the discounted expected future cash-flows of the contract. For instance, assuming the underlying asset is modeled by $(X_t)_{t\in[0,T]}$ and a contract pays $g(X_T)$, with a risk-free interest rate r, then the fair value at $t \in [0, T]$ is given by

$$\mathrm{e}^{-r(T-t)}\mathbb{E}\big[g(X_T)\,|X_t\big].$$

At first glance, the above valuation equation is straight forward and easy to understand, but one important question needs to be answered - how should the asset process be modelled and what conditions should such model satisfy? The discussion on which model to use for the asset process goes well beyond the scope of this thesis and we restrict our attention to Itô processes and, in some special cases, the more general Lévy processes.

A fundamental condition for the valuation model is that it should be free of *arbitrage*, meaning that it is not possible to have a positive probability of making a return higher than the risk-free rate without taking on any risk (zero probability of a return below the risk-free rate). In the world of option pricing, we assume that the entire market is free of arbitrage, implying that it is not possible to create a portfolio with arbitrage, or simply, that there are no arbitrage opportunities in the market. In practice, it is well known that opportunities of arbitrage exist but the assumption is that these only exist for a very limited time period (until someone takes advantage of the opportunity), implying that the arbitrage disappears.

Despite this, the assumption of an arbitrage-free market is central for mathematical finance in general and for option valuation in particular. From a mathematical perspective, this means that discounted *tradable assets* are martingales and the probability measure, under which this holds, is referred to as the *risk-neutral measure* and denoted by $\mathbb{Q}$. The expectation under the risk-neutral measure is then denoted by $\mathbb{E}_{\mathbb{Q}}$ and the fair value of an option, terminating at T, with a pay-off function g (assuming deterministic interest rates) is given by

$$V(t,x) = \mathrm{e}^{-r(T-t)}\mathbb{E}_{\mathbb{Q}}\big[g(X_T)\,|X_t = x\big].$$

In this thesis, we are particularly interested in options with early-exercise features

$$V(t,x) = \sup_{\tau\in\mathcal{T}_t} \mathbb{E}_{\mathbb{Q}}\big[\mathrm{e}^{-r(\tau-t)}g(X_\tau)\,|X_t = x\big], \tag{1.8}$$

where $\mathcal{T}_t$ is the set of all $X-$stopping times greater than or equal to t. The latter valuation problem is in fact a stochastic control problem and falls into the category of *decoupled state equation, with the DPP satisfied*, as described in the previous section.

1.2.2 Risk management

The attention for risk management in mathematical finance has increased significantly since the global financial crisis in 2007-2008. The third Basel accord, with the title *Basel III: A global regulatory framework for more resilient banks and banking systems*, but usually referred to as just *Basel III*, was in large motivated by this crisis. Both the financial crisis and the Basel accords are broad topics which both go way beyond the scope of this thesis and we limit our motivation to the so-called option *valuation adjustments* (xVA). The "x" in xVA, represents a specific risk factor, for instance, *credit valuation adjustment* (CVA), is the valuation adjustment one should make for the risk of a possible default of the counterparty.

Historically, the valuation of an option has been performed independently of the counterparty, *i.e.,* without considering the credit quality of the counterparty. Clearly, there is a difference between having a financial exposure against a stable government versus a highly leveraged real estate company. Despite the fact that this was a concern already before the 2008 financial crisis, especially among researchers, it is in the recent 15 years that this has received full attention from practitioners.

As an illustrative example, the CVA is computed as the difference between the *risk-free value* and the *risky value*. The risk-free value is the classical value of the option as described above in, for instance, (1.8), *i.e.,* the discounted expected future cash-flow. The corresponding risky value also takes the probability of default, as well as the *recovery rate* (estimated return in the event of a default of the counterparty), into account. The risky value of the option could be given by

$$V^{\text{risky}}(t,x) = \mathbb{E}_{\mathbb{Q}}\Big[\big(1 - \text{LGD}\times\mathbb{I}_{\tau^{\text{D}}<\tau^*}(\tau^*)\big)\times\mathrm{e}^{-r(\tau-t)}g(\tau^*, X_{\tau^*})\,|X_t = x\Big], \tag{1.9}$$

where τ^* is the optimal exercise strategy for the risk-free problem, τ^{D} is the time for the default event ($\tau^{\text{D}} > T$ implies that no default takes place before time T) and LGD is the loss given default which takes on values in $[0,1)$. The expression for the risk-free value could be

made more complicated by also taking into account that the optimal exercise strategy may change due to the credit quality of the counterparty. On the other hand, it could be made less complicated by the simplification of considering the LGD, τ^{D} and X as independent entities. Finally, if the risk-free and risky values are given as above, then the corresponding CVA is given by

$$\mathrm{CVA}(t,x) = V(t,x) - V^{\mathrm{risky}}(t,x). \tag{1.10}$$

In most computations of the CVA, as well as many of the other xVAs, the *financial exposure*, or simply exposure, is of central importance. The exposure is given by the positive part of the outstanding value of an option. From a risk management point of view, it is reasonable to only consider the positive part of the value since the negative will remain even in the event of the counterparts default. However, computing the exposure of an option is nothing but computing values and then truncating at zero. The difference from the valuation in Section 1.2.1 is that while classical option valuation requires the value at one point in time and space, to compute the exposure requires the option value along stochastic trajectories of the underlying asset. This becomes challenging for high-dimensional problems, *e.g.,* for basket options, especially in the presence of early-exercise features.

1.2.3 Portfolio optimization

In the world of *portfolio optimization*, contrary to option valuation, the market is not considered to be free of arbitrage. If this would be the case, then the most sensible trading decision would be to invest only in bonds bearing the risk-free interest rate, since that strategy would minimize the risk. In portfolio optimization problems, the *real-world measure*, denoted by $\mathbb{P}$ is used. This causes the delicate question how one should model a problem in which the portfolio consists of a bond (deterministic), stocks (modeled under $\mathbb{P}$) and options which should be valued under $\mathbb{Q}$. The approach followed in this thesis is to follow the convention and perform option valuation under $\mathbb{Q}$ and then let the performance be measured under $\mathbb{P}$. In fact, this is also consistent with the current value of the stocks. This can be seen by constructing a dummy option, written on the asset X and terminating at T, with a pay-off function $g(X_T) = X_T$. The fair value of this option at t is given by

$$V(t,X_t) = \mathrm{e}^{-rT}\mathbb{E}_{\mathbb{Q}}[X_T|X_t] = X_t,$$

which is clearly consistent with the value at t of the asset X.

As mentioned, we consider portfolio optimization problems where the investor is allowed to trade a bond, stocks and options written on the same stocks. The motivation for this, somewhat unusual, portfolio decomposition is that the terminal wealth distribution is very different when comparing a stock and an option written on this stock. This implies that in a portfolio which allows for trading also in options, the flexibility in controlling the distribution of the portfolio's terminal wealth is considerably higher. Another advantage is that many funds are prohibited from using leverage in their portfolios and trading in options could be seen as a way to achieve a leveraged position without the downside risk by paying the option premium.

With the increased flexibility in shaping the portfolio's terminal wealth distribution, it is reasonable to consider objective functions beyond the classical expected utility and mean-

variance functions. Expected utility functions are usually constructed so that the DPP holds. We could then either end up in the first (decoupled state equation) or second category (coupled) from the section above, depending on how the problem is formulated. The mean-variance asset allocation problem, on the other hand, leads to a problem of the third category, since the DPP is not satisfied due to the non-linearity in the conditional expectations in the variance term. However, for many standard formulations of the mean-variance problem, there are closed-form solutions to the optimization problem, which makes this a popular formulation. Another advantage of this formulation is the clear intuitive formulation of one reward term (expected terminal wealth), which we want to maximize, and one risk term (variance of terminal wealth), which we want to minimize. In addition, if the distribution of the terminal wealth could be well approximated with a normal distribution, then the mean-variance objective is optimal since the normal distribution is completely described by its first two moments. In a situation with only stocks, the normal distribution is a reasonable, although not perfect, approximation of the terminal wealth distribution. With options added to the portfolio, this does no longer hold true and another objective function should be used. To connect this problem to stochastic control, we are in the framework of control of mean-field type. In general, it is possible to use the stochastic maximum principle to transform this problem into a mean-field FBSDE of the appearing PDE. The problem is then that with such a complicated formulation with a non-standard objective function for a portfolio consisting of a bond, stocks and options, it is not even clear whether or not the solution (optimal allocation strategy) is unique and therefore the properties of the transformed problem are challenging to control. With a setting like this, most classical optimization methods break down and one has to rely on, for example, a data-driven approach.

1.3 Structure of the thesis

Chapters 2-3, which are based on [1] and [2], treat pricing of high-dimensional Bermudan options. The valuation problems belong to the category of decoupled state equation with the DPP satisfied and a neural network-based approximation algorithm is proposed. Moreover, the financial exposures are computed for a single option and for a portfolio of options in Chapters 2 and 3, respectively. This is also done with a neural network-based regression algorithm which approximates conditional expectations.

In Chapter 4, a neural network-based algorithm for approximating FBSDEs stemming from stochastic control is proposed, which is based on [3]. The algorithm uses both the stochastic control formulation and the FBSDE formulation in the loss function. This chapter treats problems from the category in which the DPP is satisfied with a coupled state process.

Finally, Chapter 5 treats stochastic control problems where the DPP does not hold with a coupled state process. In particular, the focus is on time-inconsistent portfolio optimization problems, which are approximated with neural networks in a data-driven fashion. The chapter is based on [5].

The notation used in this thesis may differ between chapters. Therefore, in each chapter, the current notation is clearly defined. The reason for this is that we want to, as far as possible, stick to standard notation in each area of mathematical finance. Due to the broad scope of

the thesis, this would cause notational conflicts if we would have used a coherent notation throughout the entire thesis.

2

A deep learning approach for computations of exposure profiles for high-dimensional Bermudan options

In this chapter, we propose a neural network-based method for approximating expected exposures and potential future exposures of Bermudan options. In a first phase, the method relies on the Deep Optimal Stopping algorithm (DOS) proposed in [9], which learns the optimal stopping rule from Monte-Carlo samples of the underlying risk factors. cash-flow paths are then created by applying the learned stopping strategy on a new set of realizations of the risk factors. Furthermore, in a second phase the cash-flow paths are projected onto the risk factors to obtain approximations of pathwise option values. The regression step is carried out by ordinary least squares as well as neural networks, and it is shown that the latter produces more accurate approximations.

The expected exposure is formulated, both in terms of the cash-flow paths and in terms of the pathwise option values and it is shown that a simple Monte-Carlo average yields accurate approximations in both cases. The potential future exposure is estimated by the empirical α-percentile.

Finally, it is shown that the expected exposures, as well as the potential future exposures can be computed under either, the risk-neutral measure, or the real-world measure, without having to re-train the neural networks.

Keywords *- Bermudan options, risk management, financial exposure, neural networks*

This chapter is based on the paper with the same title, which is published in Applied Mathematics and Computation 408 (2021): 126332

2.1 Introduction

The exposure of a financial contract is the maximum amount that an investor stands to lose if the counterparty is unable to fulfill its obligations, for instance, due to a default. This means that, in addition to the market risk, a so-called counterparty credit risk (CCR) needs to be accounted for. Furthermore, the liquidity risk, which is the risk arising from potential costs of unwinding a position, is also closely related to the financial exposure. Over the counter (OTC) derivatives, *i.e.,* contracts written directly between counterparties, instead of through a central clearing party (CCP), are today mainly subject to so-called valuation adjustments (XVA[1]). These valuation adjustments aim to adjust the value of an OTC derivative for certain risk factors, *e.g.,* credit valuation adjustment (CVA), adjusting the value for CCR, funding valuation adjustment (FVA), adjusting for funding cost of an uncollateralized derivative or capital valuation adjustment (KVA), adjusting for future capital costs. The financial exposure is central in calculations of many of the XVAs (for an in-depth overview of XVAs, we refer to [10] and [11]). In this chapter, we therefore focus on computations of financial exposures of options.

For a European style option, the exposure is simply the option value at some future time, given all the financial information available today. If t_0 is the time today and X_t the d-dimensional vector of underlying risk factors of an option, we define the exposure of the option, at time $t > t_0$, as the random variable[2]

$$\mathrm{E}_t^{\mathrm{Eur}} = V_t(X_t).$$

However, for derivatives with early-exercise features, we also need to take into account the possibility that the option has been exercised prior to t, sending the exposure to zero. The most well-known of such options is arguably the American option, which gives the holder (or buyer) the right to exercise the option at any time between t_0 and maturity T. In this chapter, the focus is on the Bermudan option which gives the holder the right to exercise at finitely many, predefined exercise dates between t_0 and maturity T. For early-exercise options, the exposure needs to be adjusted for the possibility that the stopping time τ, representing the instance of time at which the option is exercised, is smaller than or equal to t. This leads to the following definition of the exposure

$$\mathrm{E}_t^{\mathrm{Ber}} = V_t(X_t)\mathbb{I}_{\{\tau > t\}},$$

where $\mathbb{I}_{\{\cdot\}}$ is the indicator function. Two of the most common ways to measure the exposure are the expected exposure (EE), which is the expected future value of the exposure, and the potential future exposure (PFE), which is some $\alpha-$percentile of the future exposure (usually $\alpha = 97.5\%$ for the upper, and $\alpha = 2.5\%$ for the lower tails of the distribution). To accurately compute EE and PFE of a Bermudan option, it is crucial to use an algorithm which is able to accurately approximate, not only $V_t(X_t)$, but also $\mathbb{I}_{\{\tau > t\}}$.

[1] X represents arbitrary letters, *e.g.,* "C" for credit valuation adjustment, "F" for funding valuation adjustment, etc.

[2] Usually, the exposure is defined as $\max\{V_t(X_t), 0\}$, but since we only consider options with non-negative values, the max operator is omitted.

It is common to compute the EE and the PFE with simulation-based methods, usually from realizations of approximate exposures, from which the distribution of E_t is approximated by its empirical counterpart. A generic scheme for approximation of exposure profiles is given below:

1. At each exercise date, find a representation, $v_t : \mathbb{R}^d \to \mathbb{R}$, of the true value function $V_t(\cdot)$;
2. Generate trajectories of the underlying risk factors, and at each exercise date, evaluate[3] $v_t(X_t(\omega))$;
3. At the earliest exercise date where $v_t(X_t(\omega)) \leq g(X_t(\omega))$, where g is the pay-off function, set $\tau(\omega) = t$;
4. The distribution of $\mathrm{E}_t^{\mathrm{Ber}}$, is then approximated by the empirical distribution created by many trajectories of the form $v_t(X_t(\omega))\mathbb{I}_{\{\tau(\omega)>t\}}$. For instance, if the target statistic is the EE, the estimation is the sample mean of the exposure paths.

Step 1 above corresponds to the option valuation problem, which can be tackled by several different methods, all with their own advantages and disadvantages. For instance, the value function can be approximated by solving an associated partial differential equation (PDE), which is done in *e.g.,* [12],[13],[14],[15] and [16], or the value function can be approximated by a Fourier transform methodology, which is done in e.g., [17], [18] and [19]. Furthermore, classical tree-based methods such as [20], [21] and [22], can be used. These types of methods are, in general, highly accurate but they suffer severely from the curse of dimensionality, meaning that they are computationally feasible only in low dimensions (say up to $d = 4$), see [23]. In higher dimensions, Monte-Carlo-based methods are often used, see *e.g.,* [24],[25],[26],[27] and [28]. Monte-Carlo-based methods can generate highly accurate option values at t_0, *i.e.,* $v_{t_0}(X_{t_0} = x_{t_0})$, given that all trajectories of the underlying risk factors are initiated at some deterministic x_{t_0}. However, at intermediate exercise dates, $v_t(X_t(\omega))$ might not be an equally good approximation, due to the need of cross sectional regression. We show with numerical examples that, even though the approximation is good on average, the option value is underestimated in certain regions, which is compensated by overestimated values in other regions. For European options, this seems to have minor effect on EE and PFE, but for Bermudan options the effect can be significant due to step 3 above. To provide an intuitive understanding of the problem, we give an illustrative example. Assume that v_t underestimates the option value in some region A, which is close to the exercise boundary, and overestimates the option value in some region B, where it is clearly optimal to exercise the option. The effect on the exposure would be an overestimation in region A, since underestimated option values would lead to fewer exercised options. In region B, the exposure would be zero in both cases since all options in that region would be exercised immediately. In total, this would lead to overestimated exposure. In numerical examples this is, of course, more involved and we see typically several regions with different levels of under/overestimated values. This makes the phenomenon difficult to analyze since the effect may lead to underestimated exposure, unchanged exposure (from offsetting effects), or overestimated exposure. In addition to the

[3]We define a filtered probability space in the next section from which ω is an element. In this section, ω should be viewed as an outcome of a random experiment.

classical regression methods, with *e.g.,* polynomial basis functions, which are cited above, there are several papers in which a neural network plays the role of the basis functions, see *e.g.,* [29],[30], [31] and [32].

In this chapter we suggest the use of the Deep Optimal Stopping (DOS) algorithm, proposed in [9], to approximate the optimal stopping strategy. The DOS algorithm approximates the optimal stopping strategy by expressing the stopping time in terms of binary decision functions, which are approximated by deep neural networks. The DOS algorithm is very suitable for exposure computations, since the stopping strategy is computed directly, not as a consequence of the approximate value function, as is the case with most other algorithms. This procedure leads to highly accurate approximations of the exercise boundaries. Furthermore, we propose a neural network-based regression method (NN-regression) to approximate $V_t(\cdot)$, which relies on the stopping strategy, approximated by the DOS algorithm. Although NN-regression is needed in order to fully approximate the distributions of future exposures, it turns out that for some target statistics, it is in fact sufficient to approximate the optimal stopping strategy. This is, for instance, the case for some estimates of the EE.

Another advantage of the method is that the EE and the PFE easily can be computed under the risk-neutral measure, as well as the real-world measure. When the EE is used to compute CVA, the calculations should be done under the risk-neutral measure, since CVA is a tradable asset and any other measure[4] would create opportunities of arbitrage. For other applications, such as for computations of KVA (valuation adjustment to account for future capital costs associated to a derivative), the EE should be calculated under the real-world measure[5]. The reason for this is that the KVA, among other things, depends on the required CCR capital which is a function of the EE, which in this case, ideally, should be calculated under the real-world measure. For an explanation of KVA in general, see *e.g.,* [33] and for a discussion about the effect of different measures in KVA computations see [34].

Finally, we emphasize that, even though the focus of this chapter is to compute exposure profiles, the algorithms proposed are flexible and with small adjustments other kind of risk measures could be computed. The first reason for only studying exposure profiles is, as mentioned above, that it is an important building block in computations (or approximations) of many XVAs as well as expected shortfall. Another reason is that there are a number of similar studies in the existing literature (see *e.g.,* [35], [36], [37]). However, it should be pointed out that the algorithms proposed in this chapter are not limited to computations of exposures. As an example, consider the CVA of a Bermudan option, expressed as the difference between the risky (including CCR) and the risk-free (excluding CCR) option values. To accurately compute the risky option value, one needs to adjust the exercise strategy for the risk of a default of the counterparty. Therefore, it is not sufficient to compute the exposure of the option when computing the CVA (for a discussion see *e.g.,* [38], [39]). With minor adjustments to the proposed algorithm, one could compute this kind of advanced CVA (as opposed to the approximation, which is based on the exposure).

This chapter is structured as follows: In Section 2.2, the mathematical formulation of a Bermudan option and its exposure profiles are given. Furthermore, the Bermudan option, as

[4]For fixed numéraire.

[5]In this setting, the EE is an expectation under the real-world measure, which is conditioned on a filtration generated by the underlying risk factors.

well as the exposure profiles are reformulated to fit the algorithms introduced in later sections. The DOS algorithm is described in Section 2.3, and we propose some adjustments to make the training procedure more efficient. In Section 2.4, we present a classical ordinary least squares regression-based method (OLS-regression), as well as the NN-regression to approximate pathwise option values. In Section 2.5, the EE and PFE formulas are reformulated in a way such that they can easily be estimated by a combination of the algorithms presented in Sections 2.3 and 2.4, and a simple Monte-Carlo sampling. Finally, in Section 2.6, numerical results of the different algorithms, are presented for Bermudan options following the Black–Scholes dynamics as well as the Heston model dynamics.

2.2 Problem formulation

For $d, N \in \mathbb{N} \setminus \{0\}$, let $X = (X_{t_n})_{n=0}^{N}$ be an $\mathbb{R}^d-$valued discrete-time Markov process on a complete probability space $(\Omega, \mathcal{F}, \mathbb{A})$. The outcome set Ω is the set of all possible realizations of the stochastic economy, $\mathcal{F}$ is a $\sigma-$algebra on Ω and we define $\mathcal{F}_n$ as the sub-σ-algebra generated by $(X_{t_m})_{m=0}^{n}$. With little loss of generality, we restrict ourselves to the case when X is constructed from time snap shots of a continuous-time Markov process at monitoring dates $\{t_0, t_1, \ldots, t_N\}$. The probability measure $\mathbb{A}$ is a generic notation, representing either the real-world measure, or the risk-neutral measure, denoted $\mathbb{P}$ and $\mathbb{Q}$, respectively.

If not specifically stated otherwise, equalities and inequalities of random variables should be interpreted in an ω-wise sense.

2.2.1 Bermudan options, stopping decisions and exercise regions

A Bermudan option is an exotic derivative that gives the holder the opportunity to exercise the option at a finite number of exercise dates, typically one per month. We define the exercise dates as $\mathbb{T} = \{t_0, t_1, \ldots, t_N\}$, which for simplicity coincide with the monitoring dates. Furthermore, for $t_n \in \mathbb{T}$, we let the remaining exercise dates be defined as $\mathbb{T}_n = \{t_n, t_{n+1} \ldots, t_N\}$. Let τ be an $X-$stopping time, *i.e.*, a random variable defined on $(\Omega, \mathcal{F}, \mathbb{A})$, taking on values in $\mathbb{T}$ such that for all $t_n \in \mathbb{T}$, it holds that the event $\{\tau = t_n\} \in \mathcal{F}_n$. Assume a risk-free rate $r \in \mathbb{R}$ and let the risk-less savings account process, $M(t) = \mathrm{e}^{r(t-t_0)}$, be our numéraire. For all $t \in \mathbb{T}$, let[6] $g_t : \mathbb{R}^d \to \mathbb{R}$ be a measurable function which returns the immediate pay-off of the option, if exercised at market state $(t, X_t = x \in \mathbb{R}^d)$. The initial value of a Bermudan option, *i.e.,* the value at market state $(t_0, X_{t_0} = x_{t_0} \in \mathbb{R}^d)$, is given by

$$V_{t_0}(x_{t_0}) = \sup_{\tau \in \mathcal{T}} \mathbb{E}_{\mathbb{Q}} \left[\mathrm{e}^{-r(\tau - t_0)} g_\tau(X_\tau) \,|\, X_{t_0} = x_{t_0} \right], \tag{2.1}$$

where $\mathcal{T}$ is the set of all $X-$stopping times. Assuming the option has not been exercised prior to some $t_n \in \mathbb{T}$, the option value at t_n is given by

$$V_{t_n}(X_{t_n}) = \sup_{\tau \in \mathcal{T}_n} \mathbb{E}^n_{\mathbb{Q}} \left[\mathrm{e}^{-r(\tau - t_n)} g_\tau(X_\tau) \right], \tag{2.2}$$

[6] Allowing for different pay-off functions at different exercise dates makes the framework more flexible. We can *e.g.*, let g_t be the pay-off function for an entire netting set of derivatives with different maturities.

where $\mathcal{T}_n$ is the set of all $X-$stopping times greater than or equal to t_n and we define $\mathbb{E}^n_{\mathbb{A}}[\,\cdot\,] = \mathbb{E}_{\mathbb{A}}[\,\cdot\,|\mathcal{F}_n]$. To guarantee that (2.1) and (2.2) are well-posed, we assume that for all $t \in \mathbb{T}$ it holds that

$$\mathbb{E}_{\mathbb{Q}}\left[|g_t(X_t)|\right] < \infty.$$

To concretize (2.1) and (2.2), we view the problem from the holder's perspective. At each exercise date, $t_n \in \mathbb{T}$, the holder of the option is facing the decision whether or not to exercise the option, and receive the immediate pay-off. Due to the Markovian nature of X, the decisions are of Markov-type (Markov decision process (MDP)), meaning that all the information needed to make an optimal decision[7] is contained in the current market state, $(t_n,\, X_{t_n})$. With this in mind, we define for each $t_n \in \mathbb{T}$, the exercise region, $\mathcal{E}_n$, in which it is optimal to exercise, and the continuation region, $\mathcal{C}_n$, in which it is optimal to hold on, by

$$\mathcal{E}_n = \left\{x \in \mathbb{R}^d \,|\, V_{t_n}(x) = g_{t_n}(x)\right\}, \quad \mathcal{C}_n = \left\{x \in \mathbb{R}^d \,|\, V_{t_n}(x) > g_{t_n}(x)\right\}.$$

Note that $\mathcal{E}_n \cup \mathcal{C}_n = \mathbb{R}^d$ and $\mathcal{E}_n \cap \mathcal{C}_n = \emptyset$.

Below, we give a short motivation for how these decisions can be expressed mathematically and how we can formulate a stopping time in terms of (stopping) decisions at each exercise date. For a more detailed motivation we refer to [9]. We introduce the following notation for measurable functions[8]

$$\mathcal{D}(A_1; A_2) = \{\, f \colon A_1 \to A_2 \,|\, f \text{ measurable}\}.$$

Furthermore, for $P \in \mathbb{N}$, let $\mathcal{D}(A_1; A_2)^P$ denote the P :th Cartesian product of the set $\mathcal{D}(A_1; A_2)$. Define the decision functions $f_0,\, f_1,\, \ldots,\, f_N \in \mathcal{D}(\mathbb{R}^d;\, \{0,1\})$, with $f_N \equiv 1$, and denote $\boldsymbol{f}_n = (f_n,\, f_{n+1},\, \ldots,\, f_N)$ with $\boldsymbol{f} = \boldsymbol{f}_0$. An $X-$stopping time can then be defined as

$$\tau_n(\boldsymbol{f}_n) = \sum_{m=n}^{N} t_m f_m(X_{t_m}) \prod_{j=n}^{m-1} (1 - f_j(X_{t_j})), \tag{2.3}$$

where the empty product is defined as 1. We emphasize that
$\tau_n(\boldsymbol{f}_n) = \tau_n\left(f_n(X_{t_n}),\, f_{n+1}(X_{t_{n+1}}) \ldots,\, f_N(X_{t_N})\right)$ but to make the notation cleaner, we do not specify this dependency explicitly. When it is not clear from the context which process the decision function $\boldsymbol{f}_n$ is acting on, we will denote the stopping time by $\tau_n\left(\boldsymbol{f}_n(X)\right)$, where we recall that $X = (X_{t_m})_{m=n}^N$. As a candidate for optimal stopping decisions, we define

$$\boldsymbol{f}^* \in \underset{\boldsymbol{f} \in \mathcal{D}(\mathbb{R}^d;\, \{0,1\})^{N+1}}{\arg\max}\; \mathbb{E}_{\mathbb{Q}}\left[\mathrm{e}^{-r(\tau(\boldsymbol{f}) - t_0)} g_{\tau(\boldsymbol{f})}\left(X_{\tau(\boldsymbol{f})}\right) \,|\, X_{t_0} = x_0\right]. \tag{2.4}$$

We define the fair price of an option that follows the strategy constructed by combining (2.3) and (2.4) as

$$V^*_{t_n}(X_{t_n}) = \mathbb{E}^n_{\mathbb{Q}}\left[\mathrm{e}^{-r(\tau_n(\boldsymbol{f}^*_n) - t_n)} g_{\tau_n(\boldsymbol{f}^*_n)}\left(X_{\tau_n(\boldsymbol{f}^*_n)}\right)\right]. \tag{2.5}$$

[7] In the sense of using a decision policy such that the supremum in (2.1) or (2.2) is attained.

[8] We assume measurable spaces $(A_1, \mathcal{A}_1)$ and $(A_2, \mathcal{A}_2)$ and measurable functions with respect to σ-algebras $\mathcal{A}_1$ and $\mathcal{A}_2$. This assumption holds in all cases in this chapter.

The following theorem states that (2.5), in fact, coincides with the fair price of the option as defined in (2.2).

Theorem 1. *For all $t_n \in \mathbb{T}$, and V and V^* as defined in* (2.2) *and* (2.5)*, respectively, it holds that*[9]

$$V_{t_n}(X_{t_n}) = V^*_{t_n}(X_{t_n}).$$

Proof. Note that $V_{t_N}(X_{t_N}) = V^*_{t_N}(X_{t_N}) = g_{t_N}(X_{t_N})$. The proof is carried out by induction and we assume that for some $t_{n+1} \in \mathbb{T}$ it holds that

$$V_{t_{n+1}}(X_{t_{n+1}}) = V^*_{t_{n+1}}(X_{t_{n+1}}). \tag{2.6}$$

We can rewrite $V^*_{t_n}(X_{t_n})$ as

$$\begin{aligned} V^*_{t_n}(X_{t_n}) =& \mathbb{E}^n_{\mathbb{Q}}\left[\mathrm{e}^{-r(\tau_n(\boldsymbol{f}^*_n)-t_n)} g_{\tau_n(\boldsymbol{f}^*_n)}(X_{\tau_n(\boldsymbol{f}^*_n)})\right] \\ =& g_{t_n}(X_{t_n}) f^*_n(X_{t_n}) + (1 - f^*_n(X_{t_n}))\mathrm{e}^{-r(t_{n+1}-t_n)} \\ & \times \mathbb{E}^n_{\mathbb{Q}}\left[\mathrm{e}^{-r(\tau_{n+1}(\boldsymbol{f}^*_{n+1})-t_{n+1})} g_{\tau_{n+1}(\boldsymbol{f}^*_{n+1})}(X_{\tau_{n+1}(\boldsymbol{f}^*_{n+1})})\right]. \end{aligned} \tag{2.7}$$

By the law of total expectation and the assumption (2.6), the last conditional expectation satisfies

$$\mathbb{E}^n_{\mathbb{Q}}\left[\mathrm{e}^{-r(\tau_{n+1}(\boldsymbol{f}^*_{n+1})-t_{n+1})} g_{\tau_{n+1}(\boldsymbol{f}^*_{n+1})}(X_{\tau_{n+1}(\boldsymbol{f}^*_{n+1})})\right] = \mathbb{E}^n_{\mathbb{Q}}\left[V^*_{t_{n+1}}(X_{t_{n+1}})\right] = \mathbb{E}^n_{\mathbb{Q}}\left[V_{t_{n+1}}(X_{t_{n+1}})\right]. \tag{2.8}$$

We insert the rightmost part of (2.8) in (2.7) and note that $\mathbb{I}_{\{\,\cdot\,\in\mathcal{E}_n\}} \in \mathcal{D}(\mathbb{R}^d; \{0,1\})$, which implies that

$$\begin{aligned} V^*_{t_n}(X_{t_n}) \geq& \mathbb{I}_{\{X_{t_n}\in\mathcal{E}_n\}} g_{t_n}(X_{t_n}) + \mathbb{I}_{\{X_{t_n}\in\mathcal{C}_n\}} \mathrm{e}^{-r(t_{n+1}-t_n)} \mathbb{E}^n_{\mathbb{Q}}\left[V_{t_{n+1}}(X_{t_{n+1}})\right] \\ =& V_{t_n}(X_{t_n}). \end{aligned}$$

Moreover, $V^*_{t_n}(X_{t_n}) \leq V_{t_n}(X_{t_n})$ and therefore we conclude that $V^*_{t_n}(X_{t_n}) = V_{t_n}(X_{t_n})$. □

Although not completely straight-forward, the result above can actually be inferred from [9, Theorem 1].

2.2.2 Exposure profiles

For $t_n \in \mathbb{T}$ and $\alpha \in (0, 1)$, we define the expected exposure (EE) and the potential future exposure (PFE) under the generic probability measure $\mathbb{A}$ as

$$\mathrm{EE}_{\mathbb{A}}(t_n) = \mathbb{E}_{\mathbb{A}}\left[V_{t_n}(X_{t_n})\mathbb{I}_{\{\tau(\boldsymbol{f}^*)>t_n\}} \,|\, X_{t_0} = x_{t_0}\right], \tag{2.9}$$

$$\mathrm{PFE}^{\alpha}_{\mathbb{A}}(t_n) = \inf\left\{a \in \mathbb{R} \,|\, \mathbb{A}\left(V_{t_n}(X_{t_n})\mathbb{I}_{\{\tau(\boldsymbol{f}^*)>t_n\}} \leq a \,|\, X_{t_0} = x_{t_0}\right) \geq \alpha\right\}. \tag{2.10}$$

Note that the option value, given by (2.2), is a conditional expectation of future cash-flows, which is by definition measured with $\mathbb{Q}$. The exposure profiles under the $\mathbb{P}-$measure, on the

[9] We recall that equalities and inequalities are in an $\mathbb{A}-$almost sure sense.

other hand, are statistics of the option value at some future time, t_n, under the assumption that the conditional distribution X_{t_m} conditional on $X_{t_0} = x_{t_0}$ is considered under the $\mathbb{P}-$measure.

In the theorem below, the expected exposures are reformulated in terms of discounted cash-flows and the decision functions introduced in Subsection 2.2.1. It will become clear in later sections that this is a more tractable form for the algorithms used in this chapter.

Theorem 2. *Let $\mathbb{Q}$ and $\mathbb{P}$ be probability measures on the measurable space $(\Omega, \mathcal{F})$ and assume that the laws of X under $\mathbb{Q}$ and $\mathbb{P}$ are absolutely continuous with respect to the Lebesgue measure. Then, the expected exposure, (2.9) under the $\mathbb{Q}-$ and $\mathbb{P}-$measures, satisfies*

$$\mathrm{EE}_{\mathbb{Q}}(t_n) = \mathbb{E}_{\mathbb{Q}}\left[e^{-r(\tau_n(\boldsymbol{f}_n^*)-t_n)} g_{\tau_n(\boldsymbol{f}_n^*)}(X_{\tau_n(\boldsymbol{f}_n^*)}) \mathbb{I}_{\{\tau(\boldsymbol{f}^*)>t_n\}} \,|\, X_{t_0} = x_{t_0}\right], \tag{2.11}$$

$$\mathrm{EE}_{\mathbb{P}}(t_n) = \mathbb{E}_{\mathbb{Q}}\left[e^{-r(\tau_n(\boldsymbol{f}_n^*)-t_n)} g_{\tau_n(\boldsymbol{f}_n^*)}(X_{\tau_n(\boldsymbol{f}_n^*)}) \mathbb{I}_{\{\tau(\boldsymbol{f}^*)>t_n\}} l(X_{t_n}, X_{t_{n-1}}, \ldots, X_{t_0}) \,|\, X_{t_0} = x_{t_0}\right], \tag{2.12}$$

where $l(y_n, y_{n-1}, \ldots, y_0) = \prod_{k=1}^{n} \frac{p_{X_{t_k}|X_{t_{k-1}}}(y_k|y_{k-1})}{q_{X_{t_k}|X_{t_{k-1}}}(y_k|y_{k-1})}$, with $q_{X_{t_k}|X_{t_{k-1}}}(y_k|y_{k-1})$ and $p_{X_{t_k}|X_{t_{k-1}}}(y_k|y_{k-1})$ being transition densities for X under the measures $\mathbb{Q}$ and $\mathbb{P}$, respectively. Note that $l(y_n, y_{n-1}, \ldots, y_0)$ is the Radon–Nikodym derivative of $\mathbb{P}$ with respect to $\mathbb{Q}$ evaluated at $(y_n, y_{n-1}, \ldots, y_0)$.

Proof. We begin by proving (2.11). By combining (2.5) and (2.9) and setting $\mathbb{A} = \mathbb{Q}$ we obtain

$$\begin{aligned}
\mathrm{EE}_{\mathbb{Q}}(t_n) &= \mathbb{E}_{\mathbb{Q}}\left[\mathbb{E}^n_{\mathbb{Q}}[\mathrm{e}^{-r(\tau_n(\boldsymbol{f}_n^*)-t_n)} g_{\tau_n(\boldsymbol{f}_n^*)}(X_{\tau_n(\boldsymbol{f}_n^*)})] \mathbb{I}_{\{\tau(\boldsymbol{f}^*)>t_n\}} \,|\, X_{t_0} = x_{t_0}\right] \\
&= \mathbb{E}_{\mathbb{Q}}\left[\mathrm{e}^{-r(\tau_n(\boldsymbol{f}_n^*)-t_n)} g_{\tau_n(\boldsymbol{f}_n^*)}(X_{\tau_n(\boldsymbol{f}_n^*)}) \mathbb{I}_{\{\tau(\boldsymbol{f}^*)>t_n\}} \,|\, X_{t_0} = x_{t_0}\right],
\end{aligned}$$

where the final step is justified by law of total expectation. The expected exposure under the $\mathbb{P}-$measure can be rewritten in the following way

$$\begin{aligned}
\mathrm{EE}_{\mathbb{P}}(t_n) =& \mathbb{E}_{\mathbb{P}}\left[\mathbb{E}^n_{\mathbb{Q}}[\mathrm{e}^{-r(\tau_n(\boldsymbol{f}_n^*)-t_n)} g_{\tau_n(\boldsymbol{f}_n^*)}(X_{\tau_n(\boldsymbol{f}_n^*)})] \mathbb{I}_{\{\tau(\boldsymbol{f}^*)>t_n\}} \,|\, X_{t_0} = x_{t_0}\right] \\
=& \int_{(y_n,\ldots,y_1)\in\mathbb{R}^d\times\cdots\times\mathbb{R}^d} \mathbb{E}_{\mathbb{Q}}\left[\mathrm{e}^{-r(\tau_n(\boldsymbol{f}_n^*)-t_n)} g_{\tau_n(\boldsymbol{f}_n^*)}(X_{\tau_n(\boldsymbol{f}_n^*)}) \,|\, X_{t_n} = y_n\right] \\
& \times \mathbb{I}_{\{\tau(\boldsymbol{f}^*)>t_n\}} p_{X_{t_1},\ldots,X_{t_n}}(y_n, \ldots, y_1) \mathrm{d}y_n \cdots \mathrm{d}y_1 \\
=& \int_{(y_n,\ldots,y_1)\in\mathbb{R}^d\times\cdots\times\mathbb{R}^d} \mathbb{E}_{\mathbb{Q}}\left[\mathrm{e}^{-r(\tau_n(\boldsymbol{f}_n^*)-t_n)} g_{\tau_n(\boldsymbol{f}_n^*)}(X_{\tau_n(\boldsymbol{f}_n^*)}) \,|\, X_{t_n} = y_n\right] \\
& \times \mathbb{I}_{\{\tau(\boldsymbol{f}^*)>t_n\}} \frac{p_{X_{t_n},\ldots,X_{t_1}}(y_n, \ldots, y_1)}{q_{X_{t_n},\ldots,X_{t_1}}(y_n, \ldots, y_1)} q_{X_{t_n},\ldots,X_{t_1}}(y_n, \ldots, y_1) \mathrm{d}y_n \cdots \mathrm{d}y_1,
\end{aligned} \tag{2.13}$$

where $q_{X_{t_n},\ldots,X_{t_1}}(y_n, \ldots, y_1)$ and $p_{X_{t_n},\ldots,X_{t_1}}(y_n, \ldots, y_1)$ are joint densities of $X_{t_n}, \ldots, X_{t_1}$ (conditioned on $X_{t_0} = x$) under the measures $\mathbb{P}$ and $\mathbb{Q}$, respectively. The fact that $\mathbb{P}$ and $\mathbb{Q}$ are equivalent, guarantees that the quotient in (2.13) is well defined. Furthermore, due to the Markov property of X, we have

$$p_{X_{t_n},\ldots,X_{t_1}}(y_n, \ldots, y_1) = p_{X_{t_n}|X_{t_{n-1}}}(y_n|y_{n-1}) \times \cdots \times p_{X_{t_1}|X_{t_0}}(y_1|y_0). \tag{2.14}$$

The same argumentation holds for $q_{X_{t_n},\dots,X_{t_1}}(y_n,\dots,y_1)$. The proof is finalized by inserting the product of density functions (2.14) in (2.13) and writing the expression as a $\mathbb{Q}$ expectation, and again, use the law of total expectation. □

Theorem 2 opens up for approximations of the expected exposure directly from the discounted cash-flows. We now consider the special case, when X is described by a diffusion-type stochastic differential equation (SDE). Let $\mu^{\mathbb{Q}}\colon [t_0,t_N]\times\mathbb{R}^d\to\mathbb{R}^d$ and $\sigma\colon [t_0,\,t_N]\times\mathbb{R}^d\to\mathbb{R}^d\times\mathbb{R}^d$ be the drift and diffusion coefficients, respectively, and let $(W_t^{\mathbb{Q}})_{t\in[t_0,t_N]}$ be a d-dimensional standard Brownian motion under the measure $\mathbb{Q}$. We assume that $\mu^{\mathbb{Q}}$ and σ satisfy the usual conditions (see *e.g.*, [40]) for existence of a unique, strong solution X to

$$\mathrm{d}X_t=\mu^{\mathbb{Q}}(t,\,X_t)\mathrm{d}t+\sigma(t,\,X_t)\mathrm{d}W_t^{\mathbb{Q}},\;t\in[t_0,\,t_N];\quad X_{t_0}=x_{t_0}.$$

Let $\mu^{\mathbb{P}}\colon [t_0,t_N]\times\mathbb{R}^d\to\mathbb{R}^d$ and assume that $\sigma(t,\,X_t)$ is invertible and that for $t\in[t_0,t_N]$[10] $\|\sigma^{-1}(t,\,X_t)\|_{\max}$ is bounded almost surely. Then, $(W_t^{\mathbb{P}})_{t\in[t_0,t_N]}$ given by

$$\mathrm{d}W_t^{\mathbb{P}}=-\sigma^{-1}(t,\,X_t)\left(\mu^{\mathbb{P}}(t,\,X_t)-\mu^{\mathbb{Q}}(t,\,X_t)\right)\mathrm{d}t+\mathrm{d}W_t^{\mathbb{Q}}$$

is a Brownian motion under the measure $\mathbb{P}$. Furthermore, under the measure $\mathbb{P}$ it holds almost surely that X is described by

$$\mathrm{d}X_t=\mu^{\mathbb{P}}(t,\,X_t)\mathrm{d}t+\sigma(t,\,X_t)\mathrm{d}W_t^{\mathbb{P}},\;t\in[t_0,\,t_N];\quad X_{t_0}=x_{t_0}. \tag{2.15}$$

As a way to rewrite $\mathrm{EE}_{\mathbb{P}}(t)$ as a conditional expectation under the measure $\mathbb{Q}$, we define a process $U=(U_t)_{t\in[t_0,t_N]}$, which follows the SDE

$$\mathrm{d}U_t=\mu^{\mathbb{P}}(t,\,U_t)\mathrm{d}t+\sigma(t,\,U_t)\mathrm{d}W_t^{\mathbb{Q}},\;t\in[t_0,t_N];\quad U_{t_0}=x_{t_0}. \tag{2.16}$$

The reason for introducing this process is that U has the same distribution under the measure $\mathbb{Q}$ as X has under the measure $\mathbb{P}$. We can then express $\mathrm{EE}_{\mathbb{P}}(t_n)$ with only $\mathbb{Q}-$expectations in the following way

$$\begin{aligned}
&\mathrm{EE}_{\mathbb{P}}(t_n) && (2.17)\\
&=\mathbb{E}_{\mathbb{P}}\big[V_{t_n}(X_{t_n})\mathbb{I}_{\{\tau(\boldsymbol{f}^*(X))>t_n\}}\,|\,X_{t_0}=x_{t_0}\big]\\
&=\mathbb{E}_{\mathbb{Q}}\big[V_{t_n}(U_{t_n})\mathbb{I}_{\{\tau(\boldsymbol{f}^*(U))>t_n\}}\,|\,U_{t_0}=x_{t_0}\big]\\
&=\mathbb{E}_{\mathbb{Q}}\Big[\mathbb{E}_{\mathbb{Q}}\left[\mathrm{e}^{-r(\tau_n(\boldsymbol{f}_n^*(X))-t_n)}g_{\tau_n(\boldsymbol{f}_n^*(X))}\left(X_{\tau_n(\boldsymbol{f}_n^*(X))}\right)\,|\,X_{t_n}=U_{t_n}\right]\times\mathbb{I}_{\{\tau(\boldsymbol{f}^*(U))>t_n\}}\,|\,U_{t_0}=x_{t_0}\Big]. && (2.18)
\end{aligned}$$

Remark 2.2.1. *Regarding the equality between the right hand side of the first line and the second line, we want to emphasize that X under the measure $\mathbb{P}$, and U under the measure $\mathbb{Q}$ do not represent the same stochastic process (as they would have done after a change of measure). If they were, then the conditional expectation would change, and the equality would not hold. To enforce the equality to hold we could have corrected with the Radon–Nikodym derivative, and obtained* (2.12)*. However, to find a way to write $\mathrm{EE}_{\mathbb{P}}(t_n)$ without having to*

[10] For a matrix $A\in\mathbb{R}^{d\times d}$, $\|A\|_{\max}=\max_{ij}|A_{ij}|$.

include the Radon–Nikodym derivative, we introduce another process U, which is distributed in such a way that the equality holds, i.e., the conditional expectation under $\mathbb{P}$, when using X, should equal the conditional expectation under $\mathbb{Q}$, when using U. For this to hold, it is sufficient that the distribution of X under the measure $\mathbb{P}$ equals the distribution of U under the measure $\mathbb{Q}$, which is satisfied when X follows (2.15) *and U follows* (2.16).

Remark 2.2.2. *The final equality is obtained by the fact that $V_{t_n}(U_{t_n})$ is the option value at the (random) state (t_n, U_{t_n}), given by* (2.5).

Before we get rid of the inner expectation in (2.18), we need to define a process following (2.16) on $[t_0, t_n]$, and the dynamics of (2.15) on $[t_n, t_N]$ with (stochastic) initial condition $X_{t_n} = U_{t_n}$. We denote such a process by $\tilde{X}^{t_n} = (\tilde{X}^{t_n}_t)_{t\in[t_0,t_N]}$, and conclude that $\tilde{X}^{t_n}$ should satisfy the following SDE

$$\mathrm{d}\tilde{X}^{t_n}_t = \mu^{\mathbb{P},\mathbb{Q},t_n}(t,\, \tilde{X}^{t_n}_t)\mathrm{d}t + \sigma(t,\, \tilde{X}^{t_n}_t)\mathrm{d}W^{\mathbb{Q}}_t,\ t \in [t_0,\, t_N]; \quad \tilde{X}^{t_n}_{t_0} = x_{t_0}, \tag{2.19}$$

where $\mu^{\mathbb{P},\mathbb{Q},t_n}(t,\cdot) = \mu^{\mathbb{P}}(t,\cdot)\mathbb{I}_{\{t\le t_n\}} + \mu^{\mathbb{Q}}(t,\cdot)\mathbb{I}_{\{t>t_n\}}$. Note that we have implicitly assumed that also $\mu^{\mathbb{P}}$ satisfies the usual conditions for existence of a unique strong solution, U, to (2.16). As a consequence of this assumption we are also guaranteed that there exists a unique strong solution, $\tilde{X}^{t_n}$, to (2.19). We can then use the law of total expectation to obtain

$$\begin{aligned}\mathrm{EE}_{\mathbb{P}}(t_n) &= \mathbb{E}_{\mathbb{Q}}\left[V_{t_n}(\tilde{X}^{t_n}_{t_n})\mathbb{I}_{\{\tau(\boldsymbol{f}^*)>t_n\}} \,|\, \tilde{X}^{t_n}_{t_0} = x_{t_0}\right] \\ &= \mathbb{E}_{\mathbb{Q}}\left[\mathrm{e}^{-r(\tau_n(\boldsymbol{f}^*_n)-t_n)} g_{\tau_n(\boldsymbol{f}^*_n)}\left(\tilde{X}^{t_n}_{\tau_n(\boldsymbol{f}^*_n)}\right)\mathbb{I}_{\{\tau(\boldsymbol{f}^*)>t_n\}} \,|\, \tilde{X}^{t_n}_{t_0} = x_{t_0}\right],\end{aligned} \tag{2.20}$$

where we remind ourselves that $\tau(\boldsymbol{f}^*) = \tau(\boldsymbol{f}^*(\tilde{X}^{t_n}))$ and $\tau_n(\boldsymbol{f}^*_n) = \tau_n(\boldsymbol{f}^*_n(\tilde{X}^{t_n}))$.

In the next sections we describe a method to approximate $\boldsymbol{f}^*(\cdot)$. It is then straight forward to estimate (2.11), (2.12) and (2.20) by Monte-Carlo sampling. Furthermore, in Section 2.4, we introduce a method to approximate the price function $V_{t_n}(\cdot)$, which makes it straight forward to also approximate the potential future exposure (2.10).

2.3 Learning stopping decisions

In the first part of this section, we present the DOS algorithm, which was proposed in [9]. The idea is to use fully connected neural networks to approximate the decision functions introduced in the previous section. The neural networks are optimized backwards in time with the objective to maximize the expected discounted cash-flow at each exercise date. In the second part of this section, we suggest some adjustments that can be done in order to make the optimization more efficient.

2.3.1 The Deep Optimal Stopping algorithm

As indicated above, the core of the algorithm is to approximate decision functions. To be more precise, for $n \in \{0,\, 1,\, \ldots,\, N\}$, the decision function f_n is approximated by a fully connected neural network of the form $f^{\theta_n}_n \colon \mathbb{R}^d \to \{0,\, 1\}$, where $\theta_n \in \mathbb{R}^{q_n}$ is a vector containing all the $q_n \in \mathbb{N}$ trainable[11] parameters in network n. We assume that the initial state, $x_0 \in \mathbb{R}^d$,

[11] Parameters that are subject to optimization.

is such that it is sub-optimal to exercise the option at t_0, and therefore set θ_0 such that $f_0^{\theta_0}(x_0) = 0$ (for a further discussion, see Remark 6 in [9]). Since binary decision functions are discontinuous, and therefore unsuitable for gradient-type optimization algorithms, we use as an intermediate step, the neural network $F_n^{\theta_n} : \mathbb{R}^d \to (0, 1)$. Instead of a binary decision, the output of the neural network $F_n^{\theta_n}$ can be viewed as the probability[12] for exercise to be optimal. This output is then mapped to 1 for values above (or equal to) 0.5, and to 0 otherwise, by defining $f_n^{\theta_n}(\cdot) = \mathfrak{a} \circ F_n^{\theta_n}(\cdot)$, where $\mathfrak{a}(x) = \mathbb{I}_{\{x \geq 1/2\}}$. Our objective is to find θ_n such that

$$\mathbb{E}_{\mathbb{Q}}^n \left[f_n^{\theta_n}(X_{t_n}) g_{t_n}(X_{t_n}) + (1 - f_n^{\theta_n}(X_{t_n})) \mathrm{e}^{-r(\tau_{n+1}(\boldsymbol{f}_{n+1}^*) - t_n)} g_{\tau_{n+1}(\boldsymbol{f}_{n+1}^*)} \left(X_{\tau_{n+1}(\boldsymbol{f}_{n+1}^*)} \right) \right], \quad (2.21)$$

is as close as possible to $V_{t_n}(X_{t_n})$ (in mean squared sense), where $\boldsymbol{f}_{n+1}^*$ is the vector of optimal decision functions, defined in (2.4). Although (2.21) is an accurate representation of the optimization problem, it gives us some practical problems. In general, we have no access to either $\boldsymbol{f}_{n+1}^*$ or the distribution of $V_{t_n}(X_{t_n})$ and in most cases the expectation needs to be approximated. We however notice that at maturity, the option value is equal to its intrinsic value, *i.e.*, $V_{t_N}(\cdot) \equiv g_{t_N}(\cdot)$, which implies that $f_N^* \equiv 1$ and $\tau_N(\boldsymbol{f}_N^*) = t_N$. With this insight, we can write (2.21) with $n = N - 1$ in the form

$$\mathbb{E}_{\mathbb{Q}}^{N-1} \left[f_{N-1}^{\theta_{N-1}}(X_{t_{N-1}}) g_{t_{N-1}}(X_{t_{N-1}}) + (1 - f_{N-1}^{\theta_{N-1}}(X_{t_{N-1}})) \mathrm{e}^{-r(t_N - t_{N-1})} g_{t_N}(X_{t_N}) \right], \quad (2.22)$$

which can be approximated by Monte-Carlo sampling. Given $M \in \mathbb{N}$ samples, distributed as X, which for $m \in \{1, 2, \ldots, M\}$ is denoted by $x = (x_{t_n}(m))_{n=0}^N$, we can approximate (2.22) by

$$\frac{1}{M} \sum_{m=1}^{M} \Big(f_{N-1}^{\theta_{N-1}}(x_{t_{N-1}}(m)) g_{t_{N-1}}(x_{t_{N-1}}(m)) + (1 - f_{N-1}^{\theta_{N-1}}(x_{t_{N-1}}(m))) \mathrm{e}^{-r(t_N - t_{N-1})} g_{t_N}(x_{t_N}(m)) \Big). \quad (2.23)$$

Note that the only unknown entity in (2.23) is the parameter θ_{N-1} in the decision function $f_{N-1}^{\theta_{N-1}}$. Furthermore, we note that we want to find θ_{N-1} such that (2.23) is maximized, since it represents the average cash-flow in $[t_{N-1}, t_N]$. Once θ_{N-1} is optimized, we use this parameter and find θ_{N-2} such that the average cash-flow on $[t_{N-2}, t_N]$ is maximized.

In the next section, we explain the parameters θ_n and present the structure for the neural networks used in this chapter.

2.3.1.1 Specification of the neural networks used

For completeness, we introduce all the trainable parameters that are contained in each of the parameters $\theta_1, \theta_2, \ldots, \theta_{N-1}$, and present the structure of the networks.

- We denote the dimension of the input layers by $\mathfrak{D}^{\text{input}} \in \mathbb{N}$, and we assume the same input dimension for all $n \in \{1, 2, \ldots, N-1\}$ networks. The input is assumed to be the market state $x_{t_n}^{\text{train}} \in \mathbb{R}^d$, and hence $\mathfrak{D}^{\text{input}} = d$. However, we can add additional

[12] However the interpretation as a probability may be helpful, one should be careful since it is not a rigorous mathematical statement. It should be clear that there is nothing random about the stopping decisions, since the stopping time is $\mathcal{F}_t-$measurable. It can also be interpreted as a measure on how certain we can be that exercise is optimal.

information to the input that is mathematically redundant but helps the training, *e.g.*, the immediate pay-off, to obtain as input $\left(\text{vec}(x_{t_n}^{\text{train}}(m)),\, g_{t_n}\left(x_{t_n}^{\text{train}}(m)\right)\right) \in \mathbb{R}^{d+1}$, which would give $\mathfrak{D}^{\text{input}} = d+1$. In [9], the authors claim that by adding the immediate pay-off to the input, the efficiency of the algorithm was improved;

- For network $n \in \{1, 2, \ldots, N-1\}$, we denote the number of layers[13] by $\mathfrak{L}_n \in \mathbb{N}$, and for layer $\ell \in \{1, 2, \ldots, \mathfrak{L}_n\}$, the number of nodes by $\mathfrak{N}_{n,\ell} \in \mathbb{N}$. Note that $\mathfrak{N}_{n,1} = \mathfrak{D}^{\text{input}}$;
- For network $n \in \{1, 2, \ldots, N\}$, and layer $\ell \in \{2, 3, \ldots, \mathfrak{L}_n\}$ we denote the weight matrix, acting between layers $\ell-1$ and ℓ, by $w_{n,\ell} \in \mathbb{R}^{\mathfrak{N}_{n,\ell-1} \times \mathfrak{N}_{n,\ell}}$, and the bias vector by $b_{n,\ell} \in \mathbb{R}^{\ell}$;
- For network $n \in \{1, 2, \ldots, N\}$, and layer $\ell \in \{2, 3, \ldots, \mathfrak{L}_n\}$ we denote the (scalar) activation function by $a_{n,\ell} \colon \mathbb{R} \to \mathbb{R}$ and the vector activation function by $\boldsymbol{a}_{n,\ell} \colon \mathbb{R}^{\mathfrak{N}_{n,\ell}} \to \mathbb{R}^{\mathfrak{N}_{n,\ell}}$, which for $x = (x_1, x_2, \ldots, x_{\mathfrak{N}_{n,\ell}})$ is defined by

$$\boldsymbol{a}_{n,\ell}(x) = \begin{pmatrix} a_{n,\ell}(x_1) \\ \vdots \\ a_{n,\ell}(x_{\mathfrak{N}_{n,\ell}}) \end{pmatrix};$$

- The output of our network should belong to $(0,1) \subset \mathbb{R}$, meaning that the output dimension of our neural network, denoted by $\mathfrak{D}^{\text{output}}$ should equal 1. To enforce the output to only take on values in $(0,1)$, we restrict ourselves to activation functions of the form $a_{n,\mathfrak{L}_n} \colon \mathbb{R} \to (0,1)$.

Network $n \in \{1, 2, \ldots N-1\}$ is then defined by

$$F_n^{\theta_n}(\cdot) = L_{n,\mathfrak{L}_n} \circ L_{n,\mathfrak{L}_n-1} \circ \cdots \circ L_{n,1}(\cdot), \tag{2.24}$$

where for $n \in \{1, 2, \ldots, N-1\}$ and for $x \in \mathbb{R}^{\mathfrak{L}_{n,\ell-1}}$, the layers are defined as

$$L_{n,\ell}(x) = \begin{cases} x, & \text{for } \ell = 1, \\ \boldsymbol{a}_{n,\ell}(w_{n,\ell}^T x + b_{n,\ell}), & \text{for } \ell \geq 2, \end{cases}$$

where $w_{n,\ell}^T$ is the matrix transpose of $w_{n,\ell}$. The trainable parameters of network $n \in \{1, 2, \ldots, N-1\}$ are then given by the list

$$\theta_n = \{w_{n,2}, b_{n,2}, w_{n,3}, b_{n,3}, \ldots, w_{n,\mathfrak{L}_n}, b_{n,\mathfrak{L}_n}\}.$$

Furthermore, since we have $N-1$ neural networks, we denote by

$$\boldsymbol{\theta}_n = \{\theta_n, \theta_{n+1}, \ldots, \theta_{N-1}\}$$

the trainable parameters in the neural networks at exercise dates $\mathbb{T}_n$ and by $\boldsymbol{\theta} = \boldsymbol{\theta}_1$.

[13] Input and output layers included.

2.3.1.2 Training and valuation

The main idea of the training and valuation procedure is to fit the parameters to some training data, and then use the fitted parameters to make informed decisions with respect to some unseen, so-called, valuation data.

The training part of the algorithm is summarized below in pseudo code.

Training phase:
Sample $M_{\text{train}} \in \mathbb{N}$ independent realizations of X, which for $m \in \{1, 2, \dots, M_{\text{train}}\}$ are denoted $(x_{t_n}^{\text{train}}(m))_{n=0}^{N}$. At maturity, define the cash-flow as $\text{CF}_N(m) = g_{t_N}(x_{t_N}^{\text{train}}(m))$.
For $n = N-1, N-2, \dots, 1$, do the following:

1. Find a $\hat{\theta}_n \in \mathbb{R}^{q_n}$ which approximates

$$\hat{\theta}_n^* \in \arg\max_{\theta \in \mathbb{R}^{q_n}} \Big(\frac{1}{M_{\text{train}}} \sum_{m=1}^{M_{\text{train}}} F_n^\theta(x_{t_n}^{\text{train}}(m)) g_{t_n}(x_{t_n}^{\text{train}}(m)) + \Big(1 - F_n^\theta(x_{t_n}^{\text{train}}(m))\Big) \, \mathrm{e}^{-r(t_{n+1}-t_n)} \text{CF}_{n+1}(m) \Big).$$

   In machine learning terminology, this would give an (empirical) loss-function of the form

$$L(\theta; x^{\text{train}}) = - \frac{1}{M_{\text{train}}} \sum_{m=1}^{M_{\text{train}}} F_n^\theta\Big(x_{t_n}^{\text{train}}(m)\Big) g_{t_n}\Big(x_{t_n}^{\text{train}}(m)\Big) + \Big(1 - F_n^\theta\Big(x_{t_n}^{\text{train}}(m)\Big)\Big) \mathrm{e}^{-r(t_{n+1}-t_n)} \text{CF}_{n+1}(m).$$

   The minus sign in the loss-function transforms the problem from a maximization to minimization, which is the standard formulation in the machine learning community. Note the straight forward relationship between the loss function and the average cash-flows in (2.23). The only difference is that we use continuous stopping decisions here, which can be viewed as stopping probabilities. In practice, this implies that during training, it is possible to exercise a fraction of an option since the output is any number between zero and one. On the other hand, we expect a converged output to take on values very close to either one or zero. The reason for this setup is that a discontinuous output would cause difficulties for the optimization scheme, which is of gradient decent type. In practice, the data is often divided into mini-batches, for which the loss-function is minimized consecutively.

2. For $m = 1, 2, \dots, M_{\text{train}}$, update the discounted cash-flow according to:

$$\begin{aligned} &\text{CF}_n(m) \\ &= f_n^{\hat{\theta}_n}(x_{t_n}^{\text{train}}(m)) g_{t_n}(x_{t_n}^{\text{train}}(m)) + \Big(1 - f_n^{\hat{\theta}_n}(x_{t_n}^{\text{train}}(m))\Big) \, \mathrm{e}^{-r(t_{n+1}-t_n)} \text{CF}_{n+1}(m). \end{aligned}$$

The performance of the algorithm is not particularly sensitive to the specific choice of the number of hidden layers, number of nodes, optimization algorithm, etc. Below is a list of the most relevant parameters/structural choices:

- Initialization of the trainable parameters, where a typical procedure is to initialize the biases to 0, and sample the weights independently from a normal distribution;
- The activation functions $a_{\ell,n}$, which are used to add a non-linear structure to the neural networks. In our case we have the strict requirement that the activation function of the output layer maps $\mathbb{R}$ to $(0,1)$. This could, however, be relaxed as long as the activation function is both upper and lower bounded, since we can always scale and shift such output to take on values only in $(0,1)$. For a discussion of different activation functions, see *e.g.,* [41].;
- The batch size, $B_n \in \{1,2,\ldots,M_{\text{train}}\}$, is the number of training samples used for each update of θ_n, *i.e.,* with $B_n = M_{\text{train}}$, the loss function is of the form defined in step 1 above. Note that if we want all batches to be of equal size, we need to choose B_n to be a multiplier of M_{train};
- For each update of θ_n, we use an optimization algorithm, for which a common choice is the Adam optimizer, proposed in [42]. Depending on the choice of optimization algorithm, there are different parameters related to the specific algorithm to be chosen. One example is the so-called learning rate which decides how much the parameter, θ_n is adjusted after each batch.

Once the parameters, $\{\theta_1, \theta_2, \ldots, \theta_{N-1}\}$, have been optimized we can use the algorithm for valuation.

Valuation phase:
Sample $M_{\text{val}} \in \mathbb{N}$ independent realizations of X, denoted $\left(x_{t_n}^{\text{val}}(m)\right)_{n=0}^{N}$. We emphasize that the valuation data should be independent from the training data. Denote the vector of decision functions by

$$\boldsymbol{f}_n^{\hat{\boldsymbol{\theta}}} = \left(f_n^{\hat{\theta}_n},\, f_{n+1}^{\hat{\theta}_{n+1}}, \ldots, f_{N-1}^{\hat{\theta}_{N-1}}\right),$$

and $\boldsymbol{f}^{\hat{\boldsymbol{\theta}}} = \boldsymbol{f}_0^{\hat{\boldsymbol{\theta}}}$. We then obtain for sample m, *i.e.,* $x^{\text{val}}(m)$, the following stopping rule at time t_n

$$\tau_n\left(\boldsymbol{f}_n^{\hat{\boldsymbol{\theta}}}\left(x^{\text{val}}(m)\right)\right) = \sum_{m=n}^{N} t_m f_m^{\hat{\theta}_m}\left(x_{t_m}^{\text{val}}(m)\right) \prod_{j=0}^{m-1}\left(1 - f_j^{\hat{\theta}_j}\left(x_{t_j}^{\text{val}}(m)\right)\right).$$

The estimated option value at t_0 is then given by

$$\hat{V}_{t_0}(x_0) = \frac{1}{M_{\text{val}}} \sum_{m=1}^{M_{\text{val}}} \mathrm{e}^{-r\left(\tau\left(\boldsymbol{f}^{\hat{\theta}}\right)-t_0\right)} g_{\tau\left(\boldsymbol{f}^{\hat{\boldsymbol{\theta}}}\right)}\left(x_{\tau(\boldsymbol{f}^{\hat{\boldsymbol{\theta}}})}^{\text{val}}(m)\right), \tag{2.25}$$

where we recall that $\tau(\boldsymbol{f}^{\hat{\boldsymbol{\theta}}}) = \tau_0\left(\boldsymbol{f}_0^{\hat{\boldsymbol{\theta}}}\left(x^{\text{val}}(m)\right)\right)$. Note that, by construction, any stopping strategy is sub-optimal, implying that the estimate (2.25) is biased low. It should be pointed out that it is possible to derive a biased high estimate of (2.1) from a dual formulation of the optimal stopping problem, which is described in [9]. In addition, numerical results in [9] show a tight interval for the biased low and biased high estimates for a wide range of different problems.

2.3.2 Proposed adjustments to the algorithm

The presentation of the DOS algorithm in [9] is in a general form. In addition to the pricing of Bermudan options, the authors considered the non-Markovian problem to optimally stop a fractional Brownian motion (this is done by including also the historical states in the current state of the system). Since the aim of this chapter is more specific (to approximate exposure profiles of Bermudan options), it is natural to use more of the known underlying structure of this specific problem. In this section we use some properties of the specific problems, and propose some adjustments to the DOS-algorithm, which make the training procedure more efficient.

2.3.2.1 Reuse of neural network parameters

The first proposed adjustment is to reuse parameters of neural networks that have already been optimized. We note that for a single Bermudan option (possibly with a high-dimensional underlying asset) the pay-off functions are identical at all exercise dates, *i.e.,* $g_{t_n} = g_{t_m}$ for all $t_n, t_m \in \mathbb{T}$. In this case the stopping rules at adjacent exercise dates are similar, especially when $t_{n+1} - t_n$ is small. We therefore use the stopping strategy at t_{n+1} as an "initial guess" for the stopping strategy at t_n. This is done by initializing the trainable parameters in network n by the already optimized parameters in network $n+1$, *i.e.,* at t_n, initialize θ_n by $\hat{\theta}_{n+1}$. This allows us to use smaller learning rates leading to a more efficient algorithm.

This technique can be viewed as a form of *transfer learning*, in which a pre-trained model is used on a new problem. However, it is important to bear in mind that, given the high non-linearity of a neural network structure, it is not clear that two similar problems have similar optimal parameters. This may cause the optimization procedure to end up in a bad local minima.

2.3.2.2 Use simple stopping decisions when possible

The term simple stopping decisions is loosely defined as stopping decisions that follow directly without any sophisticated optimization algorithm. The most obvious example is when the contract is out-of-the-money, in which case it is never optimal to exercise. For $t_n \in \mathbb{T}$, we define the set of in-the-money points and out-of-the-money points as

$$\text{ITM}_n = \{x \in \mathbb{R}^d \,|\, g_{t_n}(x) > 0\}, \quad \text{OTM}_n = \{x \in \mathbb{R}^d \,|\, g_{t_n}(x) = 0\},$$

respectively. Another, less obvious insight is that, given a single Bermudan option with identical pay-off functions at all exercise dates, if it is optimal to exercise at (t_n, x), then it is also optimal to exercise at (t_{n+1}, x). Or in other words, the exercise region is non-decreasing with time. This statement is formulated as a theorem below.

Theorem 3. *Define the set of exercise dates by* $\{t_0, \ldots, t_n, t_{n+1}, \ldots, t_N, t_N + \Delta\}$*, and let* $\Delta = t_{n+1} - t_n = t_{N+1} - t_N \geq 0$*. Note that an equidistant time grid is sufficient, but not necessary for the above to be satisfied. Moreover, assume that*

$$V_{t_n}(\cdot\,; t_N) = V_{t_{n+1}}(\cdot\,; t_N + \Delta),$$

where t_N and $t_N + \Delta$ indicate the maturity of otherwise identical contracts of the form (2.2)*, with $g = g_{t_n}$ for all exercise dates t_n. Then, for any $\bar{x} \in \mathcal{E}_n$, it holds that $\bar{x} \in \mathcal{E}_{n+1}$.*

Proof. Since $\bar{x} \in \mathcal{E}_n$, $V_{t_n}(\bar{x}\,; t_N) = g(\bar{x})$ and $V_{t_n}(\bar{x}\,; t_N) = V_{t_{n+1}}(\bar{x}\,; t_N + \Delta)$ we have that $V_{t_{n+1}}(\bar{x}\,; t_N + \Delta) = g(\bar{x})$. From (2.2) we also see that $V_{t_{n+1}}(\bar{x}\,; t_N) \leq V_{t_{n+1}}(\bar{x}\,; t_N + \Delta)$ and $V_{t_{n+1}}(\bar{x}\,; t_N) \geq g(\bar{x})$. Therefore $V_{t_{n+1}}(\bar{x}\,; t_N) = g(\bar{x})$ and $\bar{x} \in \mathcal{E}_{n+1}$. □

The condition $V_{t_n}(\cdot; t_N) = V_{t_{n+1}}(\cdot; t_N + \Delta)$ requires that the market states are time-invariant. This is easily satisfied if the contractual details (pay-off structure) as well as the drift and diffusion coefficients are time-invariant. Theorem 3 shows that the exercise region is non-decreasing with time, but since the optimization of the neural network parameters is carried out backwards in time we instead use the fact that the continuation region is non-increasing with time. In practice, this leads to the following three alternatives:

A1. Use all training data in the optimization algorithm (as the algorithm is described in Subsection 2.3.1);

A2. At $t_n \in \mathbb{T}$, use the subset of the training data satisfying $x_{t_n}^{\text{val}}(m) \in \text{ITM}_n$ in the optimization algorithm. Define the decision functions as

$$f_n^{\hat{\theta}_n}(\cdot) = \mathbb{I}_{\{g_{t_n}(\cdot) > 0\}}(\mathfrak{a} \circ F_n^{\hat{\theta}_n}(\cdot)).$$

To only use "in the money paths" is also employed in the Least Squares Method (LSM), proposed in [25];

A3. At t_n use the subset of the training data $x_{t_n}^{\text{train}}(m) \in \mathcal{E}_{n+1}$ in the optimization algorithm. Define the decision functions as

$$f_n^{\hat{\theta}_n}(\cdot) = f_{n+1}^{\hat{\theta}_{n+1}}(\cdot)(\mathfrak{a} \circ F_n^{\hat{\theta}_n}(\cdot)).$$

In Figure 2.1 the three cases above are visualized for a two-dimensional max call option at one of the exercise dates. To the left we have the blue points belonging to $\mathcal{E}_{n+1}$ (used for optimization in A3), the blue and red points belong to ITM_n (used for optimization in A2) and the blue, red and yellow points are all the available data (used for optimization in A1). To the right, we see the fraction of the total data used in each case at each exercise date.

In general, much can be gained from carefully analysing the structure of the exercise region. Some examples of problem-specific structures are high-dimensional ($d \geq 2$) geometric and arithmetic average options. In these problems, the making exercise decisions can be reduced to one-dimensional problems, *i.e.,* the exercise decisions can be made with a one-dimensional, instead of a $d-$dimensional input. This is possible since the geometric (or arithmetic) average is a significant statistic for the exercise decision for this specific option. This means that it is sufficient to have access to the geometric (or arithmetic) average in order to make optimal exercise decisions, reducing the problem from d-dimensional to one-dimensional. The question is then whether or not this is possible for any option. The answer is, to the best of our knowledge, unfortunately, no. For instance, as noted in [43], for the two-dimensional Bermudan max-call option, it is never optimal to exercise the option prior to maturity if both assets

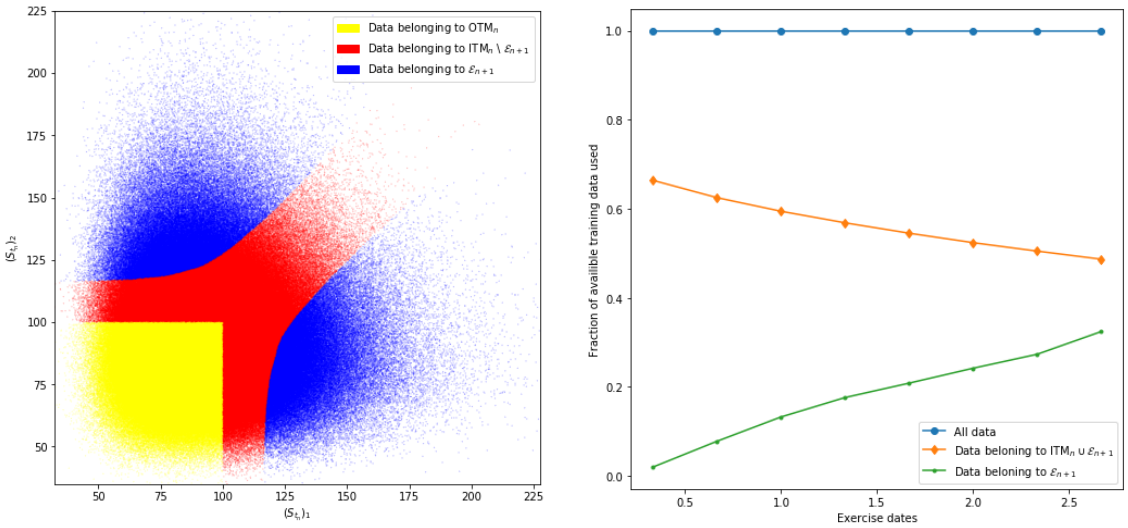

Figure 2.1: Left: Blue points in $\mathcal{E}_{n+1}$ (used for optimization in A3), blue and red points in ITM_n (used for optimization in A2) and the blue, red and yellow points are all the available data (used for optimization in A1). **Right:** The fraction of the total data used in each case at each exercise date.

have the same value. As strange as this may seem, this holds true no matter how deep in the money the option is. A direct consequence of this is that the maximum of the two assets is *not* a sufficient statistic for the exercise decision. However, by dividing the exercise region of the two-dimensional problem into two subregions, $\mathcal{E}_1$ in which the first asset is largest, and $\mathcal{E}_2$ in which the second asset is largest, certain interesting properties, that are present for one-dimensional problems appear. For one-dimensional Bermudan options, the exercise region can be expressed by a graph of a function in which the epigraph is the exercise region. For the two-dimensional Bermudan max-call option, the exercise region can be expressed with two epigraphs and exercise decisions can be made by simple and/or statements connecting these two graphs. This implies that, instead of searching for a complex exercise surface in two dimensions, one can base the exercise decisions on combining information from two simple epigraphs, similar to the ones present in one-dimensional problems. For a more detailed discussion, see [43], in which the properties of the sub-regions for two-dimensional the max-call option are discussed, or in [44], in which the proposed numerical method builds upon sub-regions and graphs of functions of the exercise region.

2.4 Learning pathwise option values

In Section 2.3 an algorithm to learn stopping decisions was described and (2.25) gives an approximation of the option value at time t_0, given some deterministic initial state $X_{t_0} = x_{t_0} \in \mathbb{R}^d$. As described in Subsection 2.2.2, to compute exposure profiles we sometimes need additional information about the future distribution of the option values. In this section, we present two methods to approximate the pathwise option values at all exercise dates. The first method is the well-established Ordinary Least Squares (OLS) regression and the second method is a neural network-based least squares regression. Both methods rely on projections of conditional expectations on a finite-dimensional function space.

2.4.1 Formulation of regression problem

Central for the regression algorithms presented in this section is the cash-flow process, $Y = (Y_{t_n})_{n=0}^N$, defined as[14]

$$Y_{t_n} = \mathrm{e}^{-r(\tau_n(\boldsymbol{f}_n^*)-t_n)} g_{\tau_n(\boldsymbol{f}_n^*)}\left(X_{\tau_n(\boldsymbol{f}_n^*)}\right), \tag{2.26}$$

where $\tau_n(\cdot)$ and $\boldsymbol{f}^*$ are defined in (2.3) and (2.4), respectively. We assume that for $t_n \in \mathbb{T}$ it holds that

$$\mathbb{E}_{\mathbb{Q}}[g_{t_n}(X_{t_n})^2] < \infty,$$

which also implies that $\mathbb{E}_{\mathbb{Q}}[Y_{t_n}^2] < \infty$. The following theorem states that the option value, at some $t_n \in \mathbb{T}$, is equivalent (in L_2 sense) to the so-called regression function. Furthermore, we see that the regression function can be obtained by solving a minimization problem.

Theorem 4. *Let Y_{t_n} be as defined in* (2.26) *and for $h_n \in \mathcal{D}(\mathbb{R}^d; \mathbb{R})$, define the so-called L_2-risk as*

$$\mathbb{E}_{\mathbb{Q}}\left[|h_n(X_{t_n}) - Y_{t_n}|^2\right].$$

It then holds that

$$\mathbb{E}_{\mathbb{Q}}\left[|V_{t_n}(X_{t_n}) - Y_{t_n}|^2\right] = \min_{h_n \in \mathcal{D}(\mathbb{R}^d; \mathbb{R})} \mathbb{E}_{\mathbb{Q}}\left[|h_n(X_{t_n}) - Y_{t_n}|^2\right],$$

or equivalently

$$V_{t_n}(\cdot) \in \operatorname*{arg\,min}_{h_n \in \mathcal{D}(\mathbb{R}^d; \mathbb{R})} \mathbb{E}_{\mathbb{Q}}\left[|h_n(X_{t_n}) - Y_{t_n}|^2\right].$$

The above is a standard result and is proved by noting that the conditional expectation is the (least-squares) projection onto the Markov states, $h_n(X_{t_n})$.

In practice, the distribution of (X, Y) is usually unknown. On the other hand, we are often able to generate samples distributed as[15] (X, Y). We consider some $t_n \in \mathbb{T}$, and generate $M_{\text{reg}} \in \mathbb{N}$ independent realizations of the regression pair (X_{t_n}, Y_{t_n}), which we denote by $(x_{t_n}^{\text{reg}}(m), y_{t_n}^{\text{reg}}(m))_{m=1}^{M_{\text{reg}}}$. Similarly, we define the empirical L_2-risk by

$$\frac{1}{M_{\text{reg}}} \sum_{m=1}^{M_{\text{reg}}} |h_n(x_{t_n}^{\text{reg}}(m)) - y_{t_n}^{\text{reg}}(m)|^2.$$

With a fixed sample of regression pairs it is possible to find a function $h \in \mathcal{D}(\mathbb{R}^d; \mathbb{R})$ such that the empirical L_2-risk equals zero. However, such a function is not a consistent estimator in general. Therefore, we want to use a smaller class of more regular functions. When choosing the function class, which we denote by $\mathcal{A}_M$, we need to keep two aspects in mind;

P1. It should be "rich enough" to be able to approximate $V_{t_n}(\cdot)$ sufficiently accurately,

[14] Note that Y is the discounted cash-flow process, which in the training phase was denoted by CF_n. The reason for using Y_{t_n} instead in this section is that we want to emphasize that the pathwise valuation problem is, in fact, a standard regression problem in which X and Y usually are used to represent the observation vector, and the response variable, respectively.

[15] In fact, we can only generate samples distributed as $(X, \hat{Y})$, where $\hat{Y}$ is the approximate discounted cash-flow process obtained by using the neural network-based decision functions instead of the optimal decision functions in (2.26). We give a short explanation of how this affects the regression in the end of this section.

P2. It should not be "too rich" since that may cause the empirical L_2-risk being an inaccurate approximation of the L_2-risk. Since this problem is more severe for smaller M_{reg}, it is reasonable to have the sample size in mind when choosing the function class, and hence the subscript M on $\mathcal{A}_M$, where "reg" is dropped for notational convenience. A too rich function class may lead to what is known as overfitting in the machine learning community.

Given a sample and a function class $\mathcal{A}_M$, we define the empirical regression function as

$$m_M(\cdot) \in \underset{h \in \mathcal{A}_M}{\arg\min} \frac{1}{M_{\text{reg}}} \sum_{m=1}^{M_{\text{reg}}} \left| h\left(x_{t_n}^{\text{reg}}(m)\right) - y_{t_n}^{\text{reg}}(m) \right|^2 .$$

From standard properties of conditional expectations, we can write the L_2-risk of the empirical regression function and the option value as

$$\mathbb{E}_{\mathbb{Q}}\left[|m_M(X_{t_n}) - V_{t_n}(X_{t_n})|^2 \right] = \mathbb{E}_{\mathbb{Q}}\left[|m_M(X_{t_n}) - Y_{t_n}|^2 \right] - \mathbb{E}_{\mathbb{Q}}\left[|V_{t_n}(X_{t_n}) - Y_{t_n}|^2 \right].$$

This in turn can be written in terms of the so-called estimation error (first term) and the approximation error (second term), *i.e.,*

$$\begin{aligned} &\mathbb{E}_{\mathbb{Q}}[|m_M(X_{t_n}) - V_{t_n}(X_{t_n})|^2] \\ &\qquad = \left(\mathbb{E}_{\mathbb{Q}}\left[|m_M(X_{t_n}) - Y_{t_n}|^2 \right] - \min_{h \in \mathcal{A}_M} \mathbb{E}_{\mathbb{Q}}[|h_n(X_{t_n}) - Y_{t_n}|^2] \right) \\ &\qquad\quad + \left(\min_{h \in \mathcal{A}_M} \mathbb{E}_{\mathbb{Q}}[|h_n(X_{t_n}) - Y_{t_n}|^2] - \mathbb{E}_{\mathbb{Q}}\left[|V_{t_n}(X_{t_n}) - Y_{t_n}|^2 \right] \right) \end{aligned} \tag{2.27}$$

The approximation error measures how well the option value can be estimated by functions in $\mathcal{A}_M$, which corresponds to (P1) above. The estimation error is the difference between the L_2-risk of the estimator m_M and the optimal h in $\mathcal{A}_M$, which corresponds to (P2) above.

There is however, one problem with the approximation error above; we have assumed that we can sample realizations of (X, Y), while we in practice only are able to sample from $(X, \hat{Y})$, with $\hat{Y} = (\hat{Y}_t)_{t \in [t_0, t_N]}$ given by

$$\hat{Y}_{t_n} = \mathrm{e}^{-r\left(\tau_n\left(f_n^{\hat{\theta}}\right) - t_n\right)} g_{\tau_n\left(f_n^{\hat{\theta}}\right)}\left(X_{\tau_n\left(f_n^{\hat{\theta}}\right)} \right).$$

By also taking into account that the regression is carried out against an approximation of Y, (2.27) becomes instead

$$\begin{aligned} &\mathbb{E}_{\mathbb{Q}}[|m_M(X_{t_n}) - V_{t_n}(X_{t_n})|^2] \\ &\qquad \leq \left(E_{\mathbb{Q}}\left[|m_M(X_{t_n}) - \hat{Y}_{t_n}|^2 \right] - \min_{h \in \mathcal{A}_M} \mathbb{E}_{\mathbb{Q}}[|h_n(X_{t_n}) - \hat{Y}_{t_n}|^2] \right) \\ &\qquad\quad + \left(\min_{h \in \mathcal{A}_M} \mathbb{E}_{\mathbb{Q}}[|h_n(X_{t_n}) - Y_{t_n}|^2] - \mathbb{E}_{\mathbb{Q}}\left[|V_{t_n}(X_{t_n}) - Y_{t_n}|^2 \right] \right) \\ &\qquad\quad + \left(\min_{h \in \mathcal{A}_M} \mathbb{E}_{\mathbb{Q}}[|h_n(X_{t_n}) - \hat{Y}_{t_n}|^2] - \min_{h \in \mathcal{A}_M} \mathbb{E}_{\mathbb{Q}}[|h_n(X_{t_n}) - Y_{t_n}|^2] \right) \\ &\qquad\quad + \mathbb{E}_{\mathbb{Q}}[|\hat{Y}_{t_n} - Y_{t_n}|^2]. \end{aligned} \tag{2.28}$$

The first two lines in (2.28) are, again, the estimation error and the approximation error, respectively. The third line represents the difference between how well a function in $\mathcal{A}_M$ can approximate $\hat{Y}_{t_n}$ and Y_{t_n} and the final row is the L_2-risk of our approximation of the discounted cash-flow and the true discounted cash-flow. Furthermore, note that the equality in (2.27) has changed to an inequality in (2.28).

In the next section, we introduce the two different types of function classes that are used in this chapter.

2.4.2 Ordinary least squares regression

At $t_n \in \mathbb{T}$, we assume that $V_{t_n}(X_{t_n})$ can be represented by a linear combination of a countable set of $\mathcal{F}_n$-measurable basis functions. We denote by $\{\phi_b\}_{b=0}^{\infty}$ the basis functions and given optimal parameters $\alpha_{t_n}^{(1)}$, $\alpha_{t_n}^{(2)}, \dots$ (in the sense that the L_2-risk against $V_{t_n}(X_{t_n})$ is minimized) and define

$$v(t_n, X_{t_n}) = \sum_{b=1}^{\infty} \alpha_{t_n}^{(b)} \phi_b(X_{t_n}).$$

For practical purposes we use the first $B \in \mathbb{N}$ basis functions, so that

$$v_B(t_n, X_{t_n}) = \sum_{b=1}^{B} \alpha_{t_n}^{(b)} \phi_b(X_{t_n}). \tag{2.29}$$

We now want to estimate (2.29) by projecting a sample of realizations of (X_{t_n}, Y_{t_n}) onto the B first basis functions. This procedure is similar to LSM, [25]. In the LSM, only ITM samples are used, which is motivated by the fact that it is never optimal to exercise an option that is OTM and the objective is to find the optimal exercise strategy. Furthermore, the authors claim that the number of basis functions needed to obtain an accurate approximation is significantly reduced since the approximation region is reduced by only considering ITM paths. However, this is not possible in our case since we need to approximate the option everywhere[16]. On the other hand, the exercise region, $\mathcal{E}_n$, has already been approximated (as described in Subsection 2.3.1) and the option value in the exercise region is always known. This means that, similar to the LSM, the approximation region can be reduced (in many cases significantly) by only considering samples belonging to the continuation region, $\mathcal{C}_n$.

Given a sample of regression pairs $(x_{t_n}^{\text{reg}}(m), y_{t_n}^{\text{reg}}(m))_{m=1}^{M_{\text{reg}}}$, we let $M_{\text{reg}}^{\mathcal{C}_n}$ denote the number of samples belonging to $\mathcal{C}_n$ and let $(x_{t_n}^{\text{reg}}(m), y_{t_n}^{\text{reg}}(m))_{m=1}^{M_{\text{reg}}^{\mathcal{C}_n}}$ be our new samples of regression pairs (where the indexation has been appropriately changed). Assuming $M_{\text{reg}}^{\mathcal{C}_n} \geq 1$, we want to find a set of regression coefficients $\hat{\boldsymbol{\alpha}}_{\boldsymbol{t_n}} = (\hat{\alpha}_{t_n}^{(1)}, \dots, \hat{\alpha}_{t_n}^{(B)})$ such that the following empirical L_2-risk is minimized

$$\frac{1}{M_{\text{reg}}^{\mathcal{C}_n}} \sum_{m=1}^{M_{\text{reg}}^{\mathcal{C}_n}} \left| \sum_{b=1}^{B} \alpha_{t_n}^{(b)} \phi_b\left(x_{t_n}^{\text{reg}}(m)\right) - y_{t_n}^{\text{reg}}(m) \right|^2. \tag{2.30}$$

[16] By "everywhere" we mean the region in which the distribution of X_{t_n} has positive density. Of course, this is in many cases $\mathbb{R}^d$, so in practice by "everywhere" we mean the region in which the density is significantly positive.

For notational convenience, we introduce the compact notation $\boldsymbol{x_{t_n}} = \left(x_{t_n}^{\text{reg}}(1), \ldots, x_{t_n}^{\text{reg}}(M_{\text{reg}}^{\mathcal{C}_n})\right)$, $\boldsymbol{y_{t_n}} = \left(y_{t_n}^{\text{reg}}(1), \ldots, y_{t_n}^{\text{reg}}(M_{\text{reg}}^{\mathcal{C}_n})\right)$ and

$$\boldsymbol{\phi}(\boldsymbol{x_{t_n}}) = \begin{pmatrix} \phi_1\left(x_{t_n}^{\text{reg}}(1)\right) & \phi_2\left(x_{t_n}^{\text{reg}}(1)\right) & \cdots & \phi_B\left(x_{t_n}^{\text{reg}}(1)\right) \\ \phi_1\left(x_{t_n}^{\text{reg}}(2)\right) & \phi_2\left(x_{t_n}^{\text{reg}}(2)\right) & \cdots & \phi_B\left(x_{t_n}^{\text{reg}}(2)\right) \\ \vdots & \vdots & \ddots & \vdots \\ \phi_1\left(x_{t_n}^{\text{reg}}(M_{\text{reg}}^{\mathcal{C}_n})\right) & \phi_2\left(x_{t_n}^{\text{reg}}(M_{\text{reg}}^{\mathcal{C}_n})\right) & \cdots & \phi_B\left(x_{t_n}^{\text{reg}}(M_{\text{reg}}^{\mathcal{C}_n})\right) \end{pmatrix}.$$

It is a well-known fact (see *e.g.*, [45]) that $\hat{\boldsymbol{\alpha}}_{\boldsymbol{t_n}}$ is given by

$$\hat{\boldsymbol{\alpha}}_{\boldsymbol{t_n}} = \left(\boldsymbol{\phi}(\boldsymbol{x_{t_n}})^T \boldsymbol{\phi}(\boldsymbol{x_{t_n}})\right)^{-1} \boldsymbol{\phi}(\boldsymbol{x_{t_n}})^T \boldsymbol{y_{t_n}}, \tag{2.31}$$

where we note that, if we choose linearly independent basis functions, matrix inversion in (2.31) exists almost surely since X_{t_n} has a density[17]. We define the estimator

$$\hat{v}_{B,K}(t_n, \cdot) = \sum_{b=0}^{B} \hat{\alpha}_{t_n}^{(b)} \phi_b(\cdot). \tag{2.32}$$

If $M_{\text{reg}}^{\mathcal{C}_n} = 0$ we know that all samples are in the exercise region and we simply set $\hat{v}_{B,K}(\cdot) \equiv g_{t_n}(\cdot)$. Since the LSM is one of the most established algorithms for valuation of Bermudan options, the theoretical properties are extensively studied and many of the results can also be applied to the algorithm above. However, we first need to make an assumption regarding $M_{\text{reg}}^{\mathcal{C}_n}$. Assume that there exists $c > 0$ such that $\mathbb{Q}\{X_{t_n} \in \mathcal{C}_n\} \geq c$. It then holds for any $C \in \mathbb{R}$ that

$$\mathbb{Q}\left\{\lim_{M_{\text{reg}} \to \infty} \sum_{m=1}^{M_{\text{reg}}} \mathbb{I}_{\{X_{t_n}^{\text{reg}}(m) \in \mathcal{C}_n\}} \geq C\right\} = 1,$$

which implies that $M_{\text{reg}}^{\mathcal{C}_n}$ approaches infinity when M_{reg} approaches infinity almost surely. Since the regression pairs are independently and identically distributed, it holds that $\hat{v}_{B,K}(t_n, X_{t_n})$ converges both in mean square and in probability to $v_B(t_n, X_{t_n})$ as M_{reg} approaches infinity (see *e.g.*, [46]). To make it more clear when comparing the OLS-approximator of the option value to the neural network approximator (to be defined in the next section), we use the following notation

$$\hat{v}_{t_n}^{\text{OLS}}(\cdot) = \hat{v}_{B,M}(t_n, \cdot), \tag{2.33}$$

where we assume that B and M are chosen such that both accuracy and time complexity are taken into account.

A nice property of OLS regression is that, given B and a sample of regression pairs, we have a closed-form expression for the optimal parameters and thus also the regression function (2.32). On the other hand, we may face memory or runtime issues for large B and $M_{\text{reg}}^{\mathcal{C}_n}$ due to (2.31). This is a problem, especially when we want to approximate a complicated function surface over a large approximation region. For example, consider an option based on 50 correlated assets. If we want to use the first and second-order polynomials as basis functions (including

[17] In practice we run into troubles if we choose B too high since the approximation of the matrix inverse may be unstable.

cross-terms) we have $B = \frac{50(50+3)}{2} = 1325$, which is often too large for practical purposes. We should also have in mind that this corresponds to an approximation with polynomials of degree 2, which is usually not sufficient for complicated problems. There are however methods to get around this problem, see *e.g.*, [28] in which the state space is divided into several bundles and regression is carried out locally at each bundle. Another suitable method to overcome these difficulties is neural network regression, which is presented in the next section.

2.4.3 Neural network regression

In this section we present, a simple neural network approximation of $V_{t_n}(\cdot)$. The neural network is a mapping, $v_{\varphi_n} : \mathbb{R}^d \to \mathbb{R}$, parametrized by the $p_{t_n} \in \mathbb{N}$ trainable parameters $\varphi_n \in \mathbb{R}^{p_{t_n}}$. The objective is to find φ_n such that the empirical L_2-risk

$$\frac{1}{M_{\text{reg}}^{\mathcal{C}_n}} \sum_{m=1}^{M_{\text{reg}}^{\mathcal{C}_n}} \left| v_{\varphi_n}(x_{t_n}^{\text{reg}}(m)) - y_{t_n}^{\text{reg}}(m) \right|^2 \tag{2.34}$$

is minimized. We denote by $\hat{\varphi}_n$ an optimized version of φ_n and define our neural network approximator of the option price at t_n by

$$\hat{v}_{t_n}^{\text{NN}}(\cdot) = v_{\hat{\varphi}_n}(\cdot).$$

To avoid repetition, the description of the neural networks in Subsection 2.3.1.1 is also valid for the neural network used here. However, one important difference is the output, which in this section is an approximation of the option value, and should hence take on values in $(0, \infty)$. A natural choice as activation function in the output layer is therefore $\text{ReLU}(\cdot) = \max\{0, \cdot\}$. Furthermore, by shifting the output with $-g_{t_n}(\cdot)$, *i.e.,* designing the neural network to output $v_{\hat{\varphi}_n}(\cdot) - g_{t_n}(\cdot)$ and defining $\hat{v}_{t_n}^{\text{NN}}(\cdot) = v_{\hat{\varphi}_n}(\cdot) + g_{t_n}(\cdot)$ we can, for all $x \in \mathbb{R}^d$, enforce $\hat{v}^{\text{NN}}(x) \geq g_{t_n}(x)$ by using ReLU as activation function in the output layer. In many cases it seems to be beneficial to use the identity as the activation function in the output layer. This could possibly be explained by the fact that when using the ReLU as activation function, the gradient of the loss function, (2.34) (with respect to the input) may vanish during training. This, in turn leads to an inefficient use of a gradient descent type algorithm in the optimization problem.

Another difference, which has to do with the training phase, is that the optimization of the parameters φ_n does not have to be carried out recursively. This opens up the possibility for parallelization of the code.

By comparing (2.34) to (2.30) we see that the optimization problems are similar. There are, however, some major differences. In Subsection 2.4.2, we have a closed-form expression for the optimal parameters resulting in the final regression function (2.32). This is not the case for the neural network regression and we therefore need to use an optimization algorithm to approximate the optimal parameters. On the other hand, as mentioned in Subsection 2.4.2 it is sometimes hard to find basis functions that are flexible enough. This problem can be overcome with neural networks, which are known to be good global approximators.

2.5 Approximation algorithms for exposure profiles

In this section, we introduce different ways to estimate (2.9) and (2.10) relying on Monte-Carlo sampling and the approximation algorithms described in Sections 2.3 and 2.4. Furthermore, a simple example is presented and visualized, which aims to provide an intuitive understanding of the different methods. Finally, the advantages and disadvantages of each method are presented in a table.

In this section, the neural network-based approximation of the value function of the option, introduced in Subsection 2.4.3 is used. However, it would have been possible to use the OLS-based approximation from Subsection 2.4.2, instead.

We use $M \in \mathbb{N}$ independent realizations of X, which for $m \in \{1, 2, \ldots, M\}$ are denoted by $x(m) = (x_t(m))_{n=0}^N$ for $t \in \mathbb{T}$. When X is given by (2.19), realization m is denoted by $\tilde{x}^{t_n}(m) = \left(\tilde{x}_t^{t_n}(m)\right)_{t\in[t_0,t_N]}$, where we recall that superscript t_n refers to the point in time where the discontinuity of the drift coefficient is located. We introduce the following two approximations of the expected exposure under the risk-neutral measure

$$\hat{\mathrm{EE}}^1_{\mathbb{Q}}(t_n) = \frac{1}{M}\sum_{m=1}^{M} \hat{v}_{t_n}^{\mathrm{NN}}(x_{t_n}(m))\mathbb{I}_{\left\{\tau\left(\boldsymbol{f}^{\hat{\boldsymbol{\theta}}}\right)>t_n\right\}}, \tag{2.35}$$

$$\hat{\mathrm{EE}}^2_{\mathbb{Q}}(t_n) = \frac{1}{M}\sum_{m=1}^{M} \mathrm{e}^{-r\left(\tau_n\left(\boldsymbol{f}_n^{\hat{\boldsymbol{\theta}}}\right)-t_n\right)} g_{\tau_n\left(\boldsymbol{f}_n^{\hat{\boldsymbol{\theta}}}\right)}\left(x_{\tau_n\left(\boldsymbol{f}_n^{\hat{\boldsymbol{\theta}}}\right)}(m)\right)\mathbb{I}_{\left\{\tau\left(\boldsymbol{f}^{\hat{\boldsymbol{\theta}}}\right)>t_n\right\}}. \tag{2.36}$$

For the expected exposure under the real-world measure, we have the following three approximations

$$\hat{\mathrm{EE}}^1_{\mathbb{P}}(t_n) = \frac{1}{M}\sum_{m=1}^{M} \hat{v}_{t_n}^{\mathrm{NN}}\left(\tilde{x}_{t_n}^{t_n}(m)\right)\mathbb{I}_{\left\{\tau\left(\boldsymbol{f}^{\hat{\boldsymbol{\theta}}}\right)>t_n\right\}}, \tag{2.37}$$

$$\hat{\mathrm{EE}}^2_{\mathbb{P}}(t_n) = \frac{1}{M}\sum_{m=1}^{M} \mathrm{e}^{-r\left(\tau_n\left(\boldsymbol{f}_n^{\hat{\boldsymbol{\theta}}}\right)-t_n\right)} g_{\tau_n\left(\boldsymbol{f}_n^{\hat{\boldsymbol{\theta}}}\right)}\left(\tilde{x}^{t_n}_{\tau_n\left(\boldsymbol{f}_n^{\hat{\boldsymbol{\theta}}}\right)}(m)\right)\mathbb{I}_{\left\{\tau\left(\boldsymbol{f}^{\hat{\boldsymbol{\theta}}}\right)>t_n\right\}}, \tag{2.38}$$

$$\hat{\mathrm{EE}}^3_{\mathbb{P}}(t_n) = \frac{1}{M}\sum_{m=1}^{M} \mathrm{e}^{-r\left(\tau_n\left(\boldsymbol{f}_n^{\hat{\boldsymbol{\theta}}}\right)-t_n\right)} g_{\tau_n\left(\boldsymbol{f}_n^{\hat{\boldsymbol{\theta}}}\right)}\left(x_{\tau_n\left(\boldsymbol{f}_n^{\hat{\boldsymbol{\theta}}}\right)}(m)\right)\mathbb{I}_{\left\{\tau\left(\boldsymbol{f}^{\hat{\boldsymbol{\theta}}}\right)>t_n\right\}} l\left(x_{t_n}(m), \ldots, x_{t_0}(m)\right), \tag{2.39}$$

where l is the likelihood ratio function defined in Theorem 2. We note that (2.35) and (2.37) are the only approximations that require a calculation of the option value. On the other hand, we need X to be described by a diffusion-type SDE in (2.38) and we need to know the density functions to calculate (2.39). To define the approximations of the potential future exposure, we start by defining the order statistic of $\left(\hat{v}_{t_n}^{\mathrm{NN}}(x_{t_n}(m))\mathbb{I}_{\left\{\tau\left(\boldsymbol{f}^{\hat{\boldsymbol{\theta}}}\right)>t_n\right\}}\right)_{m=1}^{M}$, *i.e.*, the vector given by $\left(\hat{v}_{t_n}^{\mathrm{NN}}(x_{t_n}(\tilde{m}_1))\mathbb{I}_{\left\{\tau\left(\boldsymbol{f}^{\hat{\boldsymbol{\theta}}}\right)>t_n\right\}}, \ldots, \hat{v}_{t_n}^{\mathrm{NN}}(x_{t_n}(\tilde{m}_M))\mathbb{I}_{\left\{\tau\left(\boldsymbol{f}^{\hat{\boldsymbol{\theta}}}\right)>t_n\right\}}\right)$ satisfying

Specification of requirements for each approximation				
	Approximation of the functional form $V_{t_n}(\cdot)$	Sampling from $\tilde{X}^{t_n}$	Known density functions	X given must be by diffusion-type SDE
$\hat{\mathrm{EE}}^1_{\mathbb{Q}}$	✔	✖	✖	✖
$\hat{\mathrm{EE}}^2_{\mathbb{Q}}$	✖	✖	✖	✖
$\hat{\mathrm{EE}}^1_{\mathbb{P}}$	✔	✔	✖	✖
$\hat{\mathrm{EE}}^2_{\mathbb{P}}$	✖	✔	✖	✔
$\hat{\mathrm{EE}}^3_{\mathbb{P}}$	✖	✖	✔	✖
$\hat{\mathrm{PFE}}_{\mathbb{Q}}$	✔	✖	✖	✖
$\hat{\mathrm{PFE}}_{\mathbb{P}}$	✔	✔	✖	✖

Table 2.1: A specification of which approximations/entities that are required in order to carry out the different calculations. If required ✔, otherwise ✖.

$\hat{v}^{\mathrm{NN}}_{t_n}(x_{t_n}(\tilde{m}_i)) \leq \hat{v}^{\mathrm{NN}}_{t_n}(x_{t_n}(\tilde{m}_j))$ whenever $i \leq j$. Furthermore, we define

$$i_\alpha = \begin{cases} \lceil \alpha M \rceil, & \text{for } \alpha \geq 0.5, \\ \lfloor \alpha M \rfloor, & \text{for } \alpha < 0.5. \end{cases}$$

The approximations of the potential future exposure are then defined as

$$\hat{\mathrm{PFE}}^{\alpha}_{\mathbb{Q}}(t_n) = \hat{v}^{\mathrm{NN}}_{t_n}(x_{t_n}(\tilde{m}_{i_\alpha}))\mathbb{I}_{\left\{\tau\left(\boldsymbol{f}^{\hat{\boldsymbol{\theta}}}\right) > t_n\right\}},$$
$$\hat{\mathrm{PFE}}^{\alpha}_{\mathbb{P}}(t_n) = \hat{v}^{\mathrm{NN}}_{t_n}(\tilde{x}^{t_n}_{t_n}(\tilde{m}_{i_\alpha}))\mathbb{I}_{\left\{\tau\left(\boldsymbol{f}^{\hat{\boldsymbol{\theta}}}\right) > t_n\right\}}.$$

In Table 2.1, some characteristics of the calculations needed for each approximation are given.

To explain the different approximations in a concrete setting we turn to a simple example.

Example 2.5.1. *Consider a one-dimensional American put option, where we for simplicity assume that $r = 0$. We are interested in the expected exposure and the potential future exposure at time $t_n \in (t_0, t_N)$ given that we have full knowledge of the market at time t_0. We give a short explanation of the intuition behind the different methods by referring to Figure 2.2, where the problem is visualized.*

We start by $\hat{\mathrm{EE}}^1_{\mathbb{Q}}(t_n)$ and $\hat{\mathrm{EE}}^1_{\mathbb{P}}(t_n)$ for which we only use the figure to the left. For $\hat{\mathrm{EE}}^1_{\mathbb{Q}}(t_n)$ we follow the blue samples and note that samples 2 and 3 are not exercised prior to, or at t_n, which means that the indicator function in (2.35) *equals 1. Sample 1, on the other hand, touches the exercise region prior to t_n and has therefore already been exercised, which means that*

$$\begin{aligned}\hat{\mathrm{EE}}^1_{\mathbb{Q}}(t_n) &= \frac{1}{3}\left(\hat{v}^{\mathrm{NN}}(x_{t_n}(1))\mathbb{I}_{\{\tau(\boldsymbol{f}^{\hat{\boldsymbol{\theta}}})>t_n\}} + \hat{v}^{\mathrm{NN}}(x_{t_n}(2))\mathbb{I}_{\{\tau(\boldsymbol{f}^{\hat{\boldsymbol{\theta}}})>t_n\}} + \hat{v}^{\mathrm{NN}}(x_{t_n}(3))\mathbb{I}_{\{\tau(\boldsymbol{f}^{\hat{\boldsymbol{\theta}}})>t_n\}}\right) \\ &= \frac{1}{3}\left(\hat{v}^{\mathrm{NN}}(x_{t_n}(2)) + \hat{v}^{\mathrm{NN}}(x_{t_n}(3))\right).\end{aligned}$$

When focusing on the red samples instead, we see that no sample touches the exercise region prior to t_n *and we obtain*

$$\hat{\mathrm{EE}}^1_{\mathbb{P}}(t_n) = \frac{1}{3}\left(\hat{v}^{\mathrm{NN}}(\tilde{x}^{t_n}_{t_n}(1)) + \hat{v}^{\mathrm{NN}}(\tilde{x}^{t_n}_{t_n}(2)) + \hat{v}^{\mathrm{NN}}(\tilde{x}^{t_n}_{t_n}(3))\right).$$

Similarly, we can e.g., state that $\hat{\mathrm{PFE}}^{2.5}_{\mathbb{Q}} = 0$ *and* $\hat{\mathrm{PFE}}^{97.5}_{\mathbb{P}} = \hat{v}^{\mathrm{NN}}(\tilde{x}^{t_n}_{t_n}(2))$.

Moving on to $\hat{\mathrm{EE}}^2_{\mathbb{P}}(t_n)$ *and* $\hat{\mathrm{EE}}^3_{\mathbb{P}}(t_n)$*, we shift focus to the figure to the right. For* $\hat{\mathrm{EE}}^2_{\mathbb{P}}(t_n)$ *we want to use the red samples and notice that samples 2 and 3 end up out of the money. We therefore obtain*

$$\hat{\mathrm{EE}}^2_{\mathbb{P}}(t_n) = \frac{1}{3} g_{\tau_n(\boldsymbol{f}^{\boldsymbol{\theta}}_n)}\left(\tilde{x}^{t_n}_{\tau_n(\boldsymbol{f}^{\boldsymbol{\theta}}_n)}(1)\right).$$

For $\hat{\mathrm{EE}}^3_{\mathbb{P}}(t_n)$*, we instead consider the blue samples and see that sample 1 is exercised prior to* t_n *and samples 2 and 3 end up in the money. However, we also need to adjust the estimate for using the wrong state process*[18]*. This is done by multiplying each term with the likelihood ratios* $l(x(2))$ *and* $l(x(3))$ *to finally obtain*

$$\hat{\mathrm{EE}}^3_{\mathbb{P}}(t_n) = \frac{1}{3}\left(g_{\tau_n(\boldsymbol{f}^{\boldsymbol{\theta}}_n)}\left(x_{\tau_n(\boldsymbol{f}^{\boldsymbol{\theta}}_n)}(2)\right) l\left(x(2)\right) + g_{\tau_n(\boldsymbol{f}^{\boldsymbol{\theta}}_n)}\left(x_{\tau_n(\boldsymbol{f}^{\boldsymbol{\theta}}_n)}(3)\right) l\left(x(3)\right)\right).$$

The last estimate, $\hat{\mathrm{EE}}^2_{\mathbb{Q}}(t_n)$*, is obtained by removing the likelihood ratios from the estimate for* $\hat{\mathrm{EE}}^3_{\mathbb{P}}(t_n)$.

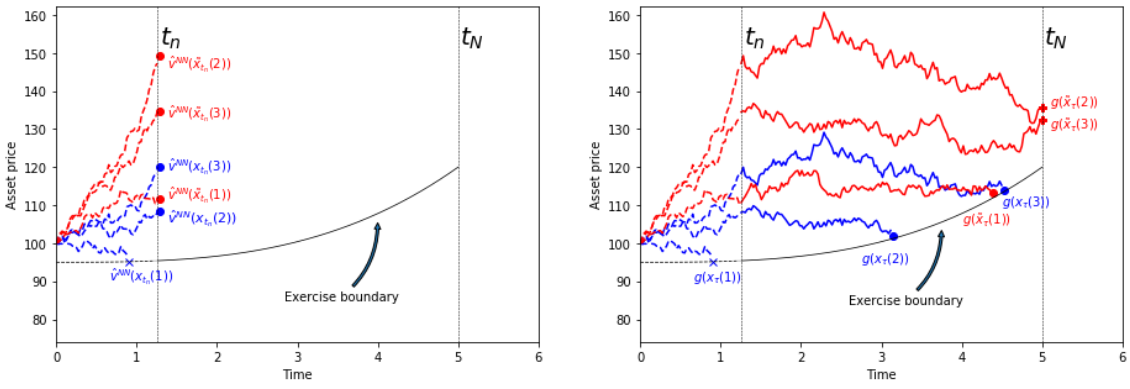

Figure 2.2: Blue trajectories are distributed as X and red trajectories are distributed as $\tilde{X}^{t_n}$. The boundary for immediate exercise is pointed out and should be interpreted as, as soon a trajectory touches the boundary, the option is exercised and the holder receives the immediate pay-off. Recall that the exercise boundaries are calculated in order to be optimal under the $\mathbb{Q}$-measure.

To conclude, we note that Figure 2.2 displays, to the left, the cases where functional form approximations of the option values are used and to the right the cases where cash-flow paths are used (this can be read in Table 2.1). Furthermore, blue and red trajectories are distributed according to X and $\tilde{X}^{t_n}$, respectively (this can also be read in Table 2.1).

[18]The samples are generated from the state process under the $\mathbb{Q}-$measure between t_0 and t_n which, if not corrected, would be in conflict with the definition of $\mathrm{EE}_{\mathbb{P}}(t_n)$.

2.6 Numerical results

This section is divided into two parts: in the first part we use the Black–Scholes dynamics for the underlying assets. The proposed algorithm is compared with two different Monte-Carlo-based algorithms for a two-dimensional case. We focus on both the accuracy of the computed exercise boundaries and the exposure profiles. Furthermore, the exercise boundary is compared with the exercise boundary for the corresponding American option, computed by a finite element method, from the PDE-formulation of the problem. Exposure profiles are then computed both under the risk-neutral and the real-world measures for problems in 2 and 30 dimensions. Finally, we compare the OLS-regression with the NN-regression.

In the second part we consider stochastic volatility and compute exposure profiles under the Heston model.

2.6.1 Black–Scholes dynamics

In the Black–Scholes setting the only risk factors are the $d \in \mathbb{N}$, underlying assets, denoted by S. We assume a constant risk-free interest rate $r \in \mathbb{R}$, and for each asset $i \in \{1, 2, \ldots, d\}$, volatility $\sigma_i \in (0, \infty)$, and constant dividend rate $q_i \in (0, \infty)$. The state process X is then given by the asset process $S = (S_t)_{t\in[t_0,t_N]}$, *i.e.,* $X = S$, following the SDE

$$\frac{\mathrm{d}(S_t)_i}{(S_t)_i} = (A_i - q_i)\mathrm{d}t + \sigma_i \mathrm{d}(W_t^{\mathbb{A}})_i,\ (S_{t_0})_i = (s_{t_0})_i; \quad t \in [t_0,\, t_N], \tag{2.40}$$

with initial state $(s_{t_0})_i \in (0, \infty)$, and where $\mathbb{A}$ is either the real-world measure $\mathbb{P}$ or the risk-neutral measure $\mathbb{Q}$. In the real-world setting $A_i = \mu_i \in \mathbb{R}$ and in the risk-neutral setting $A_i = r$. The process $W^{\mathbb{A}} = (W_t^{\mathbb{A}})_{t\in[t_0,t_n]}$ is a d-dimensional Brownian motion under the measure $\mathbb{A}$. Furthermore, $W^{\mathbb{A}}$ is correlated with correlation parameters $\rho_{ij} \in [-1, 1]$, *i.e.,* for $i, j \in \{1, 2, \ldots, d\}$, we have that $\mathbb{E}^{\mathbb{A}}[\mathrm{d}(W_t^{\mathbb{A}})_i \mathrm{d}(W_t^{\mathbb{A}})_j] = \rho_{ij}\mathrm{d}t$, with $\rho_{ii} = 1$. Moreover, for $t_0 \leq u \leq t \leq t_N$ a closed-form solution to (2.40) is given by

$$(S_t)_i = (S_u)_i \exp\left((A_i - q_i - \frac{1}{2}\sigma_i^2)(t-u) + \sigma_i\left((W_t^{\mathbb{A}})_i - (W_u^{\mathbb{A}})_i\right)\right). \tag{2.41}$$

We note that $\log S$ has Gaussian increments under both the $\mathbb{P}-$ and the $\mathbb{Q}-$measures which implies that we have access to a closed-form expressions for the transition densities of S, and in turn also to the likelihood ratio in (2.39).

2.6.1.1 Bermudan max-call option

At exercise, a Bermudan max-call option pays the positive part of the maximum of the underlying assets after subtraction of a fixed amount $K \in \mathbb{R}$. This implies an identical pay-off function at all exercise dates, given by

$$g(s) = \left(\max\{s_1,\, s_2\, \ldots,\, s_d\} - K\right)^+,$$

where $s = (s_1, s_2, \ldots, s_d) \in (0, \infty)^d$ and for $c \in \mathbb{R}$, $(c)^+ = \max\{c,\, 0\}$.

We choose to focus on Bermudan max-call options for two reasons: first, there exist plenty of comparable results in the literature (see *e.g.,* [9], [27], [28]); second, and more importantly, the exercise region is nontrivial with several interesting features. For example, $\max\{s_1, s_2, \ldots, s_d\}$ is not a sufficient statistic for making optimal exercise decisions, meaning that we cannot easily reduce the dimensionality (all dimensions are needed in order to price the option). This is not the case with, *e.g.,* the geometric-average call option with pay-off function $g(s) = \left(\frac{1}{d}\prod_{i=1}^{d} s_i - K\right)^+$, since $\frac{1}{d}\prod_{i=1}^{d} s_i$ is a sufficient statistic for optimal exercise decisions (only the geometric average is needed to price the option, meaning that the problem can be reduced to 1 dimension, see *e.g.,* [47]). Another example is the arithmetic-average call option with pay-off function $g(s) = \left(\frac{1}{d}\sum_{i=1}^{d} s_i - K\right)^+$. Similar to the max-call option, $\frac{1}{d}\sum_{i=1}^{d} s_i$ is not a sufficient statistic for optimal exercise decisions, but on the other hand the exercise region is convex[19,20] (see [43, Proposition 6.1]). Convexity of the exercise region does not hold for the max-call option, making it hard to capture the exercise region when global polynomials are used as basis functions in *e.g.,* the LSM. Methods which instead rely on local regression can, to some extent, overcome this problem but it is difficult to decide how the localization (usually localization of the state space) should be done, especially in high dimensions.

In the numerical experiments we use the following parameters $r = 0.05$ and for $i, j \in \{1, 2, \ldots, d\}$, $q_i = 0.1$, $\sigma_i = 0.2$, for $i \neq j$, $\rho_{ij} = 0$, $N = 9$, $t_0 = 0$, $t_N = 3$, $(s_{t_0})_i = 100$ and $K = 100$. We want to emphasize that the choice of having no correlation between assets is due to the fact that this case has been studied thoroughly in the literature (see *e.g.,* [9], [27], [28], [43]). To verify that the algorithm is also able to tackle problems with correlated assets, we replicated an experiment in [32], of pricing a put option on the arithmetic average of 5 correlated assets and obtained the same price[21].

2.6.1.2 Approximation of the option value at initial time

The performance of the DOS algorithm is thoroughly explored for a wide range of different examples in [9]. However, the convergence with respect to the amount of training data used, M_{train} was not given. We therefore present, in Figure 2.3, an example of how the option price at t_0 is converging to a reference value in terms of the amount of training data. In the considered example, we use the parameter values given at the end of Subsection 2.6.1.1 with $d = 2$, *i.e.,* a two-dimensional Bermudan max-call option. The reference value (13.902) for $V_{t_0}(x_{t_0})$ is computed by a binomial lattice model in [24]. The DOS approximation, denoted by $\hat{V}_{t_0}(x_{t_0})$, is computed according to (2.25). To be more specific, one neural network is trained for each $M_{\text{train}} \in \{2^{12}, 2^{14}, 2^{16}, 2^{18}, 2^{20}, 2^{22}, 2^{25}\}$, as described in Subsection 2.3.1.2. For each M_{train}, the option value, $V_{t_0}(x_{t_0})$ is computed 20 times (with 20 independent realizations of X) with $M_{\text{val}} = 2^{20}$ and the average of these 20 values is showed in Figure 2.3. Furthermore, the figure to the right displays empirical 95%-confidence intervals of the sample mean, which is computed by adding and subtracting $\frac{1.96}{\sqrt{19}}$ times the sample standard deviation.

[19] In this section we have assumed that results for exercise regions for American options also hold for their Bermudan counterparts.

[20] Convex in the underlying assets for fixed t.

[21] We obtained the price 0.1804, which is the same price up to the given accuracy of 4 digits, presented in [32, Parameters as in Set II, results in Table 3 on p. 22].

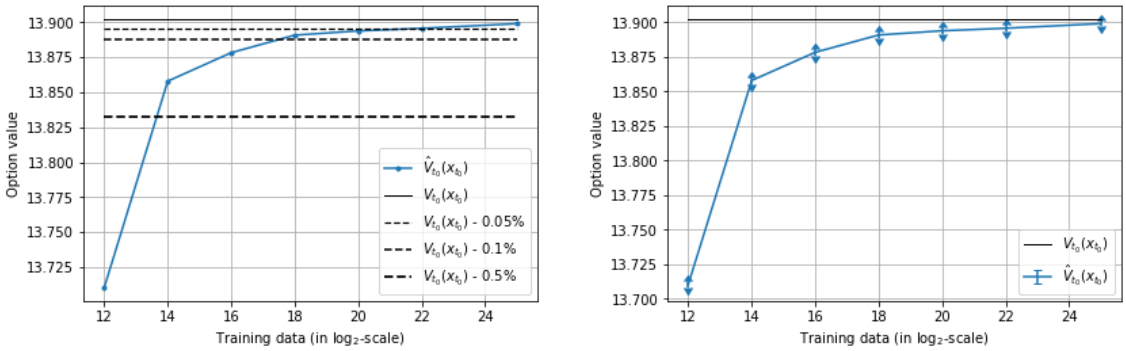

Figure 2.3: Convergence of approximate option values in the amount of training data. Reference value 13.902, computed by a binomial lattice model in [24]. **Left:** With different levels of deviation from the reference value. **Right:** With empirical 95%-confidence intervals of sample mean.

2.6.1.3 Comparison with Monte-Carlo-based algorithms

We start with a short introduction of the two Monte-Carlo-based algorithms with which we compare results for the two-dimensional max-call option.

The least squares method (LSM) proposed in [25], is one of the most used methods for pricing American/Bermudan options. The method approximates exercise boundaries and computes the option value from discounted cash-flows. The regression is carried out globally, *i.e.,* with the same regression function over the entire state space. However, if one is only interested in the option value at t_0, it is beneficial to only consider ITM-samples in the regression state. This is done when the exercise boundary, and $\text{PFE}_{97.5}$ are approximated. For approximating EE and $\text{PFE}_{2.5}$ we need the entire distribution of future option values forcing us to also include OTM-samples in the regression. We use as basis functions the first 6 Laguerre polynomials $L_n(x) = \mathrm{e}^{-x/2}\frac{\mathrm{e}^x}{n!}\frac{\mathrm{d}^n}{\mathrm{d}x^n}(x^n\mathrm{e}^{-x})$ of $(S_{t_n})_1$ and $(S_{t_n})_2$ and the first 4 powers of $\max\{\log((S_{t_n})_1), \log((S_{t_n})_2)\}$ for all $t_n \in \mathbb{T}$. Note that since we have no correlation between $(S)_1$ and $(S)_2$, we do not include cross-terms in the basis.

The second algorithm is the stochastic grid bundling method (SGBM) proposed in [28], in which regression is carried out locally in so-called bundles. In the SGBM the target function in the regression phase is the option value which makes it suitable for approximations of exposure profiles. There are several papers on computations of exposure profiles with the SGBM, see *e.g.,* [35]. At each exercise date $t_n \in \mathbb{T}$, we use as basis functions: a constant, the first 4 powers of $(S_{t_n})_1$ and $(S_{t_n})_2$, and the first 2 powers of $\max\{\log((S_{t_n})_1), \log((S_{t_n})_2)\}$. Furthermore, the state space is divided into 32 equally-sized bundles based on $\max\{(S_{t_n})_1, (S_{t_n})_2\}$.

Before presenting any results, we recall from equations (2.9) and (2.10) that the expected exposure and the potential future exposure are two statistics of

$$V_{t_n}(S_{t_n})\mathbb{I}_{\{\tau > t_n\}}, \tag{2.42}$$

where τ is an $S-$stopping time. This means that approximations of the exposure profiles are sensitive to, not only the option value itself, but also the exercise strategy. The SGBM and LSM only compute the exercise strategy implicitly, *i.e.,* by comparing the approximate option value and the immediate pay-off. Therefore small errors of the approximate option

value close to the exercise boundary can lead to significant errors in the exercise strategy[22]. We therefore start by presenting a comparison of the exercise boundaries. As a reference, we use the exercise boundary for the corresponding American option, which is computed from the PDE-formulation with the finite element method used in [48]. We note that since the PDE formulation refers to the American option, the exercise boundary differs slightly[23] from the exercise boundary of the Bermudan counterpart, which we are interested in.

In Figure 2.4, a comparison of the exercise boundaries at $t_8 \approx 2.67$ for the different algorithms is presented. As we can see, the DOS algorithm captures the shape of the exercise regions while both the SGBM and the LSM seem to struggle, especially with the part of the continuation region along (and around) the line $(S_{t_n})_1 = (S_{t_n})_2$, in particular for high values of $(S_{t_n})_1$ and $(S_{t_n})_2$. The irregular shaped features in the continuation region for the SGBM are a consequence of the local regression. In particular, the triangular shapes come from the specific choice of bundling rule (bundling based on $\max\{(S_{t_n})_1, (S_{t_n})_2\}$).

Moving on to the exposure profiles, we see in Figure 2.5, that even though the DOS algorithm and the SGBM seem to agree on the exposure profile in general, we notice a difference in the $\text{PFE}_{97.5}$. This is a consequence of the slight bias towards classifying samples as belonging to the continuation region, which is shown in Figure 2.5, to the right.

The LSM is performing worse, both in terms of accuracy of exposure profiles and bias towards miss-classification. This is however not a surprise, since the LSM is tailored to calculate the option value at t_0.

Finally, it should be pointed out that for both the SGBM and LSM, it could very well be the case that another set of basis functions would better capture the shape of the exercise boundaries. In this two-dimensional example one could probably use geometric intuition to come up with a better set of basis functions, but in higher dimensions, and for more complicated pay-off functions, this becomes difficult.

2.6.1.4 Exposure profiles under different measures

In this section we compare exposure profiles under different measures for the max-call option, in 2 and 30 dimensions. In *Case I* we set $d = 2$ and $\mathbb{P}^1$ and $\mathbb{P}^2$ such that for $i \in \{1, 2\}$, we have drifts $(\mu_1)_i = 15\%$ and $(\mu_2)_i = -5\%$. In *Case II* we set $d = 30$ and $\mathbb{P}^1$ and $\mathbb{P}^2$ such that such that for $i \in \{1, \ldots, 30\}$, we have drifts $(\mu_1)_i = 7.5\%$ and $(\mu_2)_i = 2.5\%$.

Figure 2.6 shows exposure profiles in 2 and 30 dimensions on the left side. On the right, we see a comparison of the different ways to compute expected exposures which all agree to high accuracy. Furthermore, Figure 2.7, displays that the fraction of exercised options over time is highly dependent on the choice of measure.

[22] In our experiments the errors of the approximate option values seem be of the same sign locally *i.e.,* the polynomial basis function underestimates the option value in some regions and overestimates the option value in other regions.

[23] In fact, the continuation region for an American option is a subset of the continuation region for the Bermudan counterpart.

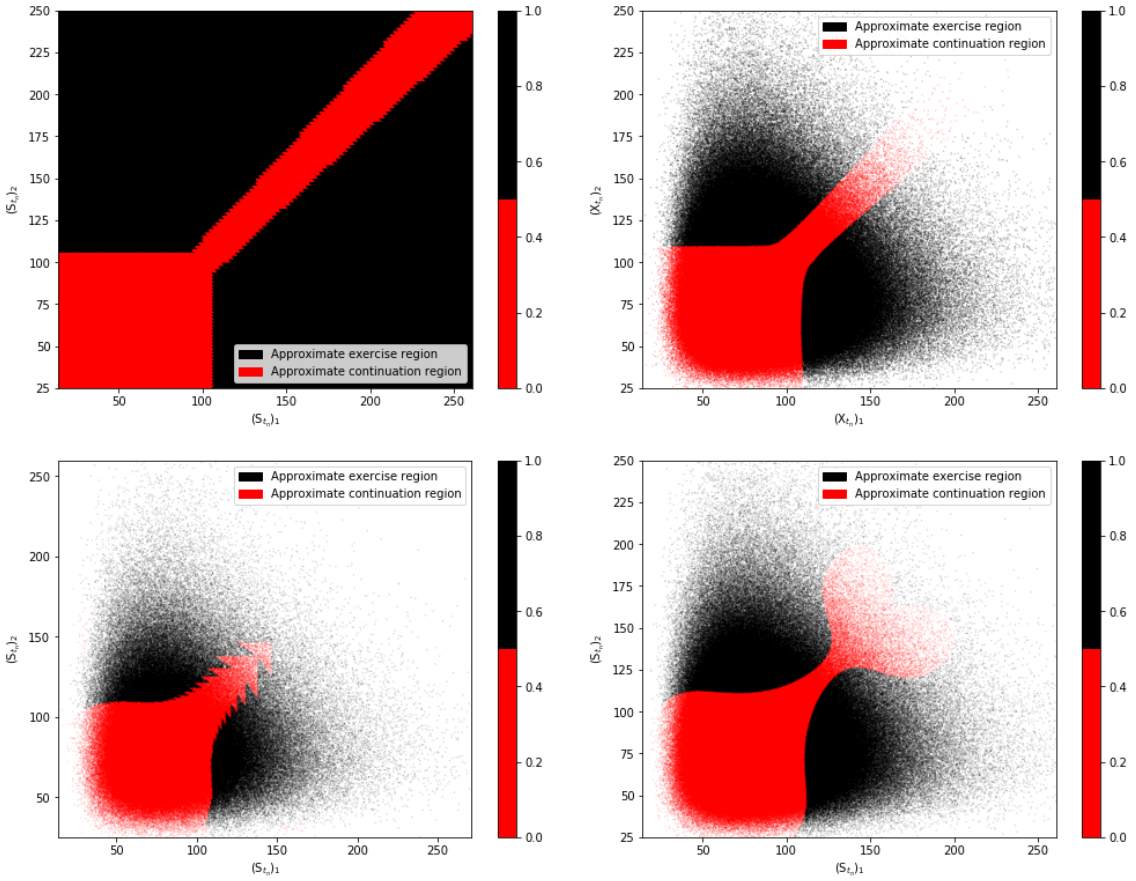

Figure 2.4: Approximate exercise boundaries for a two-dimensional max-call option at $t_8 \approx 2.67$. **From top left to bottom right:** FEM (American option), DOS, SGBM and LSM respectively.

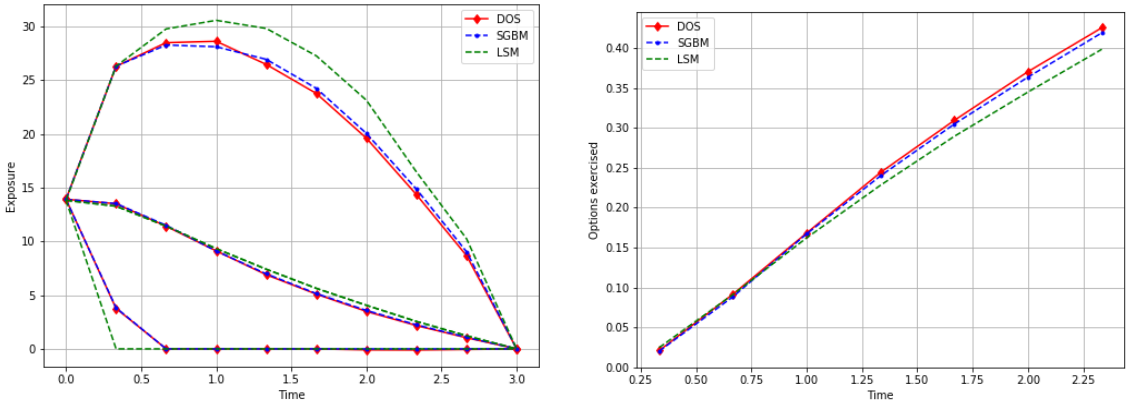

Figure 2.5: Comparison of DOS, SGBM and LSM for a two-dimensional max-call option. **Left:** Expected exposure and potential future exposures at 97.5%- and 2.5%-levels. **Right:** Proportion of options exercised at different exercise dates.

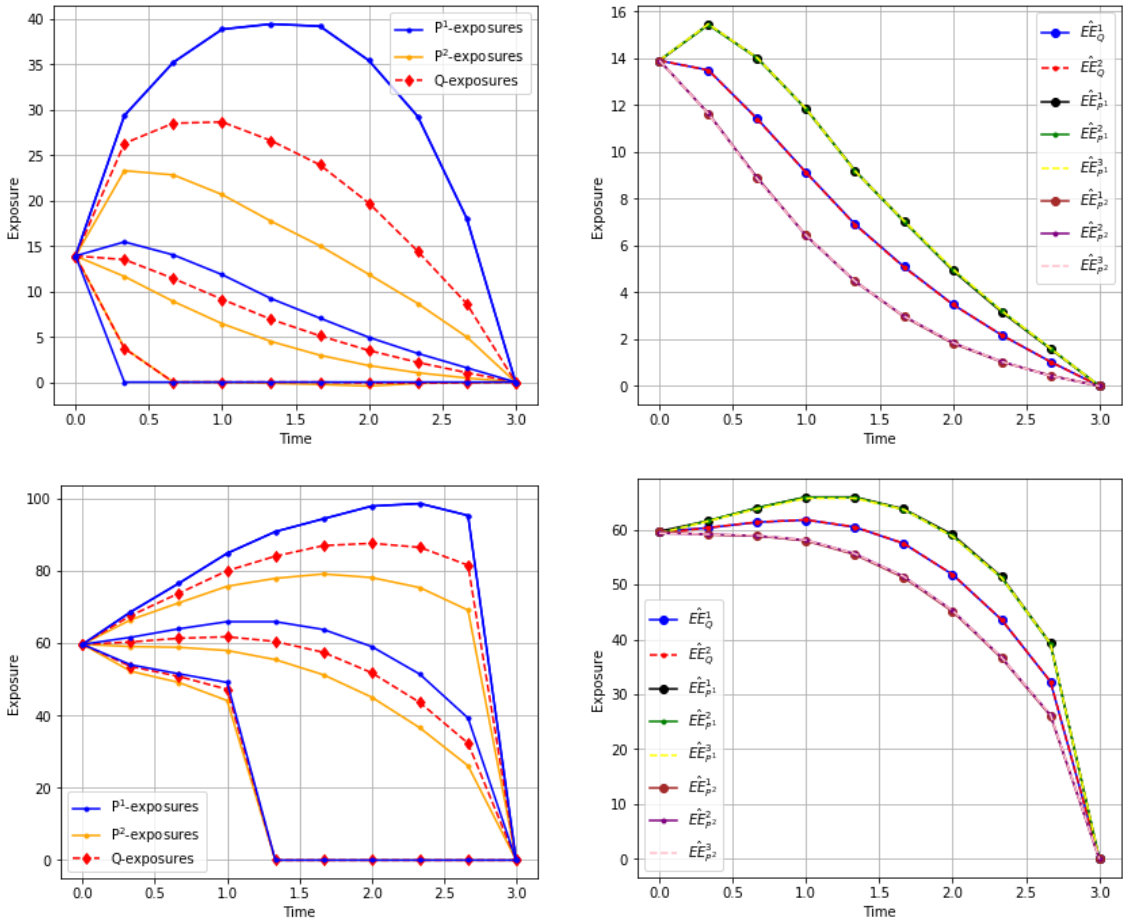

Figure 2.6: Approximate exposures, at the exercise dates over the life time of the contract. **Top:** Case I. **Bottom:** Case II. **Left:** For $i = 1$ and $i = 2$, $\hat{\text{PFE}}^{97.5}_{\mathbb{P}^i}$, $\hat{\text{PFE}}^{97.5}_{\mathbb{Q}}$, $\hat{\text{EE}}^{1}_{\mathbb{P}^i}$, $\hat{\text{EE}}^{1}_{\mathbb{Q}}$ $\hat{\text{PFE}}^{2.5}_{\mathbb{P}^i}$ and $\hat{\text{PFE}}^{2.5}_{\mathbb{Q}}$. **Right:** For $i = 1$ and $i = 2$, $\hat{\text{EE}}^{1}_{\mathbb{Q}}$, $\hat{\text{EE}}^{2}_{\mathbb{Q}}$, $\hat{\text{EE}}^{1}_{\mathbb{P}^i}$, $\hat{\text{EE}}^{2}_{\mathbb{P}^i}$ and $\hat{\text{EE}}^{3}_{\mathbb{P}^i}$.

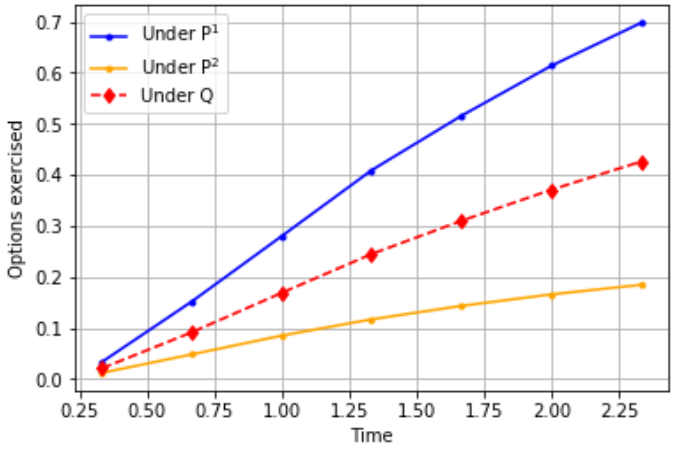

Figure 2.7: Proportion of samples exercised at different exercise dates under measures $\mathbb{Q}$, $\mathbb{P}^1$ and $\mathbb{P}^2$ for Case I.

2.6.1.5 Comparison of the OLS-regression and the NN-regression for approximation of pathwise option values

Finally, we compare the performance of the OLS-regression, with the NN-regression, introduced in Subsections 2.4.2 and 2.4.3. We emphasise that both OLS-regression and NN-regression are regressing the current state on discounted cash-flow paths approximated by the DOS algorithm (described in Section 2.3). They can therefore be seen as phase 2 of an algorithm producing pathwise option values.

After conducting numerical experiments on a variety of different examples we conclude that the expected exposures are very similar for the two regression methods. However, the potential future exposure is not always captured by the OLS-regression. The difficulty lies in finding a set of computationally feasible basis functions, flexible enough to accurately capture a complicated function surface on a large domain (similar problem as for the LSM in Subsection 2.6.1.3). To overcome this problem, we also implement a slightly different version of the algorithm, where we instead carry out local regression in the continuation regions. To be able to differentiate between the local and global OLS-regressions, we denote the regression functions by $v^{\mathrm{OLS}_{\mathrm{loc}}}$ and $v^{\mathrm{OLS}_{\mathrm{glob}}}$, respectively. The localization procedure is done similarly as in the SGBM, *i.e.,* at each $t \in \mathbb{T}$, the state space is divided into bundles of equal size (in terms of the number of samples in each bundle), based on $\max\{(S_t)_1, (S_t)_2\}$. With local OLS-regression we obtain almost identical exposure profiles as with NN-regression. In Figure 2.8, on top to the left, the exposure profiles computed with the three different algorithms are displayed. Furthermore, from top right to bottom right, we compare the approximate risk premia for holding instead of immediately exercising the option at $t_2 \approx 0.667$ and some $x \in \mathbb{R}^2$, *i.e.,* $v_{t_2}^Z(x) - g_{t_2}(x)$, with Z representing, in order NN, $\mathrm{OLS}_{\mathrm{glob}}$ and $\mathrm{OLS}_{\mathrm{loc}}$. We know that for all $x \in \mathbb{R}^2$, it holds that $V_{t_2}^Z(x) - g_{t_2}(x) \geq 0$. We see in Figure 2.8, top right, that this is captured by the NN-regression, since the values range from 0 to just above 12. When we carefully evaluate the values of the risk premia computed with local and global OLS-regression (Figure 2.8 bottom left and bottom right) we see that negative values exist in both plots. We see similar phenomena for high values, *i.e.,* the range is stretched upwards in comparison to the values obtained with NN-regression. The reason for the negative values is that $v^{\mathrm{OLS}_{\mathrm{loc}}}$ and $v^{\mathrm{OLS}_{\mathrm{glob}}}$ underestimate the option values close to the boundary (which in this case coincides with the exercise boundary since the regression is carried out only in the continuation region). To compensate for this, we see a tendency of higher values in the center (not close to the exercise boundaries) of the continuation regions. As a final remark, we note that this behaviour is reduced by using local regression instead of global regression (the range of values in Figure 2.8 is tighter for local than global regression). On the other hand, we note discontinuities in Figure 2.8 bottom, left, stemming from the localized regression, of the risk premium computed with the local OLS-regression. The discontinuity in Figure 2.8, bottom right, is a boundary issue of the OLS-regression (regression is only carried out in the continuation region since we set the value equal to the immediate pay-off in the exercise region).

Even though, the local OLS-regression is less accurate when it comes to computations of exposure profiles than the other two algorithms in this section, it is clear by comparing Figures 2.5 and 2.8, that it outperforms the LSM. This is not a surprise since:

1. The accumulated error in the LSM (because of recursive dependency of the regression functions) is significantly reduced since the regression functions are not sequentially dependent. The discounted cash-flows are projected onto the risk factors directly. This implies that the only error accumulation (over time), in the OLS-regression originates from the DOS algorithm, which computes the exercise boundaries with high accuracy;

2. By recalling equation (2.42), we note that a less accurate stopping strategy may cause a less accurate exposure.

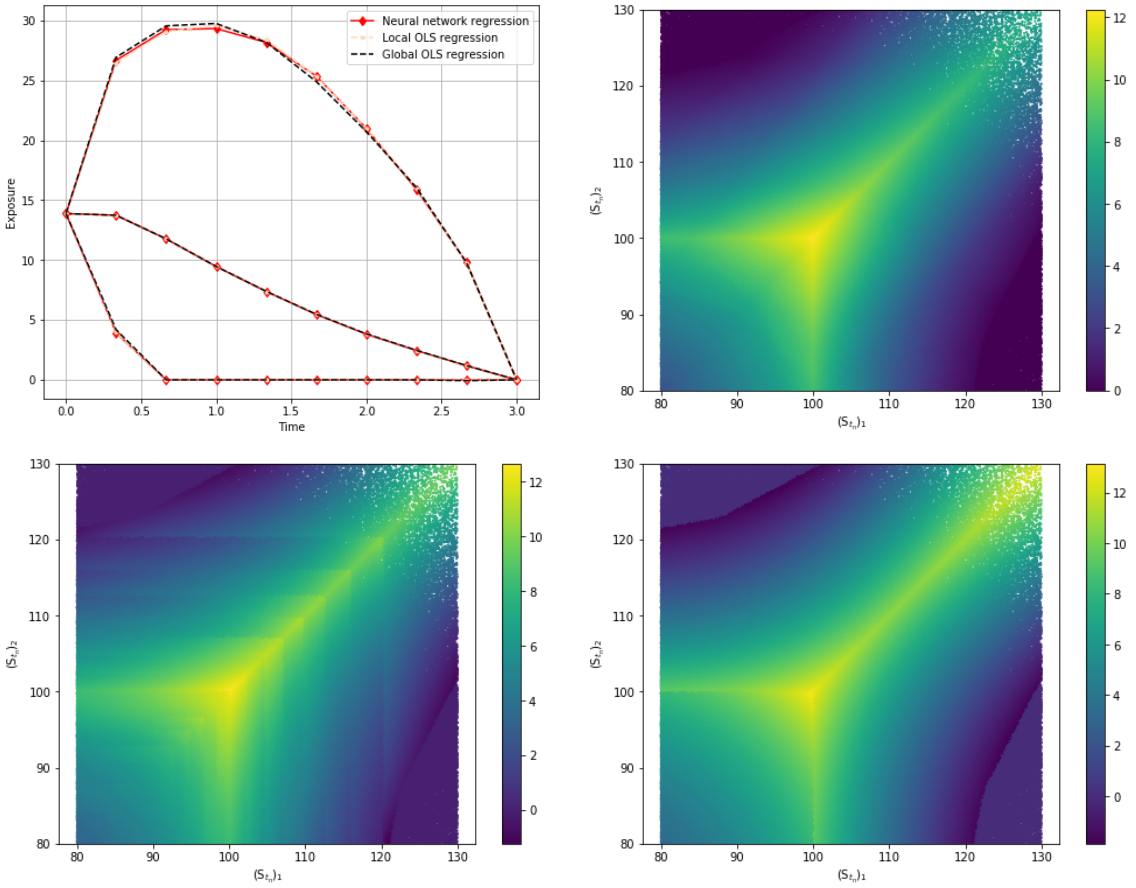

Figure 2.8: Comparison of NN-regression, local OLS-regression and global OLS-regression for a two-dimensional max-call option. For all three regression techniques, the DOS is used to approximate the optimal stopping strategy. **Top left:** Exposure profiles. **From top right to bottom left:** Approximate risk premium for holding option instead of immediate exercise at $t_2 \approx 0.667$, *i.e.,* $v^Z_{t_2}(\cdot) - g_{t_2}(\cdot)$, with Z representing NN, OLS_{loc} and OLS_{glob}, respectively.

2.6.2 Heston model dynamics

In this section we assume a one-dimensional underlying asset following the Heston stochastic volatility model [49], which is considered only under the risk-neutral measure. We therefore omit the explicit notation of the probability measure used in this section. In this setting, the market is described by, not only the underlying asset price process $S = (S_t)_{t\in[t_0,t_N]}$ itself , but also by the instantaneous variance process $\nu = (\nu_t)_{t\in[t_0,t_N]}$. The state process is then the

two-dimensional process $X = (\nu, S)$ which satisfies the system of SDEs

$$\mathrm{d}S_t = (r-q)S_t\mathrm{d}t + \sqrt{\nu_t}S_t\mathrm{d}W_t^S,\ S_{t_0} = s_{t_0};\quad t \in [t_0,\, t_N], \tag{2.43}$$

$$\mathrm{d}\nu_t = \kappa(\theta-\nu_t)\mathrm{d}t + \xi\sqrt{\nu_t}\mathrm{d}W_t^\nu,\ \nu_{t_0} = \nu_0;\quad t \in [t_0, t_N], \tag{2.44}$$

with risk-free interest rate $r \in \mathbb{R}$, dividend rate $q \in (0,\infty)$, initial conditions $s_{t_0}, \nu_0 \in (0,\infty)$, speed of mean reversion $\kappa \in (0,\infty)$, long term mean of the variance process $\theta \in (0,\infty)$, and volatility coefficient of the variance process $\xi \in (0,\infty)$. Furthermore, $(W_t^S)_{t\in[t_0,t_N]}$ and $(W_t^\nu)_{t\in[t_0,t_N]}$ are two one-dimensional, standard Brownian motions satisfying $\mathbb{E}[\mathrm{d}W_t^S\mathrm{d}W_t^\nu] = \rho_{\nu,S}\mathrm{d}t$ for some correlation parameter $\rho_{\nu,S} \in (-1,1)$. We, however notice that it is important to be careful when using the Heston model, since for some parameters, moments of higher order than 1 can become infinite in finite time (see [50, Proposition 3.1]). Equation (2.44) is the SDE for the well-established Cox–Ingersoll–Ross (CIR) process, introduced in [51]. When the so-called Feller condition

$$2\kappa\theta \geq \xi^2,$$

is satisfied, it holds that 0 is an unattaiable boundary for ν. If the Feller condition is not satisfied, then 0 is an attainable, but strongly reflective[24] boundary, see *e.g.,* [52]. This leads to an accumulation of probability mass around zero, which makes it more challenging to approximate ν accurately. Unfortunately, the Feller condition is rarely satisfied for parameters calibrated to the market.

In this chapter, the QE-scheme, proposed in [53] is used to approximate[25] (ν, S). If necessary, we choose a finer time grid for the approximation of (S,ν) than the exercise dates, $\mathbb{T}$.

We consider a standard Bermudan put option, *i.e.,* identical pay-off functions at all exercise dates, only depending on the underlying asset. Furthermore, the pay-off function for the Bermudan put option is given by

$$g(s) = (K-s)^+,$$

for $s \in (0,\infty)$, and strike $K \in \mathbb{R}$.

2.6.2.1 Comparison with Monte-Carlo-based algorithms

In this section, we again compare the DOS algorithm with the two Monte-Carlo-based algorithms, SGBM and LSM. We use the following set of model parameters: $r = 0.04$, $q = 0$, $s_{t_0} = 100$, $\kappa = 1.15$, $\theta = 0.0348$, $\xi = 0.459$, $\nu_0 = 0.0348$ and $\rho_{\nu,S} = -0.64$, and the contract parameters: $T = 0.25$, $N = 10$, $K = 100$. The parameters coincide with Set B in [54], in which the valuation is carried out with the so-called 2D-COS method. The 2D-COS method is a Fourier-based method and is assumed to yield highly accurate valuation of the option.

For the LSM, we use as basis functions, for $t_n \in \mathbb{T}$, Laguerre polynomials of degree 3 of S_{t_n}, Laguerre polynomials of degree 3 of ν_{t_n} and $\nu_{t_n}S_{t_n}$ (only on constant basis function is used). For the SGBM, we use 32 equally-sized bundles based on S_{t_n} and for $t_n \in \mathbb{T}$, we use as basis functions a constant, S_{t_n}, $S_{t_n}\nu_{t_n}$ and the first 3 powers of ν_{t_n}. These parameters are

[24] Strongly reflective in the sense that the time spent at 0 is of Lebesgue measure zero, see *e.g.,* [50].

[25] For notational convenience, the state process is denoted by X, which falsely indicates that we have an exact form for X. This is because our focus is on approximating option values, not the underlying state process. It should however be mentioned that X needs to be approximated in this section.

chosen such that the approximate option value at t_0 are as close as possible to the (almost) exact value of 3.198944, for a Bermudan option with 10 exercise dates, retrieved from [54]. The obtained option values (for each algorithm, average value of 10 runs) at t_0 were 3.1792, 3.2033, and 3.1222 for the DOS algorithm, the SGBM and the LSM, respectively.

It should be stressed that the numerical results for the LSM and the SGBM in this section should not be seen as state of the art performance of the algorithms. For example, in [35], a bundling scheme based on recursive bifurcation and rotation of the state space to match the correlation between S and ν gave accurate results. Furthermore, they use as basis functions only monomials of $\log(S_{t_n})$ and letting the stochastic variance ν_{t_n} enter only through conditional expectations of the form $\mathbb{E}[(\log(S_{t_n}))^k \mid S_{t_{n-1}}, \nu_{t_{n-1}}]$. These conditional expectations are computed from the characteristic function of the $S_{t_n} \mid S_{t_{n-1}}, \nu_{t_{n-1}}$ which was presented in [49]. Similar improvements could also be done for the LSM. The reason for this comparison to still be relevant is to demonstrate the flexibility of DOS and the NN-regression, in which nothing has been changed (from the examples with Black–Scholes dynamics) except the dynamics of the stochastic process from which we generate training data.

In Figure 2.9, the exercise boundaries are presented in the state space. Worth noticing is that for $t_n \in \mathbb{T}$, both the option value and the pay-off functions are increasing in ν_{t_n} and decreasing in S_{t_n}. An immediate consequence of this is that we can have at most one exercise boundary along the lines with constant S_{t_n} or constant ν_{t_n}. We can therefore conclude inconsistencies in the exercise boundaries for both the LSM and the SGBM. In Figure 2.9, on the bottom line to the right, the empirical probability density functions (pdf) of the exposures at the exercise dates are plotted. Figure 2.10 shows the exposure profiles and the exercise frequency computed with the three algorithms. We note that the DOS and the SGBM seem to agree fairly well on the EE and the lower percentile of the PFE but differ significantly on the upper PFE.

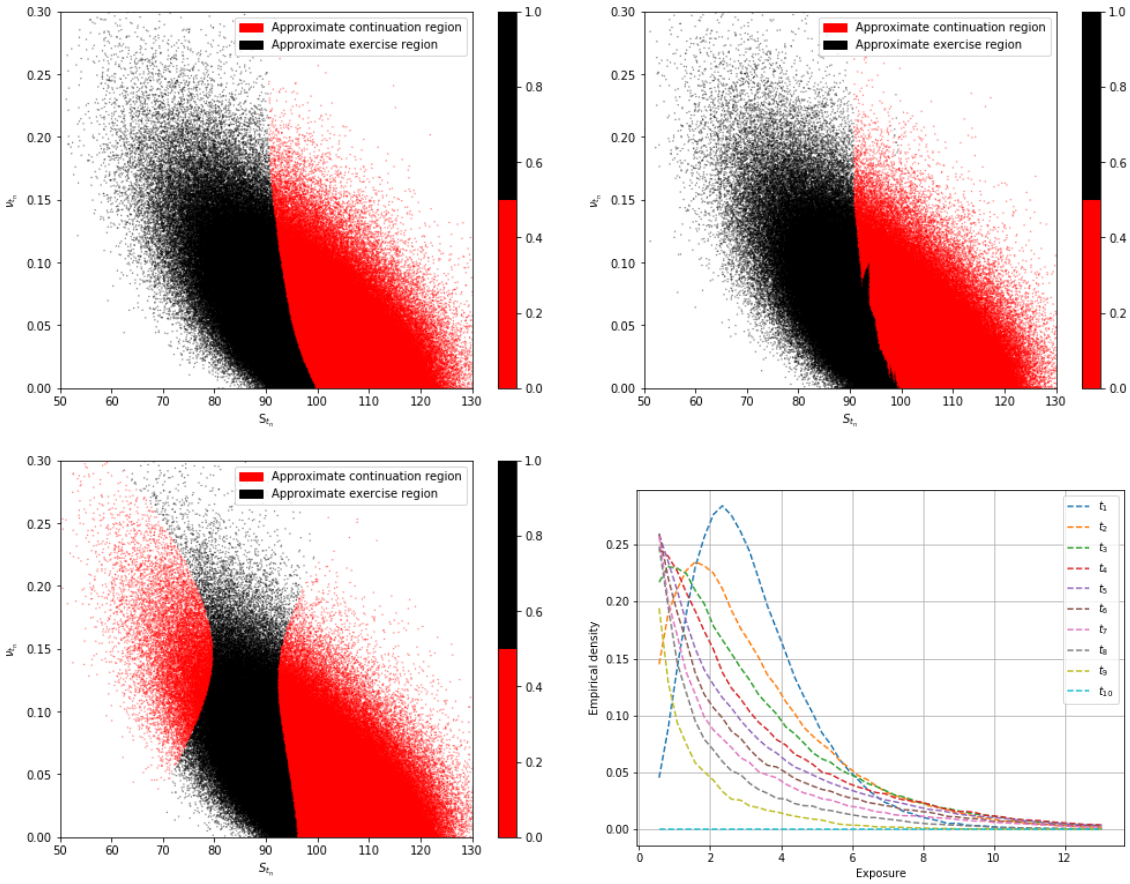

Figure 2.9: Approximate exercise boundaries for a Bermudan put option under the Heston model at $t_9 = 0.225$ and the empirical density for the exposure at all exercise dates. **From top left to bottom right:** DOS, SGBM, LSM and the empirical density of the exposure.

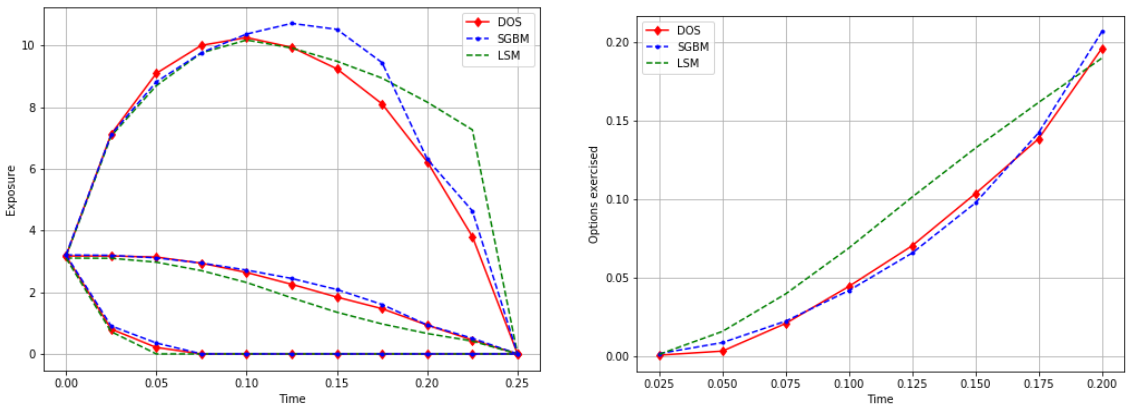

Figure 2.10: Comparison of the DOS algorithm, the SGBM and the LSM for a Bermudan put option under the Heston model. **Left:** Expected exposure and potential future exposures at 97.5%- and 2.5%-levels. **Right:** Proportion of options exercised at different exercise dates.

3

Deep learning for CVA computations of large portfolios of financial derivatives

In this chapter, we propose a neural network-based method for CVA computations of a portfolio of derivatives. In particular, we focus on portfolios consisting of a combination of derivatives, with and without true optionality, *e.g.,* a portfolio of a mix of European- and Bermudan-type derivatives. CVA is computed, with and without netting, for different levels of WWR and for different levels of credit quality of the counterparty. We show that the CVA is overestimated with up to 25% by using the standard procedure of not adjusting the exercise strategy for the default-risk of the counterparty. For the Expected Shortfall of the CVA dynamics, the overestimation was found to be more than 100% in some non-extreme cases.

Keywords *- Bermudan options, option portfolios, risk management, CVA, neural networks*

This chapter is based on the paper with the same title which is published in Applied Mathematics and Computation 409 (2021): 126399

3.1 Introduction

In this chapter, we consider a set of financial contracts, which we refer to as the *portfolio of derivatives*, or just the *portfolio*, written between two parties. The first party is referred to as the *bank* and is considered to be default-free. The second party, which may default, is referred to as the *counterparty*. We take the perspective of the default-free bank in order to investigate some of the risks associated with a defaultable counterparty. It is straight-forward to extend the methodologies used in this chapter to a defaultable bank as well as to multiple counterparties.

3.1.1 Risk-free valuation

We consider the problem of finding the value of a portfolio of derivatives with early-exercise features. In particular, we are focusing on portfolios with multiple derivatives with true optionality, *e.g.,* American or Bermudan derivatives. We construct a portfolio of J derivatives, where the individual derivatives depend on $d_1, d_2, \ldots, d_J$ risk factors. This means that we could face high-dimensionality in two ways:

1. Derivative j could depend on a large number of risk factors, *i.e.,* d_j could be large;
2. We could have many derivatives in the portfolio, *i.e.,* J could be large.

In [9], a neural network-based method for valuation of a single Bermudan derivative was proposed and proved to be highly accurate for derivatives with up to 100 risk factors. Later, the algorithm was extended in the work presented in Chapter 2 to also include pathwise valuations of the derivative (in contrast to only finding the value at the initial time). In this chapter, we extend [9] and the material from Chapter 2 to the portfolio case, *i.e.,* finding the value of a large portfolio of, possibly high-dimensional, derivatives with true optionality, without having to compute the value of each individual derivative.

In a traditional setting, the so-called continuation value is computed, and subsequently, the value of the derivative is given by the maximum of the continuation value and the immediate pay-off. For a single derivative, this is straight-forward. For instance, the continuation value can be computed by solving an associated PDE, which is done in *e.g.,* [12], [13], [14], [15] and [16], or the continuation value can be approximated by a Fourier transform methodology, which is done in e.g., [17], [18] and [19]. Furthermore, classical tree-based methods such as [20], [21] and [22], can be used. These types of methods are, in general, highly accurate but they suffer severely from the curse of dimensionality, meaning that they are computationally feasible only in low dimensions (say up to 4 risk factors), see [23]. In higher dimensions, Monte-Carlo-based methods are often used, see *e.g.,* [24], [25], [26], [27] and [28]. Monte-Carlo-based methods can generate highly accurate derivative values at the initial time, but often less accurate values between the initial time and maturity of the contract.

In contrast to the single derivative case, it is not enough to know the continuation value of a portfolio (with more than one derivative) in order to decide optimally which derivatives should be exercised. Therefore, it is common to do the valuation at the level of each derivative, and then add the individual values of each derivative to obtain the portfolio value. This becomes cumbersome for large portfolios. As mentioned above, the methodology used in this

chapter, generalizes [9] and Chapter 2, in which the optimal exercise policy is approximated by maximizing expected discounted cash-flows, *i.e.,* the continuation value is not computed. By not relying on computations of the continuation value, the algorithm is able to compute the portfolio value without having to compute the individual values for each derivative.

3.1.2 Risky valuation and CVA

The Credit Valuation Adjustment (CVA) is the difference between the risk-free portfolio value and the risky portfolio value, where the risky portfolio value is defined as the portfolio value when taking default risk of the counterparty into account. While there is no ambiguity of the risk-free portfolio value, it is not completely clear how the risky portfolio value should be computed. The question is whether the exercise policy should be adjusted for the fact that the counterparty may default. For instance, if the counterparty ends up in financial distress, it is reasonable to assume that the bank (which in this chapter is assumed to be the risk-free party) would be more willing to exercise the callable derivatives, in order to lower its exposure to the counterparty. Even though it seems common to ignore the effect of a defaultable counterparty when computing risky derivative values, it has been discussed in the literature, see *e.g.,* [55], [38], [39] and [56]. In the case of a single derivative [55] states that the exercise region for a risk-free derivative is always a subset of the exercise region for a risky counterpart. However, in the case of a portfolio, the situation is more complex, and depends on contractual details such as the close-out and netting agreements. One consequence is that, in the presence of a netting agreement, the exercise decisions can no longer be made individually. To explain this, we give a simple example.

Example 3.1.1. *Assume that we have a portfolio, consisting of three derivatives, one European future and two American options. All contracts are initialized at time 0, mature at time T and depend on the same risk-factor $(X_t)_{t\in[0,T]}$. Assume that, at time $t \in (0,T)$, and given $X_t = x$, the intrinsic values are*

$$V^{\text{future}}(t,x) = -10, \quad V_1^{\text{Am}}(t,x) = 10, \quad V_2^{\text{Am}}(t,x) = 10,$$

and the immediate pay-off for the American options satisfy

$$g_1^{\text{Am}}(t,x) < 10, \quad g_2^{\text{Am}}(t,x) < 10.$$

In a risk-free environment (no-defaultable counterparty), it is sub-optimal to exercise the American options. However, in case of a defaultable counterparty, the situation is less trivial. In Table 3.1, the exposure to the counterparty, given different exercise decisions at t, is given with and without a netting agreement. If the counterparty is in severe financial distress, then

	Without netting	With netting
Exposure - no exercise	20	10
Exposure - exercise one of the American options	10	0
Exposure - exercise both American options	0	0

Table 3.1: Exposure given different exercise decisions at t, with and without a netting agreement.

it is likely optimal for the bank to exercise both American options in the case of no netting

agreement, and one of them in the case of a netting agreement. From this simple example, two things become clear; 1) The exercise decisions for the American options are affected not only by a risky counterparty, but also by whether or not a netting agreement exists. 2) in the presence of netting, exercise decisions cannot be made for one derivative in isolation, but only for all the American options simultaneously.

In general, for a risky portfolio, it is not possible to describe the value of a single derivative, but only the value of the entire portfolio. This is an interesting problem since almost all existing algorithms rely on exercise decisions made in isolation and risky derivative values that can be added up to obtain the risky portfolio value.

If this is not taken into account we would obtain a biased low valuation for the risky portfolio by using a sub-optimal exercise strategy. Since the CVA is the difference between the risk-free and risky portfolio values, we would obtain an overestimation of the CVA. Furthermore, this effect is likely to increase with decreasing credit quality of the counterparty. In practice, this means that the counterparty is paying a CVA which is based on a sub-optimal exercise strategy used by the bank, which is out of control for the counterparty. Even more problematic is that the overestimation of the CVA is higher for counterparties that already are under financial distress.

One could argue that it is reasonable for the bank to charge the counterparty the higher CVA, since the bank will probably not follow the theoretically optimal risky exercise strategy. However, there is another level of complexity not yet discussed. When the mark-to-market (MtM) CVA moves in time against the bank, the bank could face losses, not because the counterparty actually defaults, but because disadvantageous changes in the MtM CVA. For instance, in Basel III [57] the following is stated:

Under Basel II, the risk of counterparty default and credit mitigation risk were addressed but mark-to-market losses due to credit valuation adjustments (CVA) were not. During the global financial crisis, however, roughly two-thirds of losses attributed to counterparty credit risk were due to CVA losses and only about one-third were due to actual defaults.

This is further discussed in [58], in which the authors also recommend computations of different risk measures for the future distribution of CVA. Two examples of such measures are the Value at Risk of the CVA (VaR-CVA) and the Expected Shortfall of the CVA (ES-CVA). The advantage of the ES-CVA is that it is a coherent risk-measure, and we therefore focus on ES-CVA in this chapter.

3.1.3 Structure of the chapter

In Section 3.2 the mathematical problem formulation is given. We define the risk-free and risky portfolios, close-out agreements both with and without netting agreements and the associate CVA. Furthermore, the problems are formulated in terms of so-called decision functions, which control the exercise strategies. In Section 3.3, the algorithms are presented. In the first part, the algorithm for learning optimal exercise strategies is given and in the second part, an algorithm for learning pathwise entities such as the pathwise portfolio exposure is presented. Finally, in Section 3.4 numerical experiments are presented. The experiments include a first part, in which risk-free values are computed and compared to a well-established regression-based method. In

the second part we compare CVA computed with the risk-free and the risky exercise strategy to verify that, indeed, the CVA is often overestimated with algorithms in use today. We present comparisons with and without netting, for different levels of Wrong Way Risk (WWR), and for different credit quality of the counterparty. As a final example, we analyse the effect of the different exercise strategies on ES-CVA. In the Appendix, we provide some additional details on the algorithms and the specific choice of neural networks.

3.2 Problem formulation

Let $(\Omega, \mathcal{F}, \mathbb{Q})$ be a probability space completed with the $\mathbb{Q}-$null-sets of $\mathcal{F}$. For $T \in (0, \infty)$, and[1] $d \in \mathbb{N}$, let $X\colon [0,T] \times \Omega \to \mathbb{R}^d$ and $r\colon [0,T] \times \Omega \to \mathbb{R}$ represent the (market) risk-factors of the portfolio and the short rate, respectively. Furthermore, we denote by τ^{D} the default event of the counterparty, which is a stopping time defined on $(\Omega, \mathcal{F}, \mathbb{Q})$ and we let $\mathbb{1}^{\mathrm{D}} : [0,T] \times \Omega \to \{0,1\}$, be the *jump-to-default* process given by

$$\mathbb{1}_t^{\mathrm{D}} \coloneqq \mathbb{I}_{\{t < \tau^{\mathrm{D}}\}}. \tag{3.1}$$

The information structure is given by the sub$-\sigma-$algebras generated by X, r and $\mathbb{1}^{\mathrm{D}}$, *i.e.,* $\mathcal{H}_t^X = \sigma\left(X_s\colon s \in [0,t]\right)$, $\mathcal{H}_t^r = \sigma\left(r_s\colon s \in [0,t]\right)$ and $\mathcal{G}_t = \sigma\left(\mathbb{1}_s^{\mathrm{D}}\colon s \in [0,t]\right)$ and we define the enlarged filtrations $\mathcal{H}_t = \mathcal{H}_t^X \wedge \mathcal{H}_t^r$ and $\mathcal{F}_t = \mathcal{H}_t \wedge \mathcal{G}_t$. In this chapter, we use either a constant short rate (risk-free rate), or we view the short rate as one of the risk factors. In the latter case, we model the short rate as one of the d component processes of X, which implies that, $\mathcal{H}_t = \mathcal{H}_t^X$. The motivation for introducing a separate notation for the short rate is to simplify the notation when the short rate is used to discount cash-flows. For commonly used conditional expectations, we introduce the short-hand notations $\mathbb{E}_{t,x}\left[\,\cdot\,\right] \coloneqq \mathbb{E}^{\mathbb{Q}}\left[\,\cdot \,|\, X_t = x\right]$, $\mathbb{E}_{t,x,\nu}\left[\,\cdot\,\right] \coloneqq \mathbb{E}^{\mathbb{Q}}\left[\,\cdot \,|\, X_t = x, \mathbb{1}_t^{\mathrm{D}} = \nu\right]$ and $\mathbb{E}_t\left[\,\cdot\,\right] \coloneqq \mathbb{E}^{\mathbb{Q}}\left[\,\cdot \,|\, \mathcal{H}_t\right]$.

We use a numéraire, which, for $t \in [0,T]$, is defined by $B_t \coloneqq \exp\left(\int_0^t r_s \mathrm{d}s\right)$, which should be interpreted as the value at time t of a savings-account, which was worth 1 at time 0. For $t, u \in [0,T]$ with $t \leq u$, we use $D_{t,u} \coloneqq \frac{B_t}{B_u}$ to discount a cash-flow obtained at time u back to time t. The measure $\mathbb{Q}$ is the risk-free measure, under which all tradeable assets are martingales relative to the numéraire, *e.g.,* if component $i \in \{1, 2, \ldots, d\}$ of X is tradeable, then $\frac{(X_t)_i}{B_t}$ is a $\mathbb{Q}-$martingale.

If not specifically stated otherwise, equalities and inequalities of random variables should be interpreted in a $\mathbb{Q}-$almost sure sense.

3.2.1 A portfolio of derivatives

We assume a portfolio of $J \in \mathbb{N}$ derivatives. For $t \in [0,T]$, and for derivative $j \in \{1, 2, \ldots, J\}$, we denote the set of exercise dates greater than or equal to t by $\mathbb{T}_j(t) \subseteq [0,T]$, and set $\mathbb{T}(t) = \{\mathbb{T}_1(t), \mathbb{T}_2(t), \ldots, \mathbb{T}_J(t)\}$. Note that for a European-type contract, the only exercise date is at the maturity, for a Bermudan-type contract there are multiple exercise dates, and for an American-type contract, there are infinitely many exercise dates. We emphasize that the exercise dates are simply subsets of the time interval $[0,T]$, and provide no information

[1] We use $\mathbb{N} = \{1, 2, 3 \ldots\}$ and $\mathbb{N}_0 = \mathbb{N} \cup \{0\}$, and $\mathbb{R}_+ = (0, \infty)$.

on which exercise policy to follow, except in some trivial cases *e.g.,* when there is only one exercise date.

Since we want to be able to treat derivatives with early-exercise features, we need to introduce a framework for stopping times. For $j \in \{1, 2, \dots, J\}$, an $X-$stopping time with respect to $\mathbb{T}_j(0)$, is a random variable, τ_j, defined on $(\Omega, \mathcal{F}, \mathbb{Q})$, taking on values in $\mathbb{T}_j(0)$, such that for all $s \in \mathbb{T}_j(t)$, it holds that the event $\{\tau_j = s\} \in \mathcal{H}_s$. Furthermore, we define an $X^{t,x}-$stopping time as an $X-$stopping time, conditional on $X_t = x$, and $\tau \geq t$.

For each derivative, $j \in \{1, 2, \dots, J\}$ we use individual pay-off functions, $g_j \colon [0, T] \times \mathbb{R}^d \to \mathbb{R}$, which, for $t, s \in [0, T]$, with $s \geq t$, are assumed to satisfy

$$\mathbb{E}_0[|D_{t,s} g_j(s, X_s)|^2] < \infty. \tag{3.2}$$

Since we are treating portfolios where the individual derivatives may have different maturities, we set each pay-off function to zero for all times larger than its maturity, *i.e.,* for $j \in \{1, 2, \dots, J\}$, $x \in \mathbb{R}^d$ and for $t > \max\{\mathbb{T}_j(0)\}$, we set $g_j(t, x) \equiv 0$, where $\max\{\mathbb{T}_j(0)\}$ represents the largest element belonging to the set $\mathbb{T}_j(0)$.

3.2.2 Risk-free and risky portfolio valuation without netting

The value of a derivative (not) taking default risk of the counterparty into account is referred to as the *risky* (*risk-free*) value. We define the risk-free and the risky values of derivative $j \in \{1, 2, \dots, J\}$, at market state $(t \in [0, T], X_t = x \in \mathbb{R}^d)$, and default state $\mathbb{1}_t^{\mathrm{D}} = \nu \in \{0, 1\}$, by

$$V_j(t, x) \coloneqq \sup_{\tau \in \mathcal{T}_j(t)} \mathbb{E}_{t,x} \left[D_{t,\tau} g_j(\tau, X_\tau) \right], \quad \text{(risk-free value)}, \tag{3.3}$$

$$\begin{aligned} U_j(t, x, \nu) \coloneqq & \nu \sup_{\tau \in \mathcal{T}_j(t)} \mathbb{E}_{t,x,1} \big[\mathbb{1}_\tau^{\mathrm{D}} D_{t,\tau} g_j(\tau, X_\tau) \\ & + (1 - \mathbb{1}_\tau^{\mathrm{D}}) D_{t,\tau^{\mathrm{D}}} \left(R V_j(\tau^{\mathrm{D}}, X_{\tau^{\mathrm{D}}})^+ + V_j(\tau^{\mathrm{D}}, X_{\tau^{\mathrm{D}}})^- \right) \big], \quad \text{(risky value)}, \end{aligned} \tag{3.4}$$

where $\mathcal{T}_j(t)$ is the set of all $X-$stopping times taking on values in $\mathbb{T}_j(t)$ and for $x \in \mathbb{R}$, $(x)^+ = \max\{0, x\}$ and $(x)^- = \min\{0, x\}$. In the above, we assume a close-out agreement which uses the risk-free derivative values as reference valuation. At default of the counterparty, the bank receives only a fraction, $R \in [0, 1)$, referred to as the *recovery-rate*, of the positive part of each derivative. On the other hand, each derivative with a negative risk-free value at default needs to be added entirely to the portfolio.

Note that for the risky value we need additional information of prior defaults of the counterparty, which is captured in the realization, $\nu \in \{0, 1\}$, of the jump-to-default process, *i.e.,* $\nu = 1$ if no default has occurred prior to, or at t, and $\nu = 0$ otherwise. The notation above trivially holds for European-type derivatives since the only exercise date is at maturity of the contract. Furthermore, a barrier-type feature could be added by also including a spatial dimension to $\mathbb{T}_j(0)$. The value of a portfolio, consisting of J derivatives, at market state $(t, X_t = x)$ and default state $\mathbb{1}_t^{\mathrm{D}} = \nu$, without netting, is given by

$$\Pi^V(t, x) \coloneqq \sum_{j=1}^{J} V_j(t, x), \quad \Pi^U(t, x, \nu) \coloneqq \sum_{j=1}^{J} U_j(t, x, \nu) = \nu \sum_{j=1}^{J} U_j(t, x, 1).$$

Using (3.3) and (3.4), the above can be written as

$$\Pi^V(t,x) = \sum_{j=1}^{J} \sup_{\tau_j \in \mathcal{T}_j(t)} \mathbb{E}_{t,x}\left[D_{t,\tau_j} g_j(\tau_j, X_{\tau_j})\right] \tag{3.5}$$

$$\begin{aligned}\Pi^U(t,x,\nu) =& \nu \sum_{j=1}^{J} \sup_{\tau_j \in \mathcal{T}_j(t)} \mathbb{E}_{t,x,1}\Big[\mathbb{1}^{\mathrm{D}}_{\tau_j} D_{t,\tau_j} g_j(\tau_j, X_{\tau_j}) \\ &+ (1 - \mathbb{1}^{\mathrm{D}}_{\tau_j}) D_{t,\tau^{\mathrm{D}}}\big(RV_j(\tau^{\mathrm{D}}, X_{\tau^{\mathrm{D}}})^+ + V_j(\tau^{\mathrm{D}}, X_{\tau^{\mathrm{D}}})^-\big)\Big].\end{aligned} \tag{3.6}$$

Since the aim is to approximate the optimal exercise policy with neural networks, we wish to re-formulate the problem into an optimization problem, in which the target function can be represented by a neural network. Following [9] and Chapter 2 we use so-called decision functions, to determine for each derivative and given a market state, whether or not to exercise the derivative. For $j \in \{1, 2, \ldots, J\}$, decision function j denoted by f_j, is of the form $f_j \colon [0,T] \times \mathbb{R}^d \to \{0,1\}$. In order to guarantee that an exercise decision can only occur at an exercise date, we require for $s \notin \mathbb{T}_j(0)$, that $f_j(s,\cdot) \equiv 0$.

We now restrict our attention to the case when there, for each derivative, is a finite number of exercise dates, *i.e.,* for $j \in \{1, 2, \ldots, J\}$, it holds that $|\mathbb{T}_j(0)| \in \mathbb{N}$. From a theoretical perspective, this excludes American-type derivatives, but from a practical perspective, an infinite number of exercise dates is often approximated by a large, but finite, number of exercise dates. This implies that we can still consider American-type derivatives by increasing the number of exercise dates until the derivative value converges (until the value does not increase with additional exercise dates). We denote by $\mathbb{T}^{\Pi}(t)$ the set of dates which represent an exercise date for at least one of the J derivatives. Mathematically, we define the exercise dates of the portfolio as

$$\mathbb{T}^{\Pi}(t) \coloneqq \bigcup_{j=1}^{J} \mathbb{T}_j(t), \tag{3.7}$$

and the number of unique exercise dates in the portfolio is given by $N = |\mathbb{T}^{\Pi}(0)|$. We assume that the initial time $T_0 = 0$ is not an exercise date for any of the derivatives.

To simplify, we use the following notation for the N exercise dates, the risk-factors evaluated at the N exercise dates, and the discounting between exercise dates

$$\mathbb{T}^{\Pi}(0) = \{T_1, T_2, \ldots, T_N = T\}, \tag{3.8}$$

$$X_k \coloneqq X_{T_k}, \text{ and } D_{k,\ell} \coloneqq D_{T_k,T_\ell} \quad \text{for } k, \ell = 1, 2, \ldots, N. \tag{3.9}$$

Furthermore, for $t \in [0,T]$, the risk-factor process on $[t,T]$, conditional on $X_t = x$, is denoted by $X^{t,x} = (X_s)_{s\in[t,T]}$, where we also note that $X^{0,x_0} = X$. The above notation allows us to express an $X^{t,x}-$stopping time in terms of decision functions[2]

$$\tau_j^n[f_j](X^{t,x}) \coloneqq \sum_{k=n}^{N} T_k f_j(T_k, X_k) \prod_{m=n}^{k-1} (1 - f_j(T_m, X_m)). \tag{3.10}$$

The notation above is used to emphasize that, the decision function f_j, controls the exercise strategy, given the stochastic process $X^{t,x}$. Moreover, $X^{t,x}$ is not just a random value at a

[2]The empty product is defined as 1.

specific time, but the entire process, starting at $X_t = x$ and until stopping occurs. In later sections the valuation of a derivative or a portfolio is formulated as an optimization problem, which is optimized by varying f_j. Although the notation is practical when optimization is discussed, it is cumbersome to use when we define the value of a derivative. We therefore use the following short-hand notation

$$\bar{\tau}_{n,j} \coloneqq \tau_j^n[f_j](X^{t,x}), \tag{3.11}$$

and keep in mind, that the strategy is controlled by a decision function, f_j, and for $u \geq t$, the event $\mathbb{I}_{\{\bar{\tau}_{n,j} \leq u\}}$ is $\sigma(X^{t,x})-$measurable. We can now define the value of the risk-free and risky derivatives, given an exercise strategy expressed in terms of decision functions. For derivative $j \in \{1, 2, \ldots, J\}$, market state $(t, x) \in (T_{n-1}, T_n] \times \mathbb{R}^d$, we define the parametrized valuation functions

$$\mathcal{V}_j(t, x \,|\, f_j) \coloneqq \mathbb{E}_{t,x}\left[D_{t,\bar{\tau}_{n,j}}\, g_j\left(\bar{\tau}_{n,j}, X_{\bar{\tau}_{n,j}}\right)\right], \tag{3.12}$$

$$\begin{aligned}\mathcal{U}_j(t, x \,|\, f_j) \coloneqq &\mathbb{E}_{t,x,1}\big[\mathbb{1}^{\mathrm{D}}_{\bar{\tau}_{n,j}} D_{t,\bar{\tau}_{n,j}} g_j(\bar{\tau}_{n,j}, X_{\bar{\tau}_{n,j}}) + (1 - \mathbb{1}^{\mathrm{D}}_{\bar{\tau}_{n,j}}) D_{t,\tau^{\mathrm{D}}}\big(RV_j(\tau^{\mathrm{D}}, X_{\tau^{\mathrm{D}}})^+ \\ &+ V_j(\tau^{\mathrm{D}}, X_{\tau^{\mathrm{D}}})^-\big)\big].\end{aligned} \tag{3.13}$$

Similarly, we define the portfolio values with respect to the exercise strategy given by $\boldsymbol{f}$, as the parametrized functions

$$\Upsilon^{V}(t, x \,|\, \boldsymbol{f}) \coloneqq \sum_{j=1}^{J} \mathcal{V}_j(t, x \,|\, f_j), \quad \Upsilon^{U}(t, x, \nu \,|\, \boldsymbol{f}) \coloneqq \nu \sum_{j=1}^{J} \mathcal{U}_j(t, x \,|\, f_j), \tag{3.14}$$

where the value of the risky portfolio also depends on the default state of the counterparty, $\mathbb{1}^{\mathrm{D}}_t = \nu \in \{0, 1\}$. We now want to find decision functions such that, when inserted in (3.12) and (3.13), we obtain (3.3) and (3.4). With this in mind, we define for $j \in \{1, 2, \ldots, J\}$, at $t \in [0, T]$, the (optimal) exercise regions, $\mathcal{E}_j^z(t)$, in which it is optimal to exercise, and the (optimal) continuation regions, $\mathcal{C}_j^z(t)$, in which it is optimal to hold on, by

$$\begin{aligned}\mathcal{E}_j^z(t) &\coloneqq \left\{x \in \mathbb{R}^d \,|\, Z_j(t, x) = g_j(t, x) \text{ and } t \in \mathbb{T}_j(0)\right\}, \\ \mathcal{C}_j^z(t) &\coloneqq \left\{x \in \mathbb{R}^d \,|\, Z_j(t, x) > g_j(t, x) \text{ or } t \notin \mathbb{T}_j(0)\right\}, \text{ for } Z \in \{V, U\}.\end{aligned}$$

The above states that the derivative should be exercised if its value equals the immediate exercise value, and we are at an exercise date and the derivative should not be exercised if its value is greater than the immediate exercise value or if we are not at an exercise date. Note that $\mathcal{E}_j^z(t) \cup \mathcal{C}_j^z(t) = \mathbb{R}^d$ and $\mathcal{E}_j^z(t) \cap \mathcal{C}_j^z(t) = \emptyset$. For $t \in [0, T]$, a decision function, $j \in \{1, 2, \ldots, J\}$, can then be defined as

$$f_j^z(t, x) \coloneqq \mathbb{I}_{\{x \in \mathcal{E}_j^z(t)\}}, \text{ for } Z \in \{V, U\}. \tag{3.15}$$

Furthermore, we denote by $\boldsymbol{f}^z$, the vector consisting of the individual decision functions

$$\boldsymbol{f}^z(t, x) \coloneqq (f_1^z(t, x), f_2^z(t, x), \ldots, f_J^z(t, x))^T, \text{ for } Z \in \{V, U\}. \tag{3.16}$$

For market states $(t, x) \in [0, T] \times \mathbb{R}^d$ and for a derivative $j \in \{1, 2, \ldots, J\}$, it holds that

$$\mathcal{V}_j(t, x \,|\, f_j^D,\, f_j^{\mathrm{D}}) = V_j(t, x), \quad \mathcal{U}_j(t, x \,|\, f_j^U) = U_j(t, x, 1).$$

The validity of the above is a direct consequence of Proposition 4 in [9]. In turn, this implies that by inserting the optimal decision functions in the functionals in equations (3.14), we obtain the risky and risk-free portfolio values, *i.e.*,

$$\Upsilon^V(t, x \,|\, \boldsymbol{f}^V) = \Pi^V(t, x), \quad \Upsilon^U(t, x, \nu \,|\, \boldsymbol{f}^U) = \Pi^U(t, x, \nu).$$

In subsequent sections the optimal decision function $\boldsymbol{f}^Z$, for $Z \in \{V, U\}$, is approximated with a series of neural networks. The reason for using the rather complicated notation, (3.10), is that this structure allows us to view the valuation of the derivatives as an optimization problem over the set of decision functions, which we approximate on some finite-dimensional function space. One example of such function space is the functions generated by a series of neural networks with a fixed number of parameters. When we use a specific strategy, *e.g.,* $\boldsymbol{f}^V$ or $\boldsymbol{f}^U$, this is specified by adding a superscript referring to the particular strategy. For $Z \in \{V, U\}$, we define the short-hand notation

$$\bar{\tau}_{n,j}^Z \coloneqq \tau_j^n[f_j^Z](X^{t,x}), \tag{3.17}$$

where it is assumed that $t \in (T_{n-1}, T_n]$.

3.2.3 Risky portfolio valuation with netting

When considering the risky portfolio value with netting, the problem becomes nonlinear in the sense that the risky portfolio value is no longer the sum of the individual risky derivative values. In fact, there no longer exists "a risky value for a single derivative", since the valuation needs to be carried out on a portfolio level. Before we define the risky value of a netted portfolio, we need to define the process[3] $A\colon [0, T] \times \Omega \to \{0, 1\}^J$, which for $t \in [0, T]$ and $j \in \{1, 2, \ldots, J\}$ satisfies

$$(A_t)_j = \begin{cases} 0, & \text{if derivative } j \text{ has been exercised prior to } t, \\ 1, & \text{else.} \end{cases}$$

Similar to (3.9), we use the short-hand notation for A at initial date, T_0, the exercise dates, $T_1, \ldots, T_N$,

$$A_k \coloneqq A_{T_k}, \quad \text{for } k = 0, 1, \ldots, N. \tag{3.18}$$

The process A is $\mathcal{H}_t-$measurable but it is not enough to know $X_t = x$ in order to determine A_t. The reason for defining A is that the exercise decisions for the netted risky portfolio are defined by a $J-$dimensional $(X^{t,x}, A^{t,\alpha})-$stopping times vector, where $A^{t,\alpha} = (A_s)_{s \in [t,T]}$ conditional on $A_t = \alpha$. This means that, at each exercise date, in addition to the current market state, we need to know which derivatives in the portfolio have been exercised prior to the current time, in order to make optimal exercise decisions. We denote, by $\boldsymbol{\mathcal{T}'}(t)$, the space of $(X^{t,x}, A^{t,\alpha})-$stopping times vectors, taking on values in $\{\mathbb{T}_1(t), \mathbb{T}_2(t), \ldots, \mathbb{T}_J(t)\}$. Furthermore, for $t \in (T_{n-1}, T_n]$,

[3] $\{0,\, 1\}^J$ is the Cartesian product of $\{0,\, 1\} \times \{0,\, 1\} \times \cdots \times \{0,\, 1\}$, J times.

we denote by τ_j^n element j of a stopping times vector $\boldsymbol{\tau}^n \in \boldsymbol{\mathcal{T}'}(t)$. The netted portfolio value with a risky counterparty, given market state $X_t = x$, default state of the counterparty $\mathbb{1}_t^{\mathrm{D}} = \nu$ and portfolio state (exercise state of the derivatives in the portfolio) $A_t = \alpha$, is given by

$$\begin{aligned}
&\Pi^A(t,x,\nu,\alpha) \\
&\quad := \nu \sup_{\boldsymbol{\tau}\in\boldsymbol{\mathcal{T}'}(t)} \mathbb{E}_{t,x,1,a}\Bigg[\sum_{j=1}^{J} \alpha_j \mathbb{1}_{\tau_j}^{\mathrm{D}} D_{t,\tau_j} g_j(\tau_j, X_{\tau_j}) + D_{t,\tau^{\mathrm{D}}}\Bigg(R\Bigg(\sum_{k=1}^{J} \alpha_k (1-\mathbb{1}_{\tau_k}^{\mathrm{D}}) V_k(\tau^{\mathrm{D}}, X_{\tau^{\mathrm{D}}})\Bigg)^+ \\
&\qquad + \Bigg(\sum_{\ell=1}^{J} \alpha_\ell (1-\mathbb{1}_{\tau_\ell}^{\mathrm{D}}) V_\ell(\tau^{\mathrm{D}}, X_{\tau^{\mathrm{D}}})\Bigg)^-\Bigg)\Bigg],
\end{aligned} \tag{3.19}$$

where $\mathbb{E}_{t,x,\nu,\alpha}[\,\cdot\,] = \mathbb{E}^{\mathbb{Q}}[\,\cdot \,|\, X_t = x, \mathbb{1}_t^{\mathrm{D}} = \nu, A_t = \alpha]$. To emphasize the importance, we put the following observations about (3.19) into two remarks.

Remark 3.2.1. *The optimal stopping strategies of the individual derivatives in the portfolio are no longer independent of each other, as in* (3.5) *and* (3.6)*. Furthermore, the optimal strategy depends on earlier exercise decisions, meaning that in order to make the exercise decisions Markovian, we need to include information about earlier decisions. The reason for this is the non-linearity in the two sums inside the expectation in* (3.19)*. Therefore, the optimal stopping strategies need to be computed for the entire portfolio simultaneously. To the best of our knowledge, this has not been done in an ordinary least squares setting before. However, it is discussed in a PDE framwork in the case of a portfolio of American swaptions in [56].*

Remark 3.2.2. *The value of the risky portfolio with netting, depends on the J risk-free derivative values and we are therefore required to approximate the risk-free derivative values. The reason for this is that, at default, the risk-free value of the portfolio is used as reference value in the close-out agreement (see Equation* (3.19)*). If we restrict our portfolio to derivatives with positive pay-off functions, then V_j can be replaced by g_j (by the definition of V_j and the law of iterated expectations). Furthermore, in the restricted portfolio, the values with and without netting coincide.*

An $(X^{t,x}, A^{t,\alpha})-$stopping times vector can be defined by

$$\boldsymbol{\tau}^n[\boldsymbol{f}](X^{t,x}, A^{t,x}) := \sum_{k=n}^{N} T_k \boldsymbol{f}(T_k, X_k, A_k) \odot \prod_{M=n}^{k-1} (\mathbf{1}_J - \boldsymbol{f}(T_m, X_m, A_m)), \tag{3.20}$$

where $\odot$ is element-wise multiplication and $\mathbf{1}_J$ is the $J-$dimensional vector with only ones, $(1,1,\ldots,1)^T$. We denote element j of the stopping times vector by $\tau_j^n[\boldsymbol{f}](X^{t,x}, A^{t,\alpha}) = \left(\boldsymbol{\tau}^n[\boldsymbol{f}](X^{t,x}, A^{t,x})\right)_j$. We emphasize that each element of the stopping time vector depends on $\boldsymbol{f}$ and not only an element j which is the case without netting. Similar to (3.11), we introduce a short-hand notation, which simplifies the valuation function

$$\hat{\boldsymbol{\tau}}_n := \boldsymbol{\tau}^n[\boldsymbol{f}](X^{t,x}, A^{t,x}), \quad \hat{\tau}_{n,j} := (\hat{\boldsymbol{\tau}}_n)_j. \tag{3.21}$$

We here use "$\hat{\tau}$ ", instead of "$\bar{\tau}$" as in (3.11), to emphasize that the stopping time also takes $A^{t,\alpha}$ as an argument. The netted risky portfolio value, given the exercise strategy obtained by

decision function $\boldsymbol{f}$, is then given by

$$\Upsilon^A(t,x,\nu,\alpha \,|\, \boldsymbol{f}) \coloneqq \nu \mathbb{E}_{t,x,1,\alpha}\Bigg[\sum_{j=1}^{J} \alpha_j \mathbb{1}^{\mathrm{D}}_{\hat{\tau}_{n,j}} D_{t,\hat{\tau}_{n,j}} g_j(\hat{\tau}_{n,j}, X_{\hat{\tau}_{n,j}}) \\ + D_{t,\tau^{\mathrm{D}}}\left(R\left(\sum_{j=1}^{J} \alpha_j (1-\mathbb{1}^{\mathrm{D}}_{\hat{\tau}_{n,j}}) V_j(\tau^{\mathrm{D}}, X_{\tau^{\mathrm{D}}})\right)^+ + \left(\sum_{j=1}^{J} \alpha_j (1-\mathbb{1}^{\mathrm{D}}_{\hat{\tau}_{n,j}}) V_j(\tau^{\mathrm{D}}, X_{\tau^{\mathrm{D}}})\right)^-\right)\Bigg], \tag{3.22}$$

where we, again, remind ourselves that the exercise strategy is controlled by $\boldsymbol{f}$, and for $u \geq t$, the event $\mathbb{I}_{\{\hat{\tau}_{t,j} \leq u\}}$ is $\sigma(X^{t,x}, A^{t,\alpha})-$measurable.

For a netted portfolio, the optimal exercise regions, described in Section 3.2.2, are less trivial. Firstly, they become dependent on the state of earlier exercise decisions, $A_t = \alpha_t \in \{0,1\}^J$. Secondly, the exercise region for derivative $j \in \{1,2,\ldots,J\}$ is expressed under the condition that an optimal exercise strategy for the other $J-1$ derivatives is applied. Therefore, we only describe the optimal decision function as belonging to the supremum over the space, $\mathcal{D}$, of all measurable functions, $f\colon [0,T] \times \mathbb{R}^d \times \{0,1\}^J \to \{0,1\}^J$,

$$\boldsymbol{f}^A \in \arg\max_{\boldsymbol{f} \in \mathcal{D}} \Upsilon^A(0, x_0, \nu_0, \alpha_0 \,|\, \boldsymbol{f}), \tag{3.23}$$

where $\nu_0 = 1$ (no default prior to or at $t=0$) and $a_0 = (1,1,\ldots,1)^T$ (no derivatives have been exercised prior to $t=0$). We then assume that, given the state $(t, X_t = x, \mathbb{1}^{\mathrm{D}}_t = \nu, A_t = \alpha)$, the following holds

$$\Upsilon^A(t,x,\nu,\alpha \,|\, \boldsymbol{f}^A) = \Pi^A(t,x,\nu,\alpha). \tag{3.24}$$

Similar to (3.17), when we want to emphasize the particular choice of decision function, $\boldsymbol{f}^A$, we use the short-hand notation

$$\hat{\boldsymbol{\tau}}^A_n = \boldsymbol{\tau}^n[\boldsymbol{f}^A](X^{t,x}, A^{t,\alpha}) \quad \text{and} \quad \hat{\tau}^A_{n,j} = \left(\boldsymbol{\tau}^n[\boldsymbol{f}^A](X^{t,x}, A^{t,\alpha})\right)_j, \tag{3.25}$$

where it is assumed that $t \in (T_{n-1}, T_n]$.

3.2.4 Credit valuation adjustment of a derivative portfolio

The formal definition of CVA is the difference between the risk-free and the risky portfolio value. Given models of the underlying market and default events of our counterparty, the above definition of CVA is straight-forward for a portfolio consisting of derivatives without optionality *e.g.,* European options, barrier options etc. When it comes to portfolios consisting of derivatives with true optionality, *e.g.,* the Bermudan options, American options etc. the standard procedure is not clear. In this section, we define the CVA for portfolios of derivatives with true optionality as well as some approximations, which simplify the computations. In the definitions of CVA, we use the portfolio valuations in terms of optimally chosen decision functions given in equations (3.14) and (3.24). The CVA at $(t=0, X_0 = x_0)$, with and without

netting, respectively, are given by

$$\text{CVA} \coloneqq \Upsilon^V(0, x_0 \,|\, \boldsymbol{f}^V) - \Upsilon^U(0, x_0, 1 \,|\, \boldsymbol{f}^U), \quad \text{(without netting)}, \tag{3.26}$$

$$\text{CVA}^{\text{Net}} \coloneqq \Upsilon^V(0, x_0 \,|\, \boldsymbol{f}^V) - \Upsilon^A(0, x_0, 1, \mathbf{1}_J \,|\, \boldsymbol{f}^A), \quad \text{(with netting)}. \tag{3.27}$$

A commonly used approximation is to apply the same exercise strategy to the risk-free and risky portfolios. One such approximation is defined as

$$\overline{\text{CVA}} \coloneqq \Upsilon^V(0, x_0 \,|\, \boldsymbol{f}^V) - \Upsilon^U(0, x_0, 1 \,|\, \boldsymbol{f}^V), \quad \text{(Risk-free strategy, without netting)}, \tag{3.28}$$

$$\overline{\text{CVA}}^{\text{Net}} \coloneqq \Upsilon^V(0, x_0 \,|\, \boldsymbol{f}^V) - \Upsilon^A(0, x_0, 1, \mathbf{1}_J \,|\, \boldsymbol{f}^V), \quad \text{(Risk-free strategy, with netting)}. \tag{3.29}$$

The only difference between (3.26)-(3.27) and (3.28)-(3.29) is that in the latter the risk-free strategy is used also for the risky portfolios. One could also think of other definitions, *e.g.,* using the risky strategies for both portfolios. This particular choice is motivated by the fact that $\boldsymbol{f}^U$ and $\boldsymbol{f}^A$ are, in general, dependent on $\boldsymbol{f}^V$ through the close-out agreements in (3.13) and (3.22). Moreover, $\boldsymbol{f}^V$ is a sub-optimal strategy for both risky portfolios leading to $\text{CVA} \leq \overline{\text{CVA}}$, and $\text{CVA}^{\text{Net}} \leq \overline{\text{CVA}}^{\text{Net}}$, which is beneficial for the bank (but certainly not for the counterparty). If we instead use only the risky descision functions, *i.e.,* replacing $\boldsymbol{f}^V$ with $\boldsymbol{f}^U$ in (3.28) and $\boldsymbol{f}^A$ in (3.29), we would obtain an underestimation of the CVAs, which would be unacceptable for the bank.

As mentioned in the Introduction, the bank is exposed to the risk of CVA losses, as a consequence of the MtM value of the CVA moving against the bank. We therefore want to follow the evolution of the CVA over time, to gain insights in its distribution. Of particular interest is the tail distribution of the CVA, for times between initial time and the maturity of the portfolio. To explore this, we define the dynamic versions of (3.26)-(3.29), which are stochastic processes depending on the market and portfolio state processes X and A. For $t \in [0, T]$, the dynamic versions of the CVAs (and their approximations) are given by the following random variables

$$\begin{aligned}
\text{CVA}(t, X_t, A_t) &\coloneqq \sum_{j=1}^{J} \left(\mathcal{V}_j(t, X_t \,|\, f_j^V) - \mathcal{U}_j(t, X_t \,|\, f_j^U)\right)(A_t)_j, \\
\text{CVA}^{\text{net}}(t, X_t, A_t) &\coloneqq \sum_{j=1}^{J} \mathcal{V}_j(t, X_t \,|\, f_j^V)(A_t)_j - \Upsilon^A(t, X_t, 1, A_t \,|\, \boldsymbol{f}^A), \\
\overline{\text{CVA}}(t, X_t, A_t) &\coloneqq \sum_{j=1}^{J} \left(\mathcal{V}_j(t, X_t \,|\, f_j^V) - \mathcal{U}_j(t, X_t \,|\, f_j^V)\right)(A_t)_j, \\
\overline{\text{CVA}}^{\text{net}}(t, X_t, A_t) &\coloneqq \sum_{j=1}^{J} \mathcal{V}_j(t, X_t \,|\, f_j^V)(A_t)_j - \Upsilon^A(t, X_t, 1, A_t \,|\, \boldsymbol{f}^V).
\end{aligned}$$

In the above, the CVA is conditional on that the counterparty has not defaulted prior to, or at, t (it does not make sense to calculate the CVA if the counterparty has already defaulted). From the above we can define the Expected value of the CVA (E-CVA), and for $\alpha \in (0, 1)$, the $\alpha-$level of Value at Risk of the CVA (VaR-CVA) and Expected Shortfall of the CVA

(ES-CVA),

$$\text{E-CVA}(t) \coloneqq \mathbb{E}\left[\text{CVA}(t, X_t, A_t) \,|\, \mathbb{1}_t = 1\right], \tag{3.30}$$

$$\text{VaR-CVA}_\alpha(t) \coloneqq \inf\Big\{P \in \mathbb{R} \,|\, \mathbb{Q}(\text{CVA}(t, X_t, A_t) \leq P) \geq \alpha\Big\}, \tag{3.31}$$

$$\text{ES-CVA}_\alpha(t) \coloneqq \mathbb{E}\big[\text{CVA}(t, X_t, A_t) \,|\, \mathbb{1}_t = 1\,,\, \text{CVA}(t, X_t, A_t) \geq \text{VaR-CVA}_\alpha(t)\big]. \tag{3.32}$$

In a similar way E-$\overline{\text{CVA}}(t)$, ES-$\overline{\text{CVA}}_\alpha(t)$, E-CVA$^{\text{net}}(t)$, ES-CVA$^{\text{net}}_\alpha(t)$, E-$\overline{\text{CVA}}^{\text{net}}(t)$ and ES-$\overline{\text{CVA}}^{\text{net}}_\alpha(t)$ are defined. The expression for the ES-CVA looks complicated but is basically just the expected value of the $\alpha-$tail of the CVA distribution. We focus on ES-CVA instead of VaR-CVA because it is a coherent risk measure and VaR-CVA is not.

Remark 3.2.3. *Since* ES-CVA *is a non-traded risk measure, it should ideally be computed under the real world measure* $\mathbb{P}$*, see e.g., [58] for a detailed discussion. To be precise,* (X_t, A_t) *should be generated under the* $\mathbb{P}-$*measure and, the* CVA*, which is a tradeable asset, should be computed under the* $\mathbb{Q}-$*measure. It is straight-forward to adjust the algorithms in this chapter be able to compute* ES-CVA *under the* $\mathbb{P}-$*measure, see Chapter 2 for details in the special case* $J = 1$.

3.2.5 Exposure profiles

In this subsection we discuss the concept of exposure profiles for a portfolio of derivatives. The financial exposure (of the bank) is defined as the maximum amount the bank stands to loose if the counterparty defaults. The exposure profile is loosely defined as the distribution of the exposure over time. The exposures, with and without netting, are defined as

$$\text{E}^{\text{Net}}_t \coloneqq \max\Big\{\sum_{j=1}^{J} V_j(t, X_t)(A_t)_j,\, 0\Big\}, \quad \text{E}_t \coloneqq \sum_{j=1}^{J} \max\left\{V_j(t, X_t)(A_t)_j,\, 0\right\},$$

where we recall that $(A_t)_j = \mathbb{I}_{\{\tau_j > t\}}$ with τ_j being the exercise date for derivative j. Furthermore, for a portfolio without netting, the expected exposure (EE), and for $\alpha \in (0, 1)$, the potential future exposure (PFE) are defined as

$$\text{EE}(t) \coloneqq \mathbb{E}_0[D_{0,t}\,\text{E}_t], \tag{3.33}$$

$$\text{PFE}_\alpha(t) \coloneqq \inf\left\{P \in \mathbb{R} \,|\, \mathbb{Q}(D_{0,t}\,\text{E}_t \leq P) \geq \alpha\right\}. \tag{3.34}$$

Both the expectation and the probability in (3.33) and (3.34) should be interpreted as conditional on $X_0 = x_0 \in \mathbb{R}^d$. The EE and PFE in the presence of netting, denoted by $\text{EE}^{\text{Net}}(\cdot)$ and $\text{PFE}^{\text{Net}}_\alpha(\cdot)$, and are obtained by instead using the netted exposure in (3.33) and (3.34).

If we assume a constant recovery rate $R \in [0, 1)$, and that X and $\mathbb{1}^{\text{D}}$ are independent, *i.e.,* the default event of the counterparty is independent of the risk factors, then (3.28) and (3.29) can be written as

$$\overline{\text{CVA}} = (1-R)\int_0^T \text{EE}(t)\mathbb{Q}\left(\tau^{\text{D}} \in [t + \mathrm{d}t)\right), \quad \overline{\text{CVA}}^{\text{Net}} = (1-R)\int_0^T \text{EE}^{\text{Net}}(t)\mathbb{Q}\left(\tau^{\text{D}} \in [t + \mathrm{d}t)\right),$$

which can be approximated by

$$\begin{aligned}\overline{\mathrm{CVA}} &\approx (1-R)\sum_{m=1}^{M}\mathrm{EE}(t_m)\mathbb{Q}\left(\tau^{\mathrm{D}}\in(t_{m-1},t_m]\right),\ \overline{\mathrm{CVA}}^{\mathrm{Net}}\\ &\approx (1-R)\sum_{m=1}^{M}\mathrm{EE}^{\mathrm{Net}}(t_m)\mathbb{Q}\left(\tau^{\mathrm{D}}\in(t_{m-1},t_m]\right),\end{aligned}$$

for some partition of $[0,T]$, with $t_0=0$ and $t_M=T$. The above formulations require access to the density of default events, but may be more accurate, especially for large M and a small probability of default (with a simulation-based approach, problems with a low probability of default can often be tackled with variance reduction techniques).

3.3 Algorithms

In the first part of this section, we present a neural network-based method to approximate the decision functions introduced in the previous section. The method generalizes the Deep Optimal Stopping proposed in [9] and extended in Chapter 2, which approximates stopping decisions for a single derivative, to be applicable also for portfolios of derivatives with early-exercise features. Furthermore, for the risky portfolios, the algorithm is extended to be able to deal with default risk of the counterparty. The algorithm is based on a series of neural networks, which are optimized backwards in time with the objective to maximize the expected discounted cash-flows.

In the second part of this section, the exercise policy obtained from the approximate decision functions is applied pathwise on realizations of the risk factors of each derivative in the portfolio to generate pathwise cash-flows. These cash-flows are used in a neural network-based regression algorithm to approximate pathwise derivative values. These pathwise derivative values can then be used to compute important risk management measures.

3.3.1 Phase I: Learning exercise strategy

As indicated above, the core of the algorithm is to approximate decision functions, in order to obtain good approximations of the value of a portfolio of derivatives. We approximate the decision functions $\boldsymbol{f}^V$, $\boldsymbol{f}^U$ and $\boldsymbol{f}^A$, with fully connected neural networks. To be more precise, let $N=|\mathbb{T}^{\Pi}(0)|$, for $n\in\{1,2,\ldots,N\}$ and for $Z\in\{V,U\}$, the decision function $\boldsymbol{f}^Z(T_n,\cdot)$, is approximated by a fully connected neural network of the form $\boldsymbol{f}^{\theta_n}\colon\mathbb{R}^d\to\{0,\,1\}^J$, where $\theta_n\in\mathbb{R}^{q_n}$ is a vector containing all the $q_n\in\mathbb{N}$ trainable parameters in network n. The decision function $\boldsymbol{f}^A(T_n,\cdot,\cdot)$ is approximated by similar neural networks, with the only difference that the input also includes information of which derivatives in the portfolio have been exercised prior to T_n, *i.e.,* $\boldsymbol{f}^{\theta_n}\colon\mathbb{R}^d\times\{0,1\}^J\to\{0,\,1\}^J$.

Since binary decision functions are discontinuous, and therefore unsuitable for gradient-type optimization algorithms, we use as an intermediate step, the neural network $\boldsymbol{F}^{\theta_n}\colon\mathbb{R}^d\to(0,\,1)^J$. Instead of a binary decision, the output of the neural network $\boldsymbol{F}^{\theta_n}$ can be viewed as the probability[4] for exercise to be optimal. This output is then mapped to 1 for values above (or

[4]However the interpretation as a probability may be helpful, one should be careful since it is not a rigorous mathematical statement. It should be clear that there is nothing random about the stopping decisions, since

equal to) 0.5, and to 0 otherwise, by defining $\boldsymbol{f}^{\theta_n}(\cdot) = \mathfrak{a} \circ \boldsymbol{F}^{\theta_n}(\cdot)$, where $\mathfrak{a}$ is a component-wise round-off function, *i.e.,* for $j \in \{1, 2, \ldots, J\}$, and $x \in \mathbb{R}^d$, the j:th component of $\mathfrak{a}(x)$ is given by $(\mathfrak{a}(x))_j = \mathbb{I}_{\{x_j \geq 1/2\}}$. For each $Z \in \{V, U\}$, our aim is to adjust the parameters $\theta_1, \theta_2, \ldots, \theta_N$ such that

$$(\boldsymbol{f}^Z(T_1, \cdot), \boldsymbol{f}^Z(T_2, \cdot), \ldots, \boldsymbol{f}^Z(T_N, \cdot))^T \approx (\boldsymbol{f}^{\theta_1}, \boldsymbol{f}^{\theta_2}, \ldots, \boldsymbol{f}^{\theta_N})^T =: \mathbb{f}^\Theta, \tag{3.35}$$

$$(\boldsymbol{f}^A(T_1, \cdot, \cdot), \boldsymbol{f}^A(T_2, \cdot, \cdot), \ldots, \boldsymbol{f}^A(T_N, \cdot, \cdot)),^T \approx (\boldsymbol{f}^{\theta_1}, \boldsymbol{f}^{\theta_2}, \ldots, \boldsymbol{f}^{\theta_N})^T =: \mathbb{f}^\Theta, \tag{3.36}$$

where we recall that $\Theta = \{\theta_1, \theta_2, \ldots, \theta_N\}$. For $n \in \{1, 2, \ldots, N\}$, we define the sequence of neural networks, approximating the decision functions at exercise dates $T_n, T_{n+1}, \ldots T_N$, by $\mathbb{f}_n^\Theta \coloneqq (\boldsymbol{f}^{\theta_n}, \boldsymbol{f}^{\theta_{n+1}}, \ldots, \boldsymbol{f}^{\theta_N})^T$. Note that the input dimension for the neural networks is different when we want to approximate $\boldsymbol{f}^V$ and $\boldsymbol{f}^U$ compared to when we want to approximate $\boldsymbol{f}^A$. To avoid having to introduce an extra layer of notation, we use for all networks Θ to denote the set of parameters, and keep in mind that the dimension depends on the specific problem considered. Although the above provides a good intuition for what we want to accomplish, it is not clear in which sense we want the functions to be similar, or how to adjust the parameters to achieve this. To approach a more tractable form, from a computational perspective, for $t \in (T_{n-1}, T_n]$, we insert (3.35) in (3.10) and (3.36) in (3.20) to obtain

$$\boldsymbol{\tau}[\mathbb{f}_n^\Theta](X^{t,x}) = \sum_{k=n}^{N} T_k \boldsymbol{f}^{\theta_k}(X_k) \odot \prod_{m=k}^{N} \left(\mathbf{1}_J - \boldsymbol{f}^{\theta_m}(X_m)\right), \tag{3.37}$$

$$\boldsymbol{\tau}[\mathbb{f}_n^\Theta](X^{t,x}, A^{t,\alpha}) = \sum_{k=n}^{N} T_k \boldsymbol{f}^{\theta_k}(X_k, A_k) \odot \prod_{m=k}^{N} \left(\mathbf{1}_J - \boldsymbol{f}^{\theta_m}(X_m, A_m)\right). \tag{3.38}$$

Note that (3.37) is a $J-$dimensional vector of $X-$stopping times and (3.38) is a $J-$dimensional $(X, A)-$stopping times vector, which depends on $\mathbb{f}_n^\Theta$ on a structural level but also on the randomness of the stochastic process $X^{t,x}$ (and $A^{t,\alpha}$ for (3.38)). For notational convenience, we use the short hand notation $\bar{\boldsymbol{\tau}}_n^\Theta = \boldsymbol{\tau}[\mathbb{f}_n^\Theta](X^{t,x})$ (or $\hat{\boldsymbol{\tau}}_n^\Theta = \boldsymbol{\tau}[\mathbb{f}_n^\Theta](X^{t,x}, A^{t,\alpha})$, when approximating $\boldsymbol{f}^A$), and for element $j \in \{1, 2, \ldots, J\}$, $\bar{\tau}_{n,j}^\Theta = (\bar{\boldsymbol{\tau}}_n^\Theta)_j$ (or $\hat{\tau}_{n,j}^\Theta = (\hat{\boldsymbol{\tau}}_n^\Theta)_j$). We are now ready to define our objective, which, for $T_n \in \mathbb{T}^\Pi(0)$, is to find θ_n such that the expected future cash-flows are maximized. The cash-flows can be divided into three categories:

1. The cash-flows obtained by the derivatives exercised at the present time T_n;
2. The cash-flows obtained at later exercise dates prior to default of the counterparty;
3. The cash-flows obtained at default of the counterparty, according to the close-out agreement.

For $n \in \{1, 2, \ldots, N\}$ and $j \in \{1, 2, \ldots, J\}$, we denote dimension j of decision function $\boldsymbol{f}^{\theta_n}$ by

$$(\boldsymbol{f}^{\theta_n})_j = f_j^{\theta_n}, \quad \text{and } (\boldsymbol{F}^{\theta_n})_j = F_j^{\theta_n}$$

the stopping time is $\mathcal{H}_t-$measurable. It can also be interpreted as a measure on how certain we can be that exercise is optimal.

Given that no default has occurred prior to T_n, (and the exercise state $A_n = \alpha$ for the risky portfolio with netting) the expected cash-flows, that we want to maximize, are given below.

Risk-free portfolio:

$$\mathbb{E}_{T_n}\left[\sum_{j=1}^{J} f_j^{\theta_n}(X_n)\, g_j(T_n, X_n) + (1 - f_j^{\theta_n}(X_n)) D_{T_n, \bar{\tau}^V_{n+1,j}} g_j\Big(\bar{\tau}^V_{n+1,j}, X_{\bar{\tau}^V_{n+1,j}}\Big)\right], \tag{3.39}$$

Risky portfolio without netting:

$$\begin{aligned}\mathbb{E}_{T_n}\Bigg[&\sum_{j=1}^{J} f_j^{\theta_n}(X_n)\, g_j(T_n, X_n) + (1 - f^{\theta_n}(X_n))\Big(\mathbb{1}^{\mathrm{D}}_{\bar{\tau}^U_{n+1,j}} D_{T_n, \bar{\tau}^U_{n+1,j}} g_j\Big(\bar{\tau}^U_{n+1,j}, X_{\bar{\tau}^U_{n+1,j}}\Big) \\ &+ \Big(1 - \mathbb{1}^{\mathrm{D}}_{\bar{\tau}^U_{n+1,j}}\Big) D_{t,\tau^{\mathrm{D}}}\big(RV_j(\tau^{\mathrm{D}}, X_{\tau^{\mathrm{D}}})^+ + V_j(\tau^{\mathrm{D}}, X_{\tau^{\mathrm{D}}})^-\big)\Big)\Bigg],\end{aligned} \tag{3.40}$$

Risky portfolio with netting:

$$\begin{aligned}\mathbb{E}_{T_n}\Bigg[&\sum_{j=1}^{J} \alpha_j f_j^{\theta_n}(X_n, \alpha)\, g_j(T_n, X_n) + \alpha_j (1 - f_j^{\theta_n}(X_n, \alpha))\, \mathbb{1}^{\mathrm{D}}_{\hat{\tau}^U_{n+1,j}} D_{T_n, \hat{\tau}^U_{n+1,j}} g_j\Big(\hat{\tau}^U_{n+1,j}, X_{\hat{\tau}^U_{n+1,j}}\Big) \\ &+ R\Big(\sum_{k=1}^{J} \alpha_k (1 - f_k^{\theta_n}(X_n, \alpha))\Big(1 - \mathbb{1}^{\mathrm{D}}_{\hat{\tau}^U_{n+1,k}}) D_{t,\tau^{\mathrm{D}}} V_k(\tau^{\mathrm{D}}, X_{\tau^{\mathrm{D}}})\Big)^+ \\ &+ \Big(\sum_{\ell=1}^{J} \alpha_\ell (1 - f_\ell^{\theta_n}(X_n, \alpha))\Big(1 - \mathbb{1}^{\mathrm{D}}_{\hat{\tau}^U_{n+1,\ell}}\Big) D_{t,\tau^{\mathrm{D}}} V_\ell(\tau^{\mathrm{D}}, X_{\tau^{\mathrm{D}}})\Big)^-\Bigg].\end{aligned} \tag{3.41}$$

We want to optimize θ_n, such that the above are as close as possible (in mean squared sense) to $\Pi^V(T_n, X_n)$, $\Pi^U(T_n, X_n, 1)$ and $\Pi^A(T_n, X_n, 1, \alpha)$, respectively.

Remark 3.3.1. *The objectives for the risky portfolios, in* (3.40) *and* (3.41)*, both depend on the risk-free valuation of the derivatives. Therefore, in order to approximate the risky decision functions, we first need to approximate the risk-free exercise strategy, and the risk-free derivative values. In the next subsection, we explain how V_j can be approximated.*

Although (3.39)-(3.41) are accurate representations of the optimization problems, they give us some practical problems. In general, we have no access to $\boldsymbol{f}^V$, $\boldsymbol{f}^U$ and $\boldsymbol{f}^A$ which control $\bar{\boldsymbol{\tau}}^V_{n+1}$, $\bar{\boldsymbol{\tau}}^U_{n+1}$ and $\hat{\boldsymbol{\tau}}^A_{n+1}$. Another problem is that, in general, we have no access to the true distributions of the portfolio values for comparison. However, if $T_{\tilde{N}}$ is the maturity of derivative $j \in \{1, 2, \ldots, J\}$, it is optimal to exercise as long as the pay-off value is positive, by the definition of the decision functions. We can therefore set

$$f_j^{\theta_{\tilde{N}}}(\,\cdot\,) \coloneqq \mathbb{I}_{\{g_j(T_{\tilde{N}}, \,\cdot\,) > 0\}}, \quad \text{and} \quad f_j^{\theta_k}(\,\cdot\,) \coloneqq 0, \text{ for } k > \tilde{N}.$$

Furthermore, at T_N, the maturity of the portfolio, the positive part of the pay-off value equals the derivative value (if no default in $(T_{N-1}, T_N]$, in the risky cases). At T_{N-1}, Equation (3.39)

then becomes

$$\mathbb{E}_{T_{N-1}}\Big[\sum_{j=1}^{J} f_j^{\theta_{N-1}}(X_N)\, g_j(T_{N-1}, X_{N-1}) + D_{N-1,N}(1 - f_j^{\theta_{N-1}}(X_{N-1}))\, g_j(T_N, X_N)\Big]. \quad (3.42)$$

Recall that if T_N is greater than the maturity of contract j, we have $g_j(T_N, \cdot) \equiv 0$. Since all components in (3.42) are known except for the decision function $\boldsymbol{f}^{\theta_{N-1}}$, we want to find θ_{N-1}, such that a Monte-Carlo approximations of (3.42) is maximized. Given $M \in \mathbb{N}$ samples, distributed as X, which for $m \in \{1, 2, \ldots, M\}$ is denoted by $x = (x_t(m))_{t\in[0,T]}$, we approximate (3.42) by

$$\frac{1}{M}\sum_{m=1}^{M}\sum_{j=1}^{J} f_j^{\theta_{N-1}}(x_{N-1}(m)), g_j(T_{N-1}, x_{N-1}(m)) + D_{N-1,N}(1 - f_j^{\theta_{N-1}}(x_{N-1}(m)))\, g_j(T_N, x_N(m)). \quad (3.43)$$

The only unknown entity in (3.43) is the parameter θ_{N-1} in the decision function $\boldsymbol{f}^{\theta_{N-1}} = (f_1^{\theta_{N-1}}, \ldots, f_j^{\theta_{N-1}})^T$. Furthermore, we wish to find θ_{N-1} such that (3.43) is maximized, since it represents the average cash-flow in $[t_{N-1}, t_N]$. Once θ_{N-1} is optimized, we use this parameter to set up a similar expression for the expected cash-flow on $[t_{N-2}, t_N]$, which is maximized by finding an optimal θ_{N-2}. This procedure is then iteratively continued until also $\theta_{N-3}, \theta_{N-4}, \ldots, \theta_1$ are optimized. The procedure is similar for the risky portfolios, but based on (3.40) or (3.41) instead. This implies that we also need to sample default events of the counterparty. We denote by θ_n^* the optimized version of parameter θ_n and the sequence of optimized parameters for the networks at exercise dates $T_n, T_{n+1}, \ldots, T_N$ are defined as

$$\Theta_n^* \coloneqq \{\theta_n^*, \theta_{n+1}^*, \ldots, \theta_N^*\},$$

and for notational convenience, we define the complete sequence of parameters as $\Theta^* \coloneqq \Theta_1^*$.

Remark 3.3.2. *Since we are considering a portfolio in which all the derivatives may have a different set of exercise dates, we have that for $T_n \in \mathbb{T}^{\Pi}(0)$, there are $J_n^{\mathrm{Ex}} \in \{1, 2, \ldots, J\}$ derivatives that may be exercised. Therefore, we only need to compute J_n^{Ex} of the J dimensions of $\boldsymbol{f}^{\theta_n}$ and can by default set the remaining $J - J_n^{\mathrm{Ex}}$ dimensions of $\boldsymbol{f}^{\theta_n}$ to 0. This can be done by appropriately adjusting some weights and biases.*

To keep the flow of the chapter, the details of the algorithms and the parameters θ_n are given in the Appendix.

3.3.2 Phase II: Learning pathwise derivative values and portfolio exposures

As mentioned in Remark 3.3.1, the risk-free derivative values need to be approximated pathwise in order to approximate the risky decision functions. Moreover, the pathwise derivative values are required to approximate the exposure profiles for the risk-free, as well as the risky portfolios. In this subsection, we focus on the risk-free portfolios, but the extension to risky portfolios is straight-forward.

We use the risk-free, stopping strategy from subsection 3.3.1 to generate pathwise cash-flows, which are in turn used to approximate the pathwise derivative values. Let $T_{n-1}, T_n \in \mathbb{T}^{\Pi}(0)$,

for $t \in (T_{n-1}, T_n]$, we denote the vector-valued discounting process and pay-off function, respectively, by

$$\boldsymbol{D}(t, \boldsymbol{\tau}_n^V) \coloneqq \begin{pmatrix} D(t, \tau_{n,1}^V) \\ \vdots \\ D(t, \tau_{n,J}^V) \end{pmatrix} \quad \text{and,} \quad \boldsymbol{g}(\boldsymbol{\tau}_n^V, X^{t,x}) \coloneqq \begin{pmatrix} g_1(\tau_{n,1}^V, X_{\tau_{n,1}^V}) \\ \vdots \\ g_J(\tau_{n,J}^V, X_{\tau_{n,J}^V}) \end{pmatrix}. \tag{3.44}$$

Using the notation above, we define the vector-valued cash-flow process as

$$\boldsymbol{Y}_t \coloneqq \boldsymbol{D}_{t,\boldsymbol{\tau}_n^V} \odot \boldsymbol{g}(\boldsymbol{\tau}_n^V, X_{\boldsymbol{\tau}_n^V}), \tag{3.45}$$

and for $j \in \{1, 2, \ldots, J\}$, we denote the j:th element of $\boldsymbol{Y}_t$ by $Y_{t,j}$, and we emphasize that $\boldsymbol{Y}_t$ is not $\mathcal{H}_t-$measurable.

3.3.2.1 Regression problems

In this subsection we use standard regression theory, see *e.g.,* [45], to show that derivative values, exposures, and other entities of interest, can be formulated as the solution to certain minimization problems. We introduce the following notation for measurable functions[5]

$$\mathcal{D}(C_1; C_2) \coloneqq \{\, f \colon C_1 \to C_2 \,|\, f \text{ measurable}\}. \tag{3.46}$$

First, we recall a basic property of the regression function. Let $\mathcal{X} \colon [0,T] \times \Omega \to C_1$ and $\mathcal{Y} \colon [0,T] \times \Omega \to C_2$ be a stochastic process, which for $t, u \in [0,T]$, with $t \leq u$, satisfies $\mathbb{E}_0[|\mathcal{Y}_t|^2] < \infty$. We define the regression function, which satisfies

$$m(t, \cdot) \in \underset{\boldsymbol{h} \in \mathcal{D}(C_1; C_2)}{\arg\min}\ \mathbb{E}_t\left[\|\boldsymbol{h}(\mathcal{X}_t) - \mathcal{Y}_u\|_2^2\right], \tag{3.47}$$

where $\|\cdot\|_2$ is the Euclidean norm. It then holds, for $x \in \mathbb{R}^a$, that the regression function is given by the conditional expectation

$$m(t, \chi) = \mathbb{E}^{\mathbb{Q}}[\mathcal{Y}_u \,|\, \mathcal{X}_t = \chi]. \tag{3.48}$$

Using the notation from above, and by choosing C_1, C_2, $\mathcal{X}$ and $\mathcal{Y}$ in (3.47) and (3.48) wisely, we can approximate different entities related to *e.g.,* the exposure profiles or the pathwise CVA. For instance the exposures, both with and without netting, can be approximated from the solution of a minimization problem of the form in (3.47). For $T_{n-1}, T_n \in \mathbb{T}^{\Pi}(0)$, let $t \in (T_{n-1}, T_n]$ and denote $\boldsymbol{V}(t, \cdot) = (V_1(t, \cdot), \ldots V_J(t, \cdot))^T$, it then holds that

$$\boldsymbol{V}(t, \cdot) \in \underset{\boldsymbol{h} \in \mathcal{D}(\mathbb{R}^d; \mathbb{R}^J)}{\arg\min}\ \mathbb{E}_t\left[\|\boldsymbol{h}(X_t) - \boldsymbol{Y}_t\|_2^2\right], \tag{3.49}$$

$$\sum_{j=1}^{J} V_j(t, \cdot)(A_t)_j \in \underset{h \in \mathcal{D}(\mathbb{R}^d \times \{0,1\}^J; \mathbb{R})}{\arg\min}\ \mathbb{E}_t\left[\left|h(X_t, A_t) - \sum_{j=1}^{J} Y_{t,j}(A_t)_j\right|^2\right]. \tag{3.50}$$

[5]We assume measurable spaces $(C_1, \mathcal{C}_1)$ and $(C_2, \mathcal{C}_2)$ and measurable functions with respect to σ-algebras $\mathcal{C}_1$ and $\mathcal{C}_2$. This assumption holds for all cases in this chapter.

In (3.49), we have $C_1 = \mathbb{R}^d$, $C_2 = \mathbb{R}^J$, $\mathcal{X} = X$ and $\mathcal{Y} = \boldsymbol{Y}$ and in (3.50), we have $C_1 = \mathbb{R}^d \times \{0,1\}^J$, $C_2 = \mathbb{R}$, $\mathcal{X} = (X, A)$ and $\mathcal{Y} = \sum_{j=1}^J Y_{\cdot,j}(A)_j$. For $s \in [0,T]$ and for $j \in \{1,2,\dots,J\}$, by (3.2), it holds that $\mathbb{E}_0\left[|Y_{s,j}|^2\right] < \infty$, and therefore also $\mathbb{E}_0\left[\|\boldsymbol{Y}_s\|_2^2\right] < \infty$. Now, (3.49) holds trivially since by definition $\boldsymbol{V}(t,x) = \mathbb{E}_{t,x}\left[\boldsymbol{D}_{t,\boldsymbol{\tau}_n^V} \odot \boldsymbol{g}(\boldsymbol{\tau}_n^V, X_{\boldsymbol{\tau}_n^V})\right] = \mathbb{E}_{t,x}\left[\boldsymbol{Y}_t\right]$. For (3.50), it follows that

$$\sum_{j=1}^J V_j(t,x)(A_t)_j = \sum_{j=1}^J \mathbb{E}_{t,x}\left[D_{t,\tau_{n,j}^V} g_j(\tau_{n,j}^V, X_{\tau_{n,j}^V})\right](A_t)_j \tag{3.51}$$

$$= \mathbb{E}_{t,x}\left[\sum_{j=1}^J D_{t,\tau_{n,j}^V} g_j(\tau_{n,j}^V, X_{\tau_{n,j}^V})(A_t)_j\right] = \mathbb{E}_{t,x}\left[\sum_{j=1}^J Y_{t,j}(A_t)_j\right], \tag{3.52}$$

where we have used linearity of expectations and the fact that A_t is $\mathcal{H}_t-$measurable.

3.3.3 Neural network-based regression algorithm

Since the specific details of the neural networks are transferable from Appendix 3.5.1 and 3.5.2, this subsection is less detailed. The main idea is to represent $\mathcal{D}(\mathbb{R}^A; \mathbb{R}^b)$ (measurable functions from $\mathbb{R}^A$ to $\mathbb{R}^b$, defined in (3.46)) by a parametrized neural network. A minimization problem of the form (3.47) can then be used as a loss function, which should be minimized by adjusting some set of trainable parameters. However, in general we have no access to $\boldsymbol{\tau}_n^V$ for $n < N$, where $N = |\mathbb{T}^\Pi(0)|$. On the other hand, we can use the exercise strategy from Subsection 3.3.1, *i.e.,* approximate $\boldsymbol{\tau}_n^V$ by $\bar{\boldsymbol{\tau}}_n^{\Theta^*} = \left(\bar{\tau}_{n,1}^{\Theta^*}, \dots, \bar{\tau}_{n,J}^{\Theta^*}\right)^T$. Furthermore, $\mathbb{E}_t[\cdot]$ in (3.48) needs to be approximated by Monte-Carlo samples. We use $M_{\text{reg}} \in \mathbb{N}$ samples, distributed as X and $\boldsymbol{Y}$, which for $m \in \{1,2,\dots,M_{\text{reg}}\}$ are denoted by[6] $x^{\text{reg}}(m) = (x_t^{\text{reg}}(m))_{t\in[0,T]}$ and $\boldsymbol{y}(m) = (\boldsymbol{y}_t(m))_{t\in[0,T]}$. Furthermore, we use

$$A_t(m) \approx A_t^{\Theta_n^*}(m) = \begin{pmatrix} \mathbb{I}_{\left\{\bar{\tau}_{n,1}^{\Theta^*}(m) > t\right\}}, \\ \vdots \\ \mathbb{I}_{\left\{\bar{\tau}_{n,J}^{\Theta^*}(m) > t\right\}} \end{pmatrix},$$

where for $j \in \{1,2,\dots,J\}$, $\bar{\tau}_{n,j}^{\Theta^*}(m) = \left(\boldsymbol{\tau}[\mathbb{f}_n^{\Theta^*}](x^{t,x_t^{\text{reg}}(m)}(m))\right)_j$.

We define for $n \in \{1,2,\dots,N\}$, the neural networks $\boldsymbol{h}^{\Phi_n}\colon \mathbb{R}^d \to \mathbb{R}^J$ and $h^{\Phi_n}\colon \mathbb{R}^d \times \{0,1\}^J \to \mathbb{R}$, which are parametrized by $\Phi_n^{\text{IR}} \in \mathbb{R}^{u_n^{\text{IR}}}$ and $\Phi_n^{\text{PR}} \in \mathbb{R}^{u_n^{\text{PR}}}$ where $u_n^{\text{IR}}, u_n^{\text{PR}} \in \mathbb{N}$ are the number of trainable parameters in each network. We use as loss functions, the empirical counterparts of (3.49) and (3.50), which are given by

$$\text{DOS-IR:} \qquad \frac{1}{M_{\text{reg}}} \sum_{m=1}^{M_{\text{reg}}} \|\boldsymbol{h}^{\Phi_n^1}(x_t^{\text{reg}}(m)) - \boldsymbol{y}_t(m)\|_2^2, \tag{3.53}$$

$$\text{DOS-PR:} \qquad \frac{1}{M_{\text{reg}}} \sum_{m=1}^{M_{\text{reg}}} \left| h^{\Phi_n^2}(x_t^{\text{reg}}(m), A_t^{\Theta_n^*}(m)) - \sum_{j=1}^J y_{t,j}(m)\left(A_t^{\Theta_n^*}(m)\right)_j \right|^2. \tag{3.54}$$

"DOS" in DOS-IR and DOS-PR refers to the fact that the deep stopping strategy used to obtain $\boldsymbol{y}(m)$ is generated by the DOS-algorithm. 'IR' and 'PR' are abbreviations for "individual

[6] In practice we set $x_{\text{reg}} = x_{\text{val}}$, where x_{val} is defined in Phase I.

regression" and "portfolio regression", and refer to the regression at the level of each individual derivative, and the regression at portfolio level given in (3.53) and in (3.54), respectively.

The exact algorithm for computations of pathwise CVA in order to obtain ES-CVA is not presented in details here. However, it is straight-forward to adjust (3.53) and (3.54), to approximate pathwise CVA instead.

The only important adjustment to the structure of the neural networks (details in Appendix 3.5.1) is that we want the output to be unbounded and therefore use the identity as scalar activation function in the output layers.

3.3.4 Combining Phase I and Phase II

Recall that the purpose for using regression at the level of each derivative was to be able to approximate the exposure of a portfolio of derivatives without netting agreement. If we consider derivatives with non-negative value, the definitions of exposures with and without netting agreements coincide. Therefore, only derivatives with non-negative values are considered in this chapter, to be able to compare regression on derivative level with the regression on portfolio level. For $T_{n-1}, T_n \in \mathbb{T}^{\Pi}(0)$, let $t \in (T_{n-1}, T_n]$, we define the following approximators

$$\boldsymbol{V}^{\text{DOS-IR}}\left(t, \cdot \,\middle|\, \Phi_n^{\text{IR}}, \Theta^*\right) \coloneqq \boldsymbol{h}^{\Phi_n^{\text{IR}}}(\cdot), \quad \text{(individual derivative values)}, \tag{3.55}$$

$$\mathbf{E}^{\text{DOS-IR}}\left(t, \cdot \,\middle|\, \Phi_n^{\text{IR}}, \Theta^*\right) \coloneqq \boldsymbol{h}^{\Phi_n^{\text{IR}}}(\cdot) \odot A_t^{\Theta^*}, \quad \text{(derivative exposures)}, \tag{3.56}$$

$$\mathrm{E}^{\text{DOS-IR}}\left(t, \cdot \,\middle|\, \Phi_n^{\text{IR}}, \Theta^*\right) \coloneqq \sum_{j=1}^{J} \left(\boldsymbol{h}^{\Phi_n^{\text{IR}}}(\cdot)\right)_j (A_t^{\Theta^*})_j, \quad \text{(portfolio exposure)}, \tag{3.57}$$

$$\mathrm{E}^{\text{DOS-PR}}\left(t, \cdot \,\middle|\, \Phi_n^{\text{PR}}, \Theta^*\right) \coloneqq h^{\Phi_n^{\text{PR}}}(\cdot, A_t^{\Theta^*}), \quad \text{(portfolio exposure)}, \tag{3.58}$$

where Φ_n^{IR}, and Φ_n^{PR} are parameters optimized by minimizing (3.53) and (3.54), respectively, and Θ^* are parameters optimized according to the procedure described in the training procedure, described in Phase I, and $\mathbb{I}_t \in \{0,1\}^J$ represents the exercise history of each derivative in the portfolio. Note that, even though Θ^* does not appear explicitly in the right hand side of (3.55), it is crucial since the cash-flow vector $\boldsymbol{y}$, used in (3.53) and (3.54), is created by applying an exercise strategy controlled by Θ^*. The approximations of EE and PFE are constructed from $M_{\text{train}} \in \mathbb{N}$ independent realizations of X, which for $m \in \{1, 2, \ldots, M\}$ are denoted by $(x_t^{\text{train}}(m))_{t\in[0,T]}$. For $z \in \{\text{IR}, \text{PR}\}$, the approximators are given by

$$\widehat{\mathrm{EE}}^{\text{DOS-}z}(t) \coloneqq \sum_{m=1}^{M} \Pi^{\text{DOS-}z}\left(t, x(m) \,\middle|\, \Phi_n^z, \Theta^*\right), \tag{3.59}$$

$$\widehat{\mathrm{PFE}}_{\alpha}^{\text{DOS-}z}(t) \coloneqq \Pi^{\text{DOS-}z}\left(t, x(i_\alpha) \,\middle|\, \Phi_n^z, \Theta^*\right), \tag{3.60}$$

where i_α is the index of the empirical $\alpha-$percentile of the vector $\Big(\Pi^{\text{DOS-}z}\left(t, x(1) \,\middle|\, \Phi_n^z, \Theta^*\right), \ldots, \Pi^{\text{DOS-}z}\left(t, x(M) \,\middle|\, \Phi_n^z, \Theta^*\right)\Big)$.

3.4 Numerical experiments

In the numerical experiments we use a Geometric Brownian Motion (GBM) to model the asset processes and an intensity model for default events of the counterparty. To be able to

incorporate WWR, the default intensity is linked to the market state of the asset processes. The default event is triggered by an exogenous component, independent of observable market information. On the other hand, the intensity depends on the credit spread of the counterparty (observable from zero-coupon bonds) as well as a WWR-parameter. Our model choices are not necessarily used in practice but they serve the purpose of being easy to analyse. Especially the default model makes it straight-forward to analyze the effects of the credit spread of the counterparty and the WWR-parmeter. It should be pointed out that the algorithms described in this chapter are model independent in the sense that they are fully data-driven. This means that as long as we can sample (or in any other way obtain) market data and default events, we can train the neural networks and the computations below can be performed.

In addition, the algorithms have also been implemented for a portfolio of Bermudan swaptions with dynamics following the one-factor Hull–White model. The results are similar, and are therefore not included in this section.

3.4.1 Risk-factor model

In the Black–Scholes framework, the assets are described by a $\mathbb{N}_+ \ni d-$dimensional Geometric Brownian. For $t \in [0,T]$, with constant risk-free rate $r \in \mathbb{R}$, initial state $s_0, \in (0,\infty)^d$, constant dividend $q \in (0,\infty)^d$ and volatility $\sigma \in (0,\infty)^d$, component $i \in \{1,2,\dots,d\}$ of the asset process $S = (S_t)_{t\in[0,T]}$ is given by

$$(S_t)_i = (s_0)_i \exp\Big(\Big(r - q_i - \frac{\sigma_i^2}{2}\Big)t + \sigma_i (W_t)_i\Big), \tag{3.61}$$

where $W = (W_t)_{t\in[0,T]}$ is a correlated standard Brownian motion, satisfying for $i,j \in \{1,2,\dots,d\}$,
$\mathbb{E}_0[\mathrm{d}(W_t)_i \mathrm{d}(W_t)_j] = \rho_{ij}\mathrm{d}t$, with $\rho_{ij} \in [-1,1]$.

3.4.2 Default model

Following [59], we model a default event of the counterparty as

$$\tau^{\mathrm{D}} = \inf_{t\in[0,T]} \Big\{ t \colon \int_0^t \tilde{h}(u, \tilde{S}_u)\mathrm{d}u \geq E_1 \Big\},$$

where E_1 is a random variable, uniformely distributed on $[0,1]$. Furthermore, the process $(\tilde{S}_t)_{t\in[0,T]}$ is the geometric average of the d component of the dividend-free version of S, given by

$$\tilde{S}_t = \prod_{i=1}^{d} \Big((s_0)_i \exp\Big(\Big(r - \frac{\sigma_i^2}{2}\Big)t + \sigma_i (W_t)_i\Big)\Big)^{1/d}.$$

For simplicity, without loss of generality, from now on, we assume no correlation between the components of the Brownian motions, *i.e.*, $\rho_{ij} = 0$ for $i \neq j$. The above is a one-dimensional GBM, which can be written as

$$\tilde{S}_t = \tilde{s}_0 \exp\Big(\Big(\tilde{\mu} - \frac{\tilde{\sigma}^2}{2}\Big)t + \tilde{\sigma}\tilde{W}_t\Big),$$

where $\tilde{s}_0 = \Big(\prod_{i=1}^d (s_0)_i\Big)^{1/d}$, $\tilde{\sigma} = \frac{1}{d}\Big(\sum_{i=1}^d \sigma_i^2\Big)^{1/2}$, $\tilde{\mu} = r - \frac{1}{2d}(1-\frac{1}{d})\sum_{i=1}^d \sigma_i^2$ and $\tilde{W}_t = \frac{1}{d\tilde{\sigma}}\sum_{i=1}^d \sigma_i (W_t)_i$. For $(t,x) \in [0,T]\times\mathbb{R}_+$, $\tilde{h}$ is of the form $\tilde{h}(t,x) = c(t) + b\log x$. It can be checked that $\tilde{W} = (\tilde{W}_t)_{t\in[0,T]}$, is a 1-dimensional standard Brownian motion. By setting

$$c(t) = \bar{h} + b\log\tilde{S}_0 - (r - \frac{\tilde{\sigma}^2}{2})bt + \frac{1}{2}b^2\tilde{\sigma}^2t^2,$$

we obtain

$$\tilde{h}_t = \tilde{h}(t,\tilde{S}_t) = \bar{h} + \frac{1}{2}\tilde{\sigma}^2t^2b^2 + b\tilde{\sigma}\tilde{W}_t.$$

The economic interpretation of the above is that $\bar{h}$ is the credit spread for the counterparty and b controls the wrong way risk (WWR).

Recall that the jump-to-default process, $\mathbb{1}^{\mathrm{D}}$, is given by $\mathbb{1}_t^{\mathrm{D}} = \mathbb{I}_{\{t<\tau^{\mathrm{D}}\}}$ which gives a survival probability $G_t = \mathbb{E}^{\mathbb{Q}}[\mathbb{1}_t^{\mathrm{D}}|\mathcal{H}_t] = \exp\Big(-\int_0^t \tilde{h}_s \mathrm{d}s\Big)$ (for details, see [59]).

3.4.3 Experiments

Contract details:

We consider a portfolio of $J = 8$ derivatives, depending on an asset process in $d = 2$ dimensions. We set $T = 3$ and use for each derivative, $j \in \{1,2,\ldots,8\}$, the set of exercise dates $\mathbb{T}_j(t) = \mathbb{T}_j = \Big\{0, \frac{1}{3}, \frac{2}{3}, 1, \frac{4}{3}, \frac{5}{3}, 2, \frac{7}{3}, \frac{8}{3}, 3\Big\}$, and the pay–off functions given in Table 3.2.

Contract number	Contract name	Pay-off function
j=1	Max-call option	$(\max\{x_1,x_2\}-100)^+$
j=2	Max-put option	$(100-\max\{x_1,x_2\})^+$
j=3	Geometric-average-call option	$\big(\sqrt{x_1x_2}-100\big)^+$
j=4	Geometric-average-put option	$\big(100-\sqrt{x_1x_2}\big)^+$
j=5	Arithmetic-average-call option	$\Big(\frac{1}{2}(x_1+x_2)-100\Big)^+$
j=6	Arithmetic-average-put option	$\Big(100-\frac{1}{2}(x_1+x_2)\Big)^+$
j=7	1d-call option	$(x_1-100)^+$
j=8	1d-put option	$(100-x_1)^+$

Table 3.2: The two components of the asset process, S, are represented by $x_1, x_2 \in \mathbb{R}$ and $(\,\cdot\,)^+ = \max\{\,\cdot\,, 0\}$.

Dynamics details:

For $i \in \{1,2\}$, we set $(s_0)_i = 100$, $q_i = 0.1$, $r = 0.05$, $\sigma_i = 0.2$, $\rho_{ii} = 1$ and $\rho_{12} = \rho_{21} = 0$.

Neural network details:

We use $M_{\text{train}} = M_{\text{reg}} = M_{\text{reg}} = M = 2^{20}$, and for simplicity, the same structure and hyperparameter-settings are used in all networks, *i.e.,* all the networks in Phase I, and Phase II. We use training batches of size 5000, 3 hidden layers and 30 nodes in each hidden layer. Furthermore, the learning rate decreases step-wise, with equally sized steps after 100 training batches from 10^{-2} to 10^{-6}.

3.4.3.1 Risk-free valuation

The purpose of the risk-free valuation is two-fold. Firstly, since the risky derivative values are used as an input in the risky-valuations, they need to be accurate. To ensure accuracy,

the risk-free values are compared to a well-established existing valuation method, namely the Stochastic Grid Bundling Method (SGBM), see [28] for details. Secondly, it is in its own, an interesting and challenging problem to compute the value of a portfolio of complex derivatives with early-exercise features, without having to do one computation per derivative.

As mentioned above, we compare the algorithms introduced in earlier sections with the SGBM. We emphasize that the values for each derivative needs to be approximated individually when using SGBM, *i.e.,* we perform 8 regressions, one for each derivative. In Table 3.3, we compare the value at $t = 0$ for each derivative approximated with the portfolio version of the DOS-algorithm and the SGBM. In Table 3.3, we compare our values for each derivative at the initial time with the values obtained by SGBM. For the DOS-values, the neural network is trained five times, and evaluated on new, independent, samples and the average values are reported. For the SGBM, the regression is performed five times for each derivative, and the average values of the direct estimators are reported. It should be mentioned that, for $j \in \{3, 4, 5, 6, 7, 8\}$, the SGBM-values are biased high[7] and, for all j, the DOS-values are biased low. In Figure 3.1 to the left we compare EE, $\text{PFE}_{97.5}$ and $\text{PFE}_{2.5}$ approximated

	j=1	**j=2**	**j=3**	**j=4**	**j=5**	**j=6**	**j=7**	**j=8**	$\Pi(0, s_0)$
Ref.	13.902	NA	NA	NA	NA	NA	NA	NA	NA
DOS	13.902	9.520	4.363	16.770	4.919	15.313	7.965	18.021	90.773
SGBM	13.899	9.780	4.366	16.770	4.971	15.327	7.963	18.033	91.108

Table 3.3: Valuation at the level of each derivative with the portfolio version of DOS, and the SGBM. The reference solution is computed by a binomial lattice model in [24].

with SGBM and according to (3.59) and (3.60). To the right, we compare the derivative-wise EE approximated with SGBM and the DOS-IR based algorithm. In the DOS-IR based algorithm, the EE is approximated by evaluating (3.56) at $x(1), \ldots, x(M)$ and computing the component-wise sample mean.

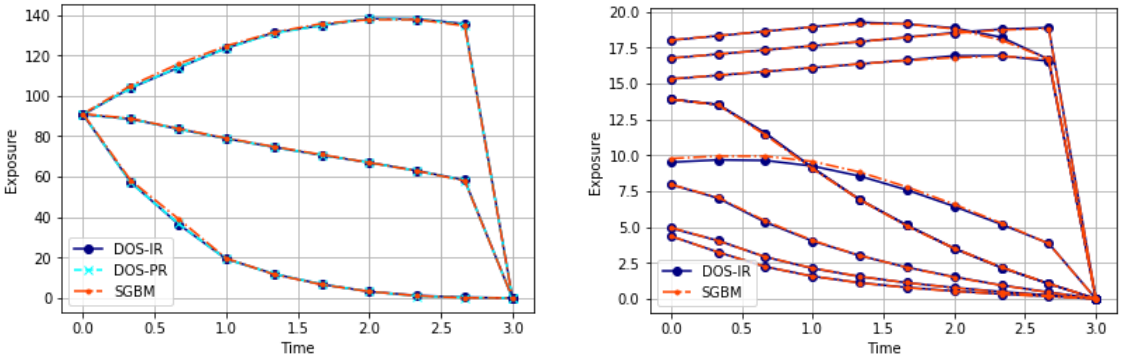

Figure 3.1: Left: EE, $\text{PFE}_{97.5}$ and $\text{PFE}_{2.5}$ at portfolio level computed with DOS-PR, DOS-IR and SGBM respectively. **Right:** EE at derivative level computed with DOS-IR and SGBM, respectively.

We conclude that, although very different in nature, the SGBM and the two methods presented in this chapter, agree on the values at time 0, the tail-distribution of the portfolio exposure over time (at least at the 97.5 and 2.5 percentiles), and the average values of each derivative over time. We emphasize, that we do not claim that one of the methods perform

[7]The reason that the SGBM-values are not biased high for $j \in \{1, 2\}$ can be explained by the non-linearity of the max-operator in the pay-off functions. For more details we refer to [28].

better than the others. For an analysis of the difference in performance between the methods (in the special case $J = 1$), we refer to Chapter2.

3.4.3.2 Risky valuation

In this subsection we focus on the impact of the exercise policy for CVA computations. To be more precise, we investigate to what extent the CVA is overestimated when using the risk-free exercise policy, with and without netting, for different levels of WWR and credit quality of the counterparty. The contracts described in Table 3.2 all have positive pay-offs meaning that the corresponding derivative values are also positive. This eliminates the netting effect, and therefore, we add a derivative to the portfolio, which may take on negative values. The 9:th derivative is a European-type future with pay-off $g_9(t, S_t) = 2 \times (80 - (S_t)_1)\mathbb{I}_{\{t=T\}}$. By the martingale property we have that

$$V_9(t, S_t) = \mathbb{E}_t\big[\mathrm{e}^{-r(T-t)} g_9(T, S_T)\big] = 160\mathrm{e}^{-r(T-t)} - 2(S_t)_1 \mathrm{e}^{-q_1(T-t)}.$$

Furthermore, in the netted portfolio, we include an interest rate-free collateral, $C = 35$. The collateral is such that, at a default, the banks exposure is lowered by 35, and added to the close-out amount. If no default occurs before maturity, then the collateral plays no role.

	$\bar{h}$	$\Upsilon^V[\boldsymbol{f}^V]$	$\Upsilon^U[\boldsymbol{f}^U]$	$\Upsilon^U[\boldsymbol{f}^V]$	$\Upsilon^A[\boldsymbol{f}^A]$	$\Upsilon^A[\boldsymbol{f}^V]$
$b = -0.2$	0	78.62	77.47	77.41	77.86	77.84
	0.1	78.62	59.07	57.22	69.20	67.72
	0.2	78.62	47.58	42.99	64.17	60.82
$b = 0$	0	78.62	78.62	78.62	78.62	78.62
	0.1	78.62	59.50	58.00	69.55	68.39
	0.2	78.62	47.83	43.58	64.37	61.32
$b = 0.2$	0	78.62	78.28	78.27	78.59	78.58
	0.1	78.62	59.65	58.31	69.78	68.75
	0.2	78.62	47.89	43.78	64.51	61.55

Table 3.4: Portfolio valuation with and without netting for different early-exercise policies. We use the short hand notations $\Upsilon^V[\boldsymbol{f}] = \Upsilon^V(0, s_0 \,|\, \boldsymbol{f})$, $\Upsilon^U[\boldsymbol{f}] = \Upsilon^U(0, s_0, 1 \,|\, \boldsymbol{f})$ and $\Upsilon^A[\boldsymbol{f}] = \Upsilon^A(0, s_0, 1, \mathbf{1}_9 \,|\, \boldsymbol{f})$. Furthermore, in this table $\boldsymbol{f}^V$, $\boldsymbol{f}^U$ and $\boldsymbol{f}^A$ represent our neural network approximations of the optimal decision functions.

	$\bar{h}$	CVA	$\overline{\text{CVA}}$	Rel. ov.est.
$b = -0.2$	0	1.15	1.21	5.2%
	0.1	19.55	21.40	9.5%
	0.2	31.04	35.63	14.8%
$b = 0$	0	0	0	0%
	0.1	19.2	20.62	7.4%
	0.2	30.79	35.04	13.8%
$b = 0.2$	0	0.34	0.35	2.9%
	0.1	18.97	20.31	7.1%
	0.2	30.73	34.84	13.4%

Table 3.5: CVA for a portfolio without netting. The relative error in the column to the right represents, in percentage, the overestimation of the CVA by using only the risk-free policy for early-exercise. 'Rel. ov.est.' is short for 'Relative overestimation'.

	$\bar{h}$	CVA^{net}	$\overline{\text{CVA}}^{\text{net}}$	Rel. ov.est.
$b = -0.2$	0	0.78	0.834	6.4%
	0.1	9.42	10.90	15.7%
	0.2	14.45	17.80	23.2%
$b = 0$	0	0	0	0%
	0.1	9.07	10.23	11.4%
	0.2	14.25	17.30	21.4%
$b = 0.2$	0	0.0364	0.0374	5.6%
	0.1	8.83	9.87	11.7%
	0.2	14.11	17.07	21.0%

Table 3.6: CVA for a portfolio with netting. The relative error in the column to the right represents, in percentage, the overestimation of the CVA by using only the risk-free policy for early-exercise. 'Rel. ov.est.' is short for 'Relative overestimation'.

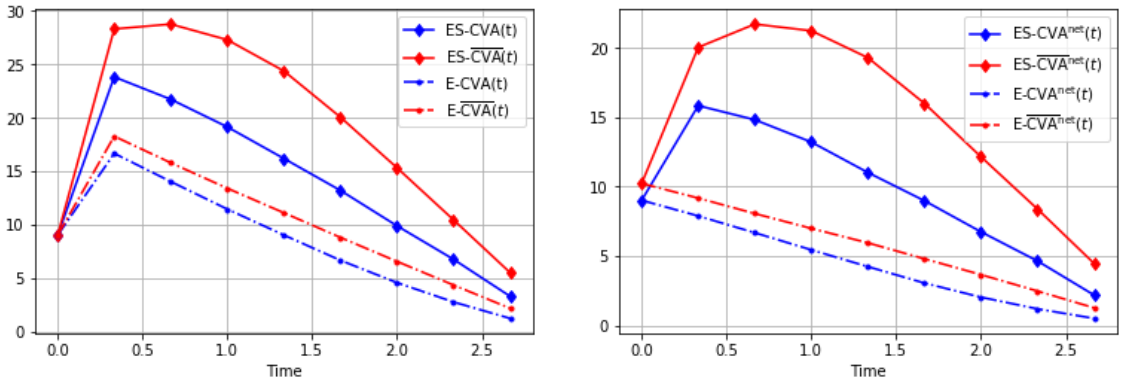

Figure 3.2: Time 0 expectation and expected shortfall at level 97.5% of CVA for the risky-free and the risky exercise policies. **Left:** Without netting. **Right:** With netting.

In Table 3.4, the portfolio values with and without netting are displayed for different exercise policies and for different credit qualities and WWR-parameters. We see that for low values of $\bar{h}$ (credit spread), *i.e.,* for high credit quality of the counterparty, the risk-free exercise policy is a relatively good approximation also for the risky portfolios. However, when the credit quality decreases, the importance of the correct exercise policy increases. This is also reflected in Tables 3.5 and 3.6 in which the corresponding CVA-values are displayed. In practice, a counterparty with a bad credit quality is penalized twice; first by paying a larger CVA (justified), and secondly for paying a CVA which is more overestimated (not justified). The effect is even more significant when we look at expected shortfalls. In Figure 3.2, we see that the ES-CVA, at 97.5% level, is overestimated by between 19% and 67% for portfolio without netting and 27% to 103% for the portfolio with netting. Bare in mind that for this particular choice of parameters, the overestimation of the CVAs are only 7.4% and 11.4%. All values reported in this the section on risky-valuation are results of only one experiment (no Monte-Carlo averages are used). The reason for this is that we are using many samples, and the variance of the results are therefore low. In addition, we have no reference values to compare with and the main point is that we obtain different values depending on what exercise strategy is used, not high accuracy of the estimated entities.

The WWR-parameter b has a relatively small impact. This is not surprising since a positive b gives WWR for call options and Right Way Risk (RWR) for put options and a the other way around with a negative b. Since we have a portfolio consisting of a combination of puts and calls, we have a significant off-setting effect.

3.5 Appendix - Neural network details

In this Appendix, we briefly explain the structure of the neural networks used in this chapter. We present pseudo code for the algorithm used to find the exercise strategy for the risk-free portfolio (the extensions to the risky-portfolios are straight-forward). Furthermore, it is described how the time 0 value for a risk-free portfolio is computed.

3.5.1 Specification of the neural networks used

For completeness, we introduce all the trainable parameters that are contained in each of the parameters $\theta_1, \theta_2, \ldots, \theta_{N-1}$, and present the structure of the networks.

In this section, the following notation is used:

- We denote the dimension of the input layers by $\mathfrak{D}^{\text{input}} \in \mathbb{N}$, and we assume the same input dimension for all $n \in \{1, 2, \ldots, N-1\}$ networks. The input is assumed to be the market state $x_n^{\text{train}} \in \mathbb{R}^d$, and hence $\mathfrak{D}^{\text{input}} = d$. However, we can add additional information to the input that is mathematically redundant but helps the training, *e.g.,* the immediate pay-off, to obtain as input[8] concat $\left(x_n^{\text{train}}(m), \left(g_1\left(T_n, x_n^{\text{train}}(m)\right), \ldots, g_J\left(T_n, x_n^{\text{train}}(m)\right)^T\right)\right)^T \in \mathbb{R}^{d+J}$, which would give $\mathfrak{D}^{\text{input}} = d + J$;
- For network $n \in \{1, 2, \ldots, N-1\}$, we denote the number of layers[9] by $\mathfrak{L}_n \in \mathbb{N}$, and for layer $\ell \in \{1, 2, \ldots, \mathfrak{L}_n\}$, the number of nodes by $\mathfrak{N}_{\ell,n} \in \mathbb{N}$. Note that $\mathfrak{N}_{n,1} = \mathfrak{D}^{\text{input}}$;

[8] concat denotes a concatenation of vectors.
[9] Input and output layers included.

- For network $n \in \{1, 2, \ldots, N\}$, and layer $\ell \in \{2, 3, \ldots, \mathfrak{L}_n\}$ we denote the weight matrix, acting between layers $\ell - 1$ and ℓ, by $w_{n,\ell} \in \mathbb{R}^{\mathfrak{N}_{n,\ell-1} \times \mathfrak{N}_{n,\ell}}$, and the bias vector by $b_{n,\ell} \in \mathbb{R}^{\ell}$;
- For network $n \in \{1, 2, \ldots, N\}$, and layer $\ell \in \{2, 3, \ldots, \mathfrak{L}_n\}$ we denote the (scalar) activation function by $a_{n,\ell} \colon \mathbb{R} \to \mathbb{R}$ and the vector activation function by $\boldsymbol{a}_{n,\ell} \colon \mathbb{R}^{\mathfrak{N}_{n,\ell}} \to \mathbb{R}^{\mathfrak{N}_{n,\ell}}$, which, for $x = (x_1, x_2, \ldots, x_{\mathfrak{N}_{n,\ell}})$, is defined by

$$\boldsymbol{a}_{n,\ell}(x) = \begin{pmatrix} a_{n,\ell}(x_1) \\ \vdots \\ a_{n,\ell}(x_{\mathfrak{N}_{n,\ell}}) \end{pmatrix};$$

- The output of the network should belong to $(0,1)^J \subset \mathbb{R}^J$, meaning that the dimension of the output, denoted by $\mathfrak{D}^{\text{output}}$ should equal J. To enforce the output to only take on values in $(0,1)$, we restrict ourselves to scalar-activation functions of the form $a_{n,\mathfrak{L}_n} \colon \mathbb{R} \to (0,1)$.

Network $n \in \{1, 2, \ldots N-1\}$ is then defined by

$$\boldsymbol{F}^{\theta_n}(\cdot) = L_{n,\mathfrak{L}_n} \circ L_{n,\mathfrak{L}_n - 1} \circ \cdots \circ L_{n,1}(\cdot), \tag{3.62}$$

where, for $n \in \{1, 2, \ldots, N-1\}$ and for $x \in \mathbb{R}^{\mathfrak{L}_{n,\ell-1}}$, the layers are defined as

$$L_{n,\ell}(x) = \begin{cases} x, & \text{for } \ell = 1, \\ \boldsymbol{a}_{n,\ell}(w_{n,\ell}^T x + b_{n,\ell}), & \text{for } \ell \geq 2, \end{cases}$$

with $w_{n,\ell}^T$ the matrix transpose of $w_{n,\ell}$. The trainable parameters of network $n \in \{1, 2, \ldots, N-1\}$ are then given by the list

$$\theta_n = \{w_{n,2}, b_{n,2}, w_{n,3}, b_{n,3}, \ldots, w_{n,\mathfrak{L}_n}, b_{n,\mathfrak{L}_n}\},$$

and as already stated, $\Theta_n = \{\theta_n, \theta_{n+1}, \ldots, \theta_{N-1}\}$ and $\Theta = \Theta_1$.

3.5.2 Training and valuation

The main idea of the training and valuation procedure is to fit the parameters to some training data, and then use the fitted parameters to make informed decisions with respect to some unseen, valuation data independent of the training data. The training and valuation is described for the risk-free problem, but the procedure is similar for the risky portfolios.

Training:
Sample $M_{\text{train}} \in \mathbb{N}$ independent realizations of X, which for $m \in \{1, 2, \ldots, M_{\text{train}}\}$ are denoted by $x^{\text{train}}(m)$. In practice, we often need to approximate X, *e.g.,* if X satisfies a stochastic differential equation (SDE) we may have to use a temporal discretization scheme. Since this is not the focus of this chapter, we assume that X can be approximated pathwise arbitrarily well.

At maturity of the portfolio, we define the cash-flow corresponding to the j:th contract and the m:th realization of X as $\mathrm{CF}_{N,j}(m) = D_{N-1,N} g_j(T_N, x_N^{\text{train}}(m))$ (recall that $g_j(t_N, \cdot) \equiv 0$ if T_N is larger than the date of maturity for contract j).
For $n = N-1, N-2, \ldots, 1$, do the following:

1. Approximate the optimal parameter $\theta_n^{**} \in \mathbb{R}^{q_n}$, given by

$$\theta_n^{**} \in \underset{\theta \in \mathbb{R}^{q_n}}{\arg\max} \Big(\frac{1}{M_{\text{train}}} \sum_{m=1}^{M_{\text{train}}} \sum_{j=1}^{J} F_j^{\theta}(x_n^{\text{train}}(m))\, g_j(T_n, x_n^{\text{train}}(m)) + (1 - F_j^{\theta}(x_n^{\text{train}}(m)))\mathrm{CF}_{n+1,j}(m) \Big),$$

   with $\boldsymbol{F}_n^{\theta}$ as in (3.62). The approximation of θ_n^{**} is denoted by θ_n^*. The above corresponds to an (empirical) loss-function of the form

$$L(\theta; x^{\text{train}}) = -\frac{1}{M_{\text{train}}} \sum_{m=1}^{M_{\text{train}}} \sum_{j=1}^{J} F_j^{\theta}(x_n^{\text{train}}(m))\, g_j\left(T_n, x_n^{\text{train}}(m)\right) + (1 - F_j^{\theta}(x_n^{\text{train}}(m)))\mathrm{CF}_{n+1,j}(m).$$

   In the above we distinguish between the theoretically optimal parameter, θ_n^{**} (which we rarely find), and the parameter obtained via an optimization algorithm θ_n^*. The minus sign in the loss-function transforms the problem from a maximization to a minimization problem, which is the standard formulation in the machine learning community. Note the straight forward relationship between the loss function and the average cash-flows in (3.43). In practice, the data is often divided into mini-batches, for which the loss-function is minimized consecutively.

2. For $m = 1, 2, \ldots, M_{\text{train}}$, and for $j = 1, 2 \ldots, J$, update the discounted cash-flows according to:

$$\begin{aligned}&\mathrm{CF}_{n,j}(m)\\ &= f_j^{\theta_n^*}(x_n^{\text{train}}(m))\, g(T_n, x_n^{\text{train}}(m)) + D_{n,n+1}(1 - f_j^{\theta_n^*}(x_n^{\text{train}}(m)))\mathrm{CF}_{n+1,j}(m).\end{aligned}$$

Note that we use the continuous versions, $F_j^{\theta_n^*}$, in the optimization phase, and the discontinuous versions $f_j^{\theta_n^*}$ when we update the cash-flows. The performance of the algorithm does not seem to be sensitive to the specific choice of the number of hidden layers, number of nodes, optimization algorithm, etc. Below is a list of the most relevant parameters/structural choices:

- Initialization of the trainable parameters, where a typical procedure is to initialize the biases to 0, and sample the weights independently from a normal distribution;
- The activation functions $a_{\ell,n}$, which are used to add a non-linear structure to the neural networks. In our case we have the strict requirement that the activation function of the output layer maps $\mathbb{R}$ to $(0, 1)$. This could, however, be relaxed as long as the activation function is both upper and lower bounded, since we can always scale and shift such output to take on values only in $(0, 1)$. For a discussion on different activation functions, see *e.g.,* [41];

- The batch size, $\mathcal{B}_n \in \{1, 2, \ldots, M_{\text{train}}\}$, is the number of training samples used for each update of θ_n, *i.e.,* with $\mathcal{B}_n = M_{\text{train}}$, the loss function is of the form defined in step 1 above. If we want all batches to be of equal size, we need to choose $\mathcal{B}_n$ to be a multiplier of M_{train};

- For each update of θ_n, we use an optimization algorithm, for which a common choice is the Adam optimizer, proposed in [42]. Depending on the choice of optimization algorithm, there are different parameters related to the specific algorithm to be chosen. One example is the so-called learning rate which decides how much the parameter, θ_n, is adjusted after each batch.

Once the parameters have been optimized we define $\Theta^* := \{\theta_1^*, \theta_2^*, \ldots, \theta_{N-1}^*\}$, which contains all the information needed in order to use the algorithm for valuation.

Remark 3.5.1. *In the training for the risky portfolio with netting, the algorithm needs to be adjusted since it depends on the exercise history. This cannot easily be included since the algorithm is carried out backwards in time recursively and therefore, at exercise date T_n, we do not yet know which derivatives have been exercised prior to T_n. This is resolved by, at each exercise date T_n, randomly assigning a state for $\alpha_n(m) \in \{0,1\}^J$, representing A_n, for each sample m. Since the future cash-flows depend on $\alpha_n(m)$, they need to be re-iterated for each m. This is done by iteratively, evaluating the already approximated decision functions at X for each exercise date greater than T_n, until all the derivatives are exercised, or a default of the counterparty occurs.*

Valuation:
Sample $M_{\text{test}} \in \mathbb{N}$ independent realizations of X, denoted $\left(x_t^{\text{val}}(m)\right)_{t\in[0,T]}$. Denote the vector of decision functions by

$$\mathbb{f}_n^{\Theta^*} = \left(\boldsymbol{f}^{\theta_n^*}, \boldsymbol{f}^{\theta_{n+1}^*}, \ldots, \boldsymbol{f}^{\theta_{N-1}^*}\right),$$

and $\mathbb{f}^{\Theta^*} := \mathbb{f}_1^{\Theta^*}$. We then obtain for sample m, *i.e.,* $x^{\text{val}}(m)$, the following stopping rule

$$\bar{\tau}_{n,j}^{\Theta^*} = \left(\boldsymbol{\tau}\left[\mathbb{f}_n^{\Theta^*}\right]\left(x^{\text{val}}(m)\right)\right)_j = \sum_{k=n}^{N} T_k\, f_j^{\theta_k^*}(x_k^{\text{val}}(m)) \prod_{\ell=1}^{k-1}\left(1 - f_j^{\theta_\ell^*}(x_\ell^{\text{val}}(m))\right).$$

The estimated portfolio value at $t = 0$ is then given by

$$\Pi^V(0, x_0) \approx \Upsilon^V(0, x_0 \,|\, \mathbb{f}^{\Theta^*}) \approx \frac{1}{M_{\text{val}}} \sum_{m=1}^{M_{\text{val}}} \sum_{j=1}^{J} D_{0,\bar{\tau}_{0,j}^{\Theta^*}}\, g_j\left(\bar{\tau}_{0,j}^{\Theta^*}, x_{\bar{\tau}_{0,j}^{\Theta^*}}^{\text{val}}(m)\right). \tag{3.63}$$

By construction, any stopping strategy is sub-optimal, implying that the estimate (3.63) is biased low. However, it should be pointed out that it is possible to derive a biased-high estimation of $\Pi^V(0, x_0)$ from a dual formulation of the optimal stopping problem, which is described in [9]. In addition, numerical results in [9] show a tight interval for the biased low and biased high estimates for a wide range of problems.

4

Convergence of a robust deep FBSDE method for stochastic control

In this chapter, we propose a deep learning-based numerical scheme for strongly coupled FBSDEs, stemming from stochastic control. It is a modification of the deep BSDE method, in which the initial value to the backward equation is not a free parameter, with a new loss function being the weighted sum of the cost of the control problem and a variance term which coincides with the mean squared error in the terminal condition. We show by a numerical example that a direct extension of the classical deep BSDE method to FBSDEs, fails for a simple linear-quadratic control problem, and motivate why the new method works. Under regularity and boundedness assumptions on the exact controls of time continuous and time discrete control problems, we provide an error analysis for our method. We show empirically that the method converges for three different problems, including the one that failed for a direct extension of the deep BSDE method.

Keywords *- Stochastic control, FBSDE, neural networks, numerical analysis*

This chapter is based on the paper with the same title, which is published in SIAM Journal on Scientific Computing 45.1 (2023): A226-A255.

4.1 Introduction

Forward backward stochastic differential equations (FBSDEs) constitute an important family of models with many applications in a wide variety of fields such as finance, physics, chemistry and engineering. As the name suggests, an FBSDE consists of two stochastic differential equations (SDE), one forward SDE and one backward SDE, commonly referred to as the forward equation and the backward equation, respectively. The forward equation is a classical SDE with a given initial value, while the backward equation has a given stochastic terminal value and the initial value is part of the solution. In this chapter, we are concerned with stochastic control problems, which typically lead to coupled FBSDEs, meaning that the primary stochastic variables in the backward SDE impact the forward SDE and vice versa. Since closed-form solutions to FBSDEs are rare, one often has to rely on numerical approximations.

In this chapter, we propose a method which falls into the rapidly growing category of neural network-based approximation schemes for FBSDEs and PDEs. Even though there are a number of works in this direction, the methods stem from the pioneering work [4], and are similar in spirit. In the present chapter, non-convergence of the method from [4], originally proposed for non-coupled FBSDEs, is identified, when applied to strongly coupled FBSDEs. We present a new family of methods and prove analytically and numerically that it overcomes the convergence problem.

Prior to the recent surge in machine learning-based algorithms, the task of approximating FBSDEs has been an active field of research for several decades. From a general perspective, approximation schemes can be categorized into *backward* and *forward* numerical methods, referring to the order in time in which the methods operate.

A backward numerical method usually relies on an initialization of the backward equation at the known terminal value (or an approximation of the terminal value). The solution is then approximated recursively, backwards in time, by approximating conditional expectations. There are several different methods to approximate these conditional expectations such as *e.g.,* tree-based methods, see [60, 61], Fourier-based methods, see *e.g.,* [62, 63, 64] and least-squares Monte Carlo (LSMC) methods, see *e.g.,* [65, 66, 67, 68, 69, 70]. A general property of backward methods is that the terminal condition of the backward equation depends on the realization of the forward equation. This is not a problem for *decoupled* FBSDEs, but for coupled FBSDEs the method becomes implicit and iterative schemes which may not always converge need to be employed. A second class of algorithms, which is more suitable for coupled FBSDEs is the class of forward methods such as PDE-based methods see *e.g.,* [71, 72] and Picard linearization schemes, see *e.g.,* [73, 74, 75]. For a summary on forward and backward numerical methods for FBSDEs, we refer to [76]. A typical drawback for the methods mentioned above is that they suffer from the curse of dimensionality (some methods such as LSMC and Picard schemes may overcome this problem to some extent), meaning that the time complexity and the memory requirements increase exponentially with the dimensions of the problem.

In addition to the classical methods described above, a new branch of algorithms based on neural networks has appeared in recent years. In [4], the authors present a method called the *Deep BSDE method*, which relies on a neural network parametrization of the control process and the initial condition of the backward equation. Both the forward and the backward equations

are then treated as forward equations and are approximated with the Euler–Maruyama scheme. To achieve an accurate approximation, the parameters of the neural networks are optimized such that the terminal condition on the backward equation is (approximately) satisfied in mean squared sense. The method has proven to be able to approximate a wide class of equations in very high dimensions (at least 100). Since the original Deep BSDE method publication, several papers with adjustments of the algorithm as in *e.g.,* [77, 79, 80, 81, 82, 83, 84, 85, 86, 78] and others with convergence analysis in *e.g.,* [87, 88, 89, 90, 91, 92, 93], have been published. In addition to being forward methods, these algorithms are *global* in their approximation, meaning that the optimization of all involved neural networks is carried out simultaneously. This implies that they are optimized subject to one single objective function, also called *loss function*, as it is usually referred to in the machine learning literature. There also exists a branch of neural network-based algorithms relying on *local* optimization techniques. Typically, these methods are of backward type and of similar nature as the LSMC algorithms, but instead of polynomials as their basis functions, neural networks are used. As for the LSMC method, these kinds of algorithms are not easily applied to coupled FBSDEs. Algorithms of this type can be found in *e.g.,* [94, 95, 96] and with error analysis [97, 98]. For an overview of machine learning algorithms for approximation of PDEs, we refer to [99].

As mentioned, we are interested in coupled FBSDEs and the focus is therefore on global algorithms operating forward in time, with a structure similar to the Deep BSDE method. We demonstrate that the approach taken in *e.g.,* [100, 101, 102, 83, 103], where the deep BSDE method is applied to the FBSDEs associated with the stochastic control problem, is problematic. As we show in the present chapter, even though an accurate approximation of the control problem can be achieved, this does not imply an approximation of the FBSDE in general. Moreover, it is observed that the deep BSDE method does not converge for certain problems and the convergence problem is prominent for the strongly coupled FBSDEs. Our proposed method overcomes this problem by employing the equivalence between the stochastic control problem and the FBSDE. To be more precise, we use the fact that the initial value of the BSDE coincides with the value function of the control problem and hence, can be expressed as a minimization problem. Moreover, we use the adaptivity property of the BSDE to conclude that a stochastic version of the value function is $\mathcal{F}_0-$measurable and therefore has zero variance. These two properties are then combined in the loss function to achieve a robust approximation scheme for coupled FBSDEs. The effectiveness of our algorithm is demonstrated empirically on a collection of problems with different characteristics. In addition, a theoretical error analysis is carried out, in which we provide convergence rates for the initial and terminal conditions of the FBSDE under a mild assumption and strong convergence of the FBSDE under stronger assumptions. Our main result is similar to the aposteriori error bound for the deep BSDE method which was established for weakly coupled FBSDEs in [87] and later extended to non-Lipschitz coefficients (but for less general diffusion coefficients) in [93]. However, these results are unlikely to be valid for strongly coupled FBSDEs and we find several examples in which the discrete terminal condition converges while the FBSDE approximation does not. Moreover, the authors in [104] provide posteriori error bounds on the time discretization error for McKean–Vlasov FBSDEs. Under some conditions, this analysis is applicable also for strongly coupled FBSDEs, for instance, when T is small. Finally, in [105,

106, 107, 108], algorithms and convergence results for gradient methods for stochastic control problems of McKean–Vlasov type are presented.

This chapter is structured as follows: In Section 4.2, we present the stochastic control problem and explain the reformulations to a PDE as well as a FBSDE. Moreover, we introduce reformulations of the FBSDE to different variational problems which are used for the algorithms in later sections. The section concludes by introducing the time discrete counterparts of the reformulations as well as a discussion on when and why the deep BSDE method fails to converge. In Section 4.3, the fully implementable algorithms are presented together with details on the neural networks used. Section 4.4 is devoted to error analysis of the proposed algorithm. Classical Euler–Maruyama type discretization errors and errors stemming from differences between time discrete and time continuous stochastic control are discussed. Finally, in Section 4.5 numerical approximations are compared with their analytic counterparts when we have such available.

4.2 The deep FBSDE method and an improved family of methods

This section contains a formal introduction to our proposed method with motivation from stochastic control and FBSDE theory. Section 4.2.1 introduces the stochastic control problem, the related Hamilton-Jacobi-Bellman equation and FBSDE. Alternative formulations of the FBSDE are presented in Section 4.2.2. In Section 4.2.3, we motivate by numerical examples why a direct extension of the deep BSDE method to FBSDEs, as in [100, 101, 102], fails for many problems. Finally, in Section 4.2.4, the proposed robust deep FBSDE method is described. In this section, we present formally the method for the sake of clarity while in Section 4.4, a more rigorous approach is taken.

4.2.1 Stochastic control and FBSDEs

Our starting point is a controlled SDE and its associated cost functional

$$\begin{cases} X_t^u = x_0 + \int_0^t \bar{b}(s, X_s^u, u_s)\mathrm{d}s + \int_0^t \sigma(s, X_s^u)\mathrm{d}W_s, & (4.1) \\ J^u(t,x) = \mathbb{E}\Big[\int_t^T \bar{f}(s, X_s^u, u_s)\mathrm{d}s + g(X_T^u)\,\Big|\, X_t^u = x\Big], \quad t \in [0,T]. & (4.2) \end{cases}$$

Here $T \in (0,\infty)$, $d, k, \ell \in \mathbb{N}$, $(W_t)_{t\in[0,T]}$ is a k-dimensional standard Brownian motion, the coefficients $\bar{b}\colon [0,T]\times\mathbb{R}^d\times\mathbb{R}^\ell \to \mathbb{R}^d$, $\sigma\colon [0,T]\times\mathbb{R}^d\times\mathbb{R}^\ell \to \mathbb{R}^{d\times k}$, $\bar{f}\colon [0,T]\times\mathbb{R}^d\times\mathbb{R}^\ell \to \mathbb{R}$ and $g\colon \mathbb{R}^d \to \mathbb{R}$ satisfy some extra regularity conditions, and the control process, $u = (u_t)_{t\in[0,T]}$, belongs to a set $\mathcal{U}$ of admissible controls, taking values in a set $U \subset \mathbb{R}^\ell$. The aim is to find a control process, $u^* \in \mathcal{U}$, that minimizes $J^u(0,x_0)$.

Assuming the cost to be bounded from below, the *value function* of the control problem is given by

$$V(t,x) = \inf_{u\in\mathcal{U}} J^u(t,x). \tag{4.3}$$

For the presentation, we assume uniqueness of the infimum. Under appropriate conditions, the value function satisfies a *Hamilton–Jacobi–Bellman* (HJB) equation, which is a non-linear

parabolic PDE given by

$$\begin{cases} \dfrac{\partial V}{\partial t}(t,x) + \dfrac{1}{2}\mathrm{Tr}(\sigma\sigma^\top \mathrm{D}_x^2 V)(t,x) + \mathcal{H}(t,x,\mathrm{D}_x V(t,x)) = 0, & (t,x) \in [0,T) \times \mathbb{R}^d, \\ V(t,x) = g(x), & (t,x) \in \{T\} \times \mathbb{R}^d. \end{cases} \tag{4.4}$$

Here Tr denotes the trace of a matrix and for $(t,x) \in [0,T] \times \mathbb{R}^d$, $p \in \mathbb{R}^d$ the *Hamiltonian*, $\mathcal{H}$, is given by

$$\mathcal{H}(t,x,p) = \inf_{v \in U} [\bar{b}(t,x,v)^\top p + \bar{f}(t,x,v)]. \tag{4.5}$$

Under conditions that guarantee a sufficiently regular solution to (4.4), and the infimum in the Hamiltonian to be attained at $v^* = v^*(t,x,p)$, the optimal control is of the feedback form $u^*(t,X_t) = v^*\big(t,X_t,\mathrm{D}_x V(t,X_t)\big)$, where we have written $X \coloneqq X^{u^*}$ for the optimally controlled X^u. The feedback map v^* is for many interesting problems easy to derive. Again, under sufficient regularity, Itô's formula applied to $V(t,X_t)$ yields

$$\begin{cases} X_t = x_0 + \displaystyle\int_0^t b(s,X_s,Z_s)\mathrm{d}s + \int_0^t \sigma(s,X_s)\mathrm{d}W_s, \\ Y_t = g(X_T) + \displaystyle\int_t^T f(s,X_s,Z_s)\mathrm{d}s - \int_t^T \langle Z_s, \mathrm{d}W_s\rangle, \quad t \in [0,T], \end{cases} \tag{4.6}$$

where $Y_t = V(t,X_t)$, $Z_t = \sigma^T(t,X_t)\mathrm{D}_x V(t,X_t)$ and, for $\theta \in \{b,f\}$, we have

$$\theta(t,X_t,Z_t) \coloneqq \bar{\theta}(t,X_t,v^*(t,X_t,(\sigma(t,X_t)\sigma^T(t,X_t))^{-1}\sigma(t,X_t)Z_t)).$$

In the rest of this section, we assume the existence of a unique solution (X,Y,Z) of (4.6) in appropriate spaces. Given Z, or equivalently $\mathrm{D}_x V$, we thus have an optimal control $u_t^* = v^*(t,X_t,(\sigma(t,X_t)\sigma^T(t,X_t))^{-1}\sigma(t,X_t)Z_t)$. This would make efficient numerical FBSDEs schemes very useful for solving the control problem. In the other direction, if we have an optimal control u^*, then in general this does not give us Z, unless $p \mapsto v^*(t,x,p)$ is invertible, and only in this case the control problem naturally suggests numerical schemes for FBSDEs. Below, we introduce a family of numerical schemes for FBSDEs that works regardless of invertibility of the feedback map, but reduces to the control problem in the case of invertibility.

4.2.2 Alternative formulations of FBSDEs

The Deep BSDE method proposed in [4], relies on a reformulation of the FBSDE (4.6) into two forward SDEs, one with apriori unknown initial value. It relies moreover on the Markov property of the FBSDE, which guarantees that $Z_t = \zeta^*(t,X_t)$, for some function $\zeta^*\colon [0,T] \times \mathbb{R}^d \to \mathbb{R}^k$, that we refer to as Markov map, and optimization is done with respect to such functions and initial values y_0. More precisely, the FBSDE (4.6) is reformulated into the following variational

problem

$$\begin{cases} \underset{y_0,\zeta}{\text{minimize}}\ \mathbb{E}|Y_T^{y_0,\zeta} - g(X_T^{y_0,\zeta})|^2, \quad \text{where} \\ X_t^{y_0,\zeta} = x_0 + \int_0^t b(s, X_s^{y_0,\zeta}, Z_s^{y_0,\zeta})\mathrm{d}s + \int_0^t \sigma(s, X_s^{y_0,\zeta})\mathrm{d}W_s, \\ Y_t^{y_0,\zeta} = y_0 - \int_0^t f(s, X_s^{y_0,\zeta}, Z_s^{y_0,\zeta})\mathrm{d}s + \int_0^t \langle Z_s^{y_0,\zeta}, \mathrm{d}W_s\rangle, \\ Z_t^{y_0,\zeta} = \zeta(t, X_t^{y_0,\zeta}), \quad t \in [0,T], \end{cases} \tag{4.7}$$

where y_0 and ζ are sought in appropriate spaces. From the theory outlined in Section 4.2.1, under well-posedness and sufficient regularity of (4.6), it holds that $Y_0 = V(0,x_0)$ and $\zeta^* = \sigma^\top \mathrm{D}_x V$, and thus we have well-posedness of (4.7). While it seems natural to propose a numerical algorithm based on a discrete version of (4.7), we demonstrate below that such an optimization problem, even for many simple problems, does not converge.

In order to introduce numerical schemes that do not suffer under the above problem, we use the following two properties of the initial value Y_0 of (4.6):

(i) Y_0 coincides with the value function of the control problem (property from the control problem);

(ii) Y_0 is $\mathcal{F}_0-$measurable and therefore has zero variance (property from the FBSDE).

The two properties are both captured in the following variational problem:

$$\begin{cases} \underset{\zeta}{\text{minimize}}\ \Phi_\lambda(\zeta) = \mathbb{E}[\mathcal{Y}_0^\zeta] + \lambda \mathrm{Var}[\mathcal{Y}_0^\zeta], \quad \text{where} \\ \mathcal{Y}_0^\zeta = g(X_T^\zeta) + \int_0^T f(t, X_t^\zeta, Z_t^\zeta)\mathrm{d}t - \int_0^T \langle Z_t^\zeta, \mathrm{d}W_t\rangle, \\ X_t^\zeta = x_0 + \int_0^t b(s, X_s^\zeta, Z_s^\zeta)\mathrm{d}s + \int_0^t \sigma(s, X_s^\zeta)\mathrm{d}W_s, \\ Y_t^\zeta = \mathbb{E}[\mathcal{Y}_0^\zeta] - \int_0^t f(s, X_s^\zeta, Z_s^\zeta)\mathrm{d}s + \int_0^t \langle Z_s^\zeta, \mathrm{d}W_s\rangle, \\ Z_t^\zeta = \zeta(t, X_t^\zeta), \quad t \in [0,T]. \end{cases} \tag{4.8}$$

We refer to $\mathcal{Y}_0^\zeta$ as the *stochastic cost* and notice that $\mathbb{E}[\mathcal{Y}_0^\zeta] = J^{u(\zeta)}(0,x_0)$, where $u(\zeta) \in \mathcal{U}$ is the control generated by ζ. Thus, the first term of the objective function Φ_λ is the cost function of the control problem. In the case of $p \mapsto v^*(t,x,p)$ being invertible, this term alone, i.e., for $\lambda = 0$, offers an equivalent formulation to (4.7). In other cases, uniqueness of minimizers of $\zeta \mapsto \mathbb{E}[\mathcal{Y}_0^\zeta]$ cannot be guaranteed, but among the minimizers, there is only one ζ^* with the property that the variance of the stochastic cost $\mathcal{Y}_0^\zeta$ equals zero. The second term of Φ_λ is introduced to penalize non-zero variance and the minimizer for $\lambda > 0$ is unique. Another important feature of the formulation (4.8), is that Y_0^ζ is determined by ζ alone and (4.8) has thus one degree of freedom less than (4.7). A final observation is that

$$\begin{aligned} \mathrm{Var}[\mathcal{Y}_0^\zeta] = \mathbb{E}[|\mathbb{E}[\mathcal{Y}_0^\zeta] - \mathcal{Y}_0^\zeta|^2] &= \mathbb{E}\left[\left|Y_0^\zeta - \int_0^t f(s, X_s^\zeta, Z_s^\zeta)\mathrm{d}s + \int_0^t Z_s^\zeta \mathrm{d}W_s - g(X_T^\zeta)\right|^2\right] \\ &= \mathbb{E}[|Y_T^\zeta - g(X_T^\zeta)|^2]. \end{aligned} \tag{4.9}$$

This implies that the second term of Φ is, up to the factor λ, the same as that of (4.7), but with Y_0^ζ not being a variable to optimize. Thus, there are strong similarities between (4.7) and (4.8), but in the time discrete setting the latter formulation is shown to be advantageous sections below.

Remark 4.2.1. *This remark aims to highlight the meaning of the two terms in the objective function of* (4.8)*. In the special case $\lambda = 0$, we have*

$$\Phi_0(\zeta) = \mathbb{E}\Big[g(X_T^\zeta) + \int_0^T f(t, X_t^\zeta, Z_t^\zeta)\mathrm{d}t - \int_0^T \langle Z_t^\zeta, \mathrm{d}W_t\rangle\Big] = \mathbb{E}\Big[g(X_T^\zeta) + \int_0^T f(t, X_t^\zeta, Z_t^\zeta)\mathrm{d}t\Big].$$

The above coincides with the cost functional from the stochastic control problem in (4.2) *(with f, instead of $\bar{f}$). Therefore, by finding the minimizer of $\Phi_0(\zeta)$, denoted by ζ^*, we have both a minimizer of* (4.2)*, as well as access to the optimal control process u^*. We here assume that it is possible to express each $u \in \mathcal{U}$, in terms of ζ through some mapping $\zeta \mapsto u$ (this condition is usually satisfied by construction of the BSDE).*

In summary, solving (4.8) *with $\lambda = 0$, yields solutions to the forward SDE and the initial condition of the BSDE in* (4.6) *i.e., X and Y_0 respectively. In addition, we obtain the optimal control process, u^* of* (4.2)*. Unfortunately, we do not have access to the full solution of* (4.6)*, since we do not, in general, have access to $(Y)_{t\in(0,T]}$ or Z. The exception is, as stated above, when there is a one-to-one mapping between u_t and Z_t, which in general does not hold. In this situation, it is crucial to use $\lambda > 0$, since, even though there are infinitely many ζ which minimize $\Phi_0(\zeta)$, there is only one ζ^* which makes $\mathcal{Y}_0^{\zeta^*}$ deterministic, and hence $Var[\mathcal{Y}_0^{\zeta^*}] = 0$ and $\Phi_\lambda(\zeta^*) = Y_0$.*

4.2.3 A direct extension of the deep BSDE method and why it fails

In this section, we present the time discrete counterpart of (4.7). We assume an equidistant time grid, $0 = t_0 < t_1 < \ldots < t_N = T$, with $h = t_{n+1} - t_n$ and denote the Brownian increment $\Delta W_n = W_{n+1} - W_n$. Throughout the chapter, we parameterize discretizations by $h \in (0, 1)$ and by this we mean all $h \in (0, 1) \cap \{T/N : N \geq 1\}$.

The time discrete version of (4.7) is given by

$$\begin{cases} \underset{y_0,\zeta}{\text{minimize}} \;\; \mathbb{E}\Big[\big|Y_N^{h,y_0,\zeta} - g(X_N^{h,y_0,\zeta})\big|^2\Big], \quad \text{where} \\ X_n^{h,y_0,\zeta} = x_0 + \sum_{k=0}^{n-1} b(t_k, X_k^{h,y_0,\zeta}, Z_k^{h,y_0,\zeta})h + \sum_{k=0}^{n-1} \sigma(t_k, X_k^{h,y_0,\zeta})\Delta W_k, \\ Y_n^{h,y_0,\zeta} = y_0 - \sum_{k=0}^{n-1} f(t_k, X_k^{h,y_0,\zeta}, Z_k^{h,y_0,\zeta})h + \sum_{k=0}^{n-1} \langle Z_k^{h,y_0,\zeta}, \Delta W_k\rangle, \\ Z_k^{h,y_0,\zeta} = \zeta_k(X_k^{h,y_0,\zeta}). \end{cases} \tag{4.10}$$

It is a direct extension of the deep BSDE method from [4]. In the literature, it was first applied experimentally to FBSDEs in the master thesis [100] and thereafter in [101], both for inverted pendulums, in [102] for an application to attitude control of unmanned aerial vehicles. More examples of implementations of the deep FBSDE method are found in [83, 103].

In [87], the authors consider FBSDEs with coefficients b, σ and f that may take the Y-component, but not the Z-component, as arguments. Under the relatively strict assumption of weak coupling (also called monotonicity condition, see *e.g.,* [109]), it is shown that for h small enough there is a constant C, independent of h, such that

$$\begin{aligned}\sup_{t\in[0,T]}\Big(\mathbb{E}\Big[\big|X_t-\hat{X}_t^{h,y_0,\zeta}\big|^2\Big]+\mathbb{E}\Big[\big|Y_t-\hat{Y}_t^{h,y_0,\zeta}\big|^2\Big]\Big)+\int_0^T\mathbb{E}\Big[\big|Z_t-\hat{Z}_t^{h,y_0,\zeta}\big|^2\Big]\mathrm{d}t\\ \leq C\Big(h+\mathbb{E}\Big[\big|Y_N^{h,y_0,\zeta}-g(X_N^{h,y_0,\zeta})\big|^2\Big]\Big),\end{aligned}\tag{4.11}$$

where for $t\in[t_k,t_{k+1})$, $\hat{X}_t^{h,y_0,\zeta}\coloneqq X_k^{h,y_0,\zeta}$, $\hat{Y}_t^{h,y_0,\zeta}\coloneqq Y_k^{h,y_0,\zeta}$ and $\hat{Z}_t^{h,y_0,\zeta}=\zeta_k(X_k^{h,y_0,\zeta})$. Under some additional assumptions on the coefficients b, f, σ and g (additional smoothness and boundedness of the coefficients to guarantee a bounded and smooth solution of the associated HJB equation), the results in [87] can be extended to the framework of interest in this chapter, *i.e.,* coefficients taking the Z-component as an argument. On the other hand, the weak coupling condition is rarely satisfied for FBSDEs stemming from stochastic control problems, and to the best of our knowledge, there is no known way to relax this condition.

To investigate convergence of (4.10) empirically, we first note that if we would know Y_0 apriori, then the variational problem (4.7) would be reduced to finding ζ. We also know that $\mathrm{D}_x V^\top\sigma$ is the minimizer, which would make the objective function identical to zero. In the discrete counterpart, we would expect that, if (4.10) converges to (4.7), then for sufficiently small h, the objective function would be close to zero if optimizing only ζ and setting $y_0=Y_0$. Moreover, for a robust algorithm to emerge from (4.10), it is important that $y_0\neq Y_0$ results in a larger value of the optimal objective function, at least when y_0 and the true initial value, Y_0, are "far away" from each other. To formalize this, we introduce the mean squared error

$$\mathrm{MSE}(y_0)\coloneqq\inf_{\zeta}\mathbb{E}\big[\big|Y_N^{h,y_0,\zeta}-g(X_N^{h,y_0,\zeta})\big|^2\big].\tag{4.12}$$

The aim is to investigate whether or not MSE is minimized at, or close to, the true initial condition Y_0. Moreover, for each y_0, we want to investigate the Markov map ζ^{y_0} that minimizes $\zeta\mapsto\mathbb{E}[|Y_N^{h,y_0,\zeta}-g(X_N^{h,y_0,\zeta})|^2]$. The discrete costs, associated with (y_0,ζ^{y_0}) and (y_0,ζ), are given by

$$J_0^{h,y_0}=J_0^{h,y_0,\zeta^{y_0}}\quad\text{and}\quad J_0^{h,y_0,\zeta}=\mathbb{E}[\mathcal{Y}_0^{h,y_0,\zeta}].$$

Here, the *discrete stochastic cost* is given by

$$\mathcal{Y}_0^{h,y_0,\zeta}=g(X_N^{h,y_0,\zeta})+\sum_{k=0}^{N-1}f(t_k,X_k^{h,y_0,\zeta},Z_k^{h,y_0,\zeta})h-\sum_{k=0}^{N-1}\langle Z_k^{h,y_0,\zeta},\Delta W_k\rangle.\tag{4.13}$$

Using the stochastic cost, we have by a substitution that

$$\mathrm{MSE}(y_0)=\inf_{\zeta}\mathbb{E}\big[\big|\mathcal{Y}_0^{h,y_0,\zeta}-y_0\big|^2\big].\tag{4.14}$$

Note that this minimization problem coincides with (4.10), with the only difference that here we only search for ζ (and not for y_0). The method used to solve this problem is the one

proposed in [4], but with a fixed y_0, *i.e.*, the discretized $Z-$process is parameterized with neural networks and the mean-squared terminal condition is minimized.

Figure 4.1 shows $y_0 \mapsto \text{MSE}(y_0)$ and $y_0 \mapsto J_0^{h,y_0}$ for two different Linear-Quadratic (LQ) control problems respectively, a one-dimensional and a two-dimensional problem. The left figure shows that there is a minimum of MSE at the correct Y_0 and for this problem the method converges. For the right figure, it is clear that there is no minimum of MSE around Y_0, or anywhere in the range. When y_0 and ζ are jointly optimized, the method has no chance to converge for this problem.

Both problems considered in this section are of the form (4.30) with parameters as in Section 4.5.1.1 for the two-dimensional problem and $A = B = C = R_x = R_u = G = 1$ and $\sigma = 0.5$ for the one-dimensional problem.

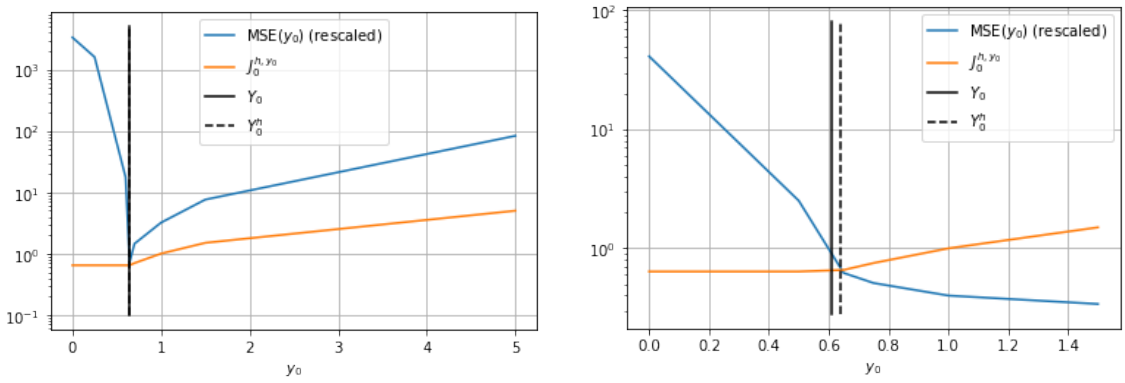

Figure 4.1: Demonstration of the performance of the direct extension of the deep BSDE method to FBSDEs corresponding to two LQ control problems. **Left:** A one-dimensional problem with $N = 10$ time steps. **Right:** A two-dimensional problem with $N = 100$ time steps.

We identify three distinct cases from Figure 4.1:

- $y_0 \approx Y_0$: In this case, MSE is the mean squared error of the discretized FBSDE and it is most reasonable that an approximation of (X, Y, Z) is obtained by optimizing over ζ.

- $y_0 > Y_0$: Under this assumption, any ζ attaining the infimum in (4.12) satisfies $J_0^{h,y_0} = y_0$. Thus minimizing (4.12) is the same as finding the ζ that generates discrete cost y_0 and that at the same time minimizes the mean squared error in the terminal condition. A ζ with cost $y_0 > Y_0$ has the possibility to generate a lower $\text{MSE}(y_0) < \text{MSE}(Y_0)$, depending on y_0 and the problem at hand. This is only possible if the coupling of Z in b is strong enough, so that $g(X_T)$ can be efficiently controlled by Z. In the case with no coupling, i.e., for a BSDE, $y_0 > Y_0$ leads to an MSE of the magnitude $y_0 - Y_0$, and thus MSE is increasing in this regime. This is the reason why non-coupled BSDEs, as in [4], or weakly coupled FBSDEs, as in [87], can be approximated with the deep BSDE method. The left plot of Figure 4.1 shows this favorable behaviour, while the right plot has a decreasing MSE and has no chance to converge. We also see that the cost increases linearly for $y_0 > Y_0$ according to $J_0^{h,y_0} = y_0$.

- $y_0 < Y_0$: Since Y_0 is (approximately) a lower bound of $(y_0, \zeta) \mapsto J^{h,y_0,\zeta}$, it holds that any ζ attaining the infimum in $\zeta \mapsto \mathbb{E}[|Y_N^{h,y_0,\zeta} - g(X_N^{h,y_0,\zeta})|^2]$ also minimizes the cost functional $\zeta \mapsto J_0^{h,y_0,\zeta}$. But y_0 does not enter $\mathcal{Y}_0^{h,y_0,\zeta}$ explicitly. Therefore, the minimizer of

$\zeta \mapsto \mathbb{E}[\mathcal{Y}_0^{h,y_0,\zeta}]$ does not depend on y_0. Thus, fixing $y_0 < Y_0$ and optimizing ζ approximates a solution to the control problem but not to the FBSDE. This can clearly be seen in Figure 4.1 from the cost being constant for $y_0 < Y_0$ in both plots. It is also clear that the MSE increases for decreasing $y_0 < Y_0$.

To further visualize the three cases, Figure 4.2 shows the empirical means and 90% credible regions (defined as the area between the 5 :th and the 95 :th empirical percentiles at each time point) for the true and approximated Y and Z processes of the two-dimensional LQ control problem discussed above. In the top row, we see that, in the case $y_0 \approx Y_0$, the two components of the Z-process are very well approximated, but the time discretization error of Y is visible. In the middle row, for $y_0 > Y_0$, we see what is expected based on the discussion above. The Y process satisfies the terminal value but is otherwise fundamentally distinct from the true Y, and Z is different and oscillating (it is specified to have cost y_0). In the bottom row, the case $y_0 < Y_0$ is shown. Just as explained above, the control problem is solved and therefore the Z process is well approximated. It should though be noted that this is only true since the map $p \mapsto v^*(t,x,p)$, for this specific problem, is invertible. Otherwise, one optimal ζ, in a set of many optimal Markov maps, is approximated. Thus, the control problem is approximately solved, however, the Z-component of the FBSDE is unlikely to be accurate. The Y process is shifted by $y_0 - Y_0$ and the terminal value is naturally not satisfied.

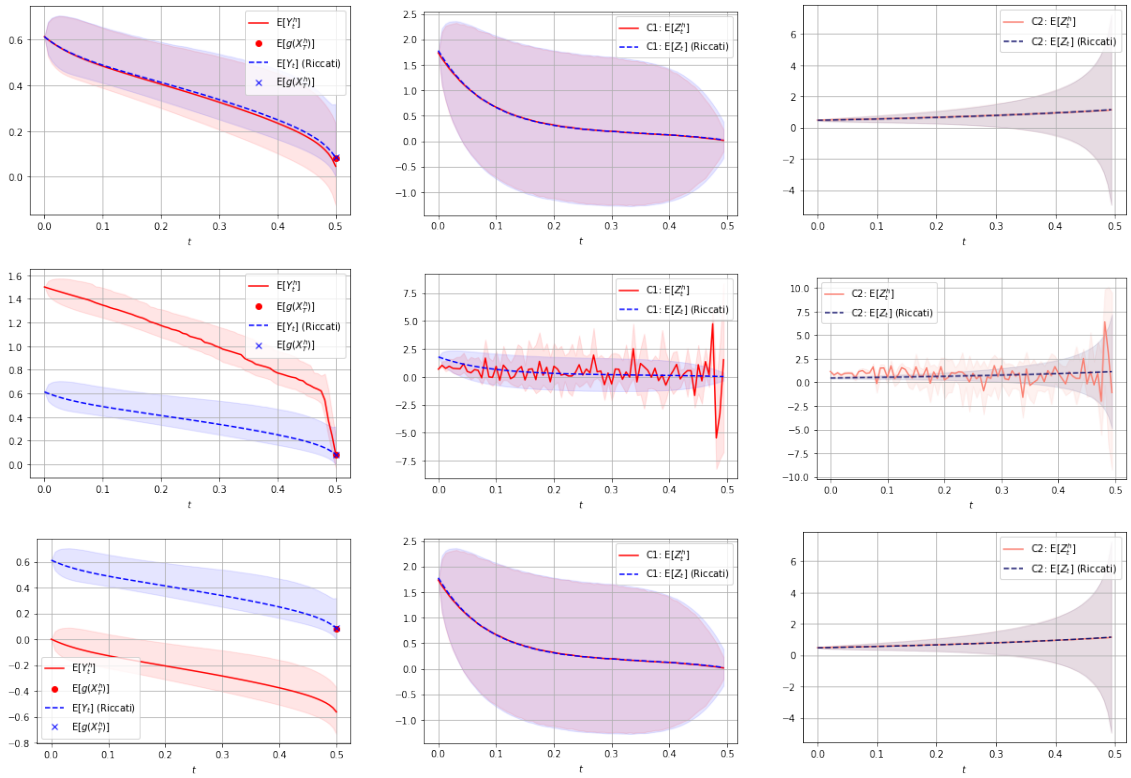

Figure 4.2: Demonstration of the performance of the Deep FBSDE solver with fixed initial condition for $N = 100$ time steps. The shaded areas represent the domain in which 90% of all trajectories lie (the area is bounded by the 5:th and the 95:th empirical percentiles). **Left** to **right**: Sample means of the approximate and the semi-analytic (Riccati solutions) $Y-$process and the first and and second components of the $Z-$process (C1 and C2). **Top** to **bottom**: Initial conditions $\hat{y}_0 = 0.612 \approx Y_0$, $\hat{y}_0 = 1.5 > Y_0$ and $\hat{y}_0 = 0.0 < Y_0$.

When searching for interesting example problems, we learned that it is much easier to find problems that do not converge than finding problems that do converge.

4.2.4 A robust deep FBSDE method

Having observed the problems with the direct extension of the deep BSDE method to FBSDEs, we here discretize the alternative formulation (4.8) of the FBSDE to obtain an alternative family of deep FBSDE methods. It reads:

$$\begin{cases} \underset{\zeta}{\text{minimize}}\ \Phi_{\lambda,h}(\zeta) = \mathbb{E}[\mathcal{Y}_0^{h,\zeta}] + \lambda \text{Var}[\mathcal{Y}_0^{h,\zeta}], \quad \text{where,} \\ \mathcal{Y}_0^{h,\zeta} := g(X_N^{h,\zeta}) + \sum_{k=0}^{N-1} f(t_k, X_k^{h,\zeta}, Z_k^{h,\zeta})h - \sum_{k=0}^{N-1} \langle Z_k^{h,\zeta}, \Delta W_k \rangle, \\ X_n^{h,\zeta} = x_0 + \sum_{k=0}^{n-1} b(t_k, X_k^{h,\zeta}, Z_k^{h,\zeta})h + \sum_{k=0}^{n-1} \sigma(t_k, X_k^{h,\zeta})\Delta W_k, \\ Y_n^{h,\zeta} = \mathbb{E}[\mathcal{Y}_0^{h,\zeta}] - \sum_{k=0}^{n-1} f(t_k, X_k^{h,\zeta}, Z_k^{h,\zeta})h + \sum_{k=0}^{n-1} \langle Z_k^{h,\zeta}, \Delta W_k \rangle, \\ Z_k^{h,\zeta} = \zeta_k(X_k^{h,\zeta}). \end{cases} \tag{4.15}$$

The main purpose of the current chapter is the theoretical and numerical error analysis of (4.15).

4.2.5 Related methods and comparison

In a comparison with the current literature we focus on methods for solving stochastic control problem, or the associate FBSDE, by deep learning in a global way, in the sense that only one global optimization problem is solved.

In the early paper [6], time discrete stochastic optimal control problems were solved with deep learning. No explicit connections to FBSDEs, or even to stochastic control in continuous time, were made. The feedback maps for the controls at all time steps were optimized over a family of neural networks, to minimize the discrete cost functional. This methodology is similar to our method when $\lambda = 0$. It only differs in its approximation of the feedback map for u instead of the Markov map for Z. The connection between our proposed method and the deep BSDE method, proposed in [4], is the second term in the loss function of (4.15), corresponding to $\lambda \to \infty$. From (4.9) it becomes clear that, if the driver does not take Y as an input, then this term coincides with the loss function used in the deep BSDE method.

The recent paper [86] is, to the best of our knowledge, the first to introduce the variance penalty term in (4.15) for an FBSDE obtained from the dynamic programming principle. The authors of [110] also use the variance penalty term, but for general decoupled FBSDEs, *i.e.,* not in the context of stochastic control. In [86], the problem is approached differently in that they have one network for the Markov map for Z and one for the feedback map for the control. In [86], the variance term in the loss function is presented as a measurability loss. Their motivation is that if the BSDE is solved, then the initial state of the $Y-$process is $\mathcal{F}_0-$measurable and hence the variance is zero. Although, one should bear in mind that this is only true for the continuous BSDE. In the discretized version, the $Y-$process is not measurable

and no arguments for convergence of the discretization are presented in [86]. However, their numerical results are convincing and an error analysis similar to the one presented in Section 4.4 in the present chapter could possibly be carried out also in their setting.

Another method which also makes use of the connection to stochastic control (of Hamiltonian systems) was proposed in [85]. In that paper, the stochastic maximum principle approach to stochastic control was used, which results in a different type of FBSDEs, compared to the one obtained with the dynamic programming principle, that we consider in this chapter. More precisely, it is the $Y-$process instead of the $Z-$process which is connected to the control of the SDE. A similarity is that, in both in the paper and in this chapter, the algorithms use a method similar to that in [6], to include the cost in the objective function.

Summarizing, only the method in [86] is fully comparable to ours, as it is designed to solve the same problem. It has the variance term but not the mean in its loss function. By introducing a loss function that includes both we are able to prove convergence and obtain a robust method.

4.2.6 Decoupled FBSDEs and why coupled FBSDEs are important

It is sometimes claimed in the literature that since coupled FBSDEs can be transformed into decoupled BSDE, it is sufficient to have schemes for the latter, see *e.g.,* [87]. Here, we explain this claim and why we, from a practical and application viewpoint, do not agree. If $\psi\colon [0,T]\times\mathbb{R}^d\times\mathbb{R}^\ell\to\mathbb{R}^d$ is sufficiently regular, then it holds by the Itô formula for $Y_t^\psi = V(t,X_t^\psi)$ and $Z_t^\psi = \sigma^T(t,X_t^\psi)\mathrm{D}_x V(t,X_t^\psi)$ that

$$\begin{cases} X_t^\psi = x_0 + \int_0^t \left(b(s,X_s^\psi,Z_s^\psi) - \psi(s,X_s^\psi,Z_s^\psi)\right)\mathrm{d}s + \int_0^t \sigma(s,X_s^\psi)\mathrm{d}W_s, \\ Y_t^\psi = g(X_T^\psi) + \int_t^T \left(f(s,X_s^\psi,Z_s^\psi) - \left\langle(\sigma(s,X_s^\psi)\sigma^T(s,X_s^\psi))^{-1}\sigma(s,X_s^\psi)Z_s, \psi(s,X_s^\psi,Z_s^\psi)\right\rangle\right)\mathrm{d}s \\ \qquad\quad - \int_t^T \langle Z_s^\psi, \mathrm{d}W_s\rangle, \quad t\in[0,T]. \end{cases}$$

Thus taking ψ in such a way that $b-\psi$ does not depend on Z, decouples the FBSDE resulting in a BSDE, where the forward equation has no coupling with the backward equation. From this observation, it might be tempting to approximate the optimal Markov map $\sigma^T(t,x)\mathrm{D}_x V(t,x)$ with the deep BSDE method. The problem with this approach is that the deep BSDE method will learn $\sigma^T(t,x)\mathrm{D}_x V(t,x)$ well only around typical trajectories of X^ψ, but not around those of X. While there is empirical evidence that the deep BSDE method overcomes the curse of dimensionality, it does not at all approximate the solution everywhere, but only around the typical solution trajectories of the forward equation. Since X is controlled, it has a different dynamics than X^ψ and this may ruin the applicability of the deep BSDE method for control problems, if the feedback map is desired. If only an approximation of the solution to the HJB equation (the value) is sought, as in [4], then decoupling is feasible.

4.3 Fully implementable scheme and neural network regression

In this section, we describe how the discrete variational problem (4.15) is approximated with the help of neural network regression. In principle, other function approximators could be used,

but neural network regression is arguably one of the most suitable choices due to the ability to approximate complicated high-dimensional functions. Although neural networks have shown empirically high quality results in many different fields, there are still many open convergence questions related to the optimization procedure of the loss function. On the other hand, the Universal Approximation Theorem (UAT) [111] guarantees that under certain conditions, there is a neural network, sufficiently deep and wide, such that it is possible to approximate a large class of continuous functions to any, pre-specified degree of accuracy.

4.3.1 Fully implementable algorithms

Without further specifications, (4.15) assumes exact optimization over an unspecified set of functions ζ and the exact computation of expectations. To define a fully implementable scheme, the Markov maps $\zeta_0, \dots, \zeta_{N-1}$ are approximated with neural networks $\phi_0^{\theta_0}, \dots, \phi_{N-1}^{\theta_{N-1}}$ with parameters $\theta = (\theta_0, \dots, \theta_{N-1})$ in some parameter space Θ. We specify them in further detail below. Moreover, expectations are approximated with batch Monte-Carlo simulation. Let $K_{\text{epochs}} \geq 1, K_{\text{batch}} \geq 1$ be the number of epochs and the number of batches per epoch, respectively. Let further $M_{\text{train}}, M_{\text{batch}} \geq 1$ be the size of the training data set and batch, respectively. We assume that $M_{\text{train}}/(2M_{\text{batch}}) = K_{\text{batch}} \in \mathbb{N}$. Training data are M_{train} independent realizations of the Wiener increments $\Delta W_0, \dots, \Delta W_{N-1} \sim \mathcal{N}(0, h)$ and the training data are reused in K_{epoch} epochs. The training is initialized by random sampling of $\theta^0 \in \Theta$. For each update step in an epoch of the training algorithm, we take $2M_{\text{batch}}$ Wiener increments $\Delta W_0(m), \dots, \Delta W_{N-1}(m)$, $m = 1, 2, \dots, 2M_{\text{batch}}$ from the training data set that were not previously used during the epoch and update θ by approximate optimization of the following problem:

$$
\begin{cases}
\underset{\theta\in\Theta}{\text{minimize}}\ \mathcal{L}_{\lambda,h}(\theta) = \dfrac{1}{M_{\text{batch}}}\left(\displaystyle\sum_{m=1}^{M_{\text{batch}}} \mathcal{Y}_0^{h,\theta}(m) + \lambda \cdot \sum_{m=M_{\text{batch}}+1}^{2M_{\text{batch}}} |g(X_N^{h,\theta}(m)) - Y_N^{h,\theta}(m)|^2\right), \\
\mathcal{Y}_0^{h,\theta}(m) := g(X_N^{h,\theta}(m)) + \displaystyle\sum_{k=0}^{N-1} f(t_k, X_k^{h,\theta}(m), Z_k^{h,\theta}(m))h - \sum_{k=0}^{N-1} \langle Z_k^{h,\theta}(m), \Delta W_k(m)\rangle, \\
X_n^{h,\theta}(m) = x_0 + \displaystyle\sum_{k=0}^{n-1} b(t_k, X_k^{h,\theta}(m), Z_k^{h,\theta}(m))h + \sum_{k=0}^{n-1} \sigma(t_k, X_k^{h,\theta}(m))\Delta W_k(m), \\
Y_n^{h,\theta}(m) = \dfrac{1}{M_{\text{batch}}} \displaystyle\sum_{r=1}^{M_{\text{batch}}} \mathcal{Y}_0^{h,\theta}(r) - \sum_{k=0}^{n-1} f(t_k, X_k^{h,\theta}(m), Z_k^{h,\theta}(m))h + \sum_{k=0}^{n-1} \langle Z_k^{h,\theta}(m), \Delta W_k(m)\rangle, \\
Z_k^{h,\theta}(m) = \phi_k^{\theta_k}(X_k^{h,\theta}(m)).
\end{cases}
\tag{4.16}
$$

When all training data has been used, a new epoch starts. When the K_{epoch}:th epoch is finished, the algorithm terminates. The neural network parameters at termination are θ^*. It is an approximation of the parameters θ^{**} that optimize (4.16) in the limit $M_{\text{batch}} \to \infty$. To complement (4.16), Algorithm 1 details the training procedure.

It should be noted that the expected value of the stochastic sum in $\mathcal{Y}_0^{h,\theta}$ equals zero. Therefore, the algorithm would also work without it, but a practical reason to keep it is that it decreases the variance of $\mathcal{Y}_0^{h,\theta}$ significantly, and this requires fewer Monte-Carlo samples to achieve the same accuracy.

Input: Initialization of neural network parameters, $\{\theta_0(1), \dots, \theta_{N-1}(1)\}$, and, for $0 \leq k \leq 2M_{\text{train}}$ and $0 \leq n \leq N-1$, Wiener increments $\Delta W_n(k)$.

Output: Approximation of the Markov map for $(Z_t)_{t\in[0,T]}$ at the time discrete mesh points.

for $k = 1, 2, \dots, K_{\text{batch}}$ ($K_{\text{batch}} = M_{\text{train}}/(2M_{\text{batch}})$ is the number of batches.) (should be carried out sequentially) **do**

  for $1 \leq m \leq M_{\text{batch}}$ (may be carried out in parallel) **do**

    Set $X_0^{h,\theta}(m) = x_0$

    for $n = 0, \dots, N-1$ (should be done sequentially) **do**

      $Z_n^{h,\theta}(m) = \phi_n(X_n^{h,\theta}(m) \,|\, \theta_n(k))$

      $X_{n+1}^{h,\theta}(m) = X_n^{h,\theta}(m) + b\big(t_n, X_n^{h,\theta}(m), Z_n^{h,\theta}(m)\big)h + \sigma(t_n, X_n^{h,\theta}(m))\Delta W_n(m)$

    end

  end

  for $m \in \{M_{\text{batch}}+1, \dots, 2M_{\text{batch}}\}$ (may be carried out in parallel) **do**

    Set $X_0^{h,\theta}(m) = x_0$ and $Y_0^{h,\theta}(m) = \frac{1}{M_{\text{batch}}} \sum_{m=M_{\text{batch}}+1}^{2M_{\text{batch}}} g(X_N^{h,\theta}(m)) +$
    $\sum_{n=0}^{N-1} f\big(t_n, X_n^{h,\theta}(m), Z_n^{h,\theta}(m)\big)h - \sum_{n=0}^{N-1} \langle Z_n^{h,\theta}(m), \Delta W_n(m)\rangle$

    for $n = 0, \dots, N-1$ (should be carried out sequentially) **do**

      $Z_n^{h,\theta}(m) = \phi_n(X_n^{h,\theta}(m) \,|\, \theta_n(k))$

      $X_{n+1}^{h,\theta}(m) = X_n^{h,\theta}(m) + b\big(t_n, X_n^{h,\theta}(m), Z_n^{h,\theta}(m)\big)h + \sigma(t_n, X_n^{h,\theta}(m))\Delta W_n(m)$

      $Y_{n+1}^{h,\theta}(m) = Y_n^{h,\theta}(m) - f\big(t_n, X_n^{h,\theta}(m), Z_n^{h,\theta}(m)\big)h + \langle Z_n^{h,\theta}(m), \Delta W_n(m)\rangle$

    end

  end

  $\theta = \{\theta_0, \theta_1, \dots, \theta_{N-1}\}$ (trainable parameters)

  $\mathcal{L}(\theta) = \frac{1}{M_{\text{batch}}}\Big(\sum_{m=1}^{M_{\text{batch}}} Y_0^{\theta}(m) + \lambda \sum_{m=M_{\text{batch}}+1}^{2M_{\text{batch}}} |g(X_N^{h,\theta}(m)) - Y_N^{h,\theta}(m)|^2\Big)$
  (Loss-function)

  $\theta(k+1) \leftarrow \arg\min_\theta \mathcal{L}(\Theta)$ (some optimization algorithm, usually of gradient decent type)

end

Algorithm 1: Pseudo-code of one epoch of the neural network training

4.3.2 Specification of the neural networks

Here, we introduce the neural networks that we use in our implementations in Section 4.5. The generality is kept to a minimum and more general architectures are of course possible. For each $\phi_k^{\theta_k}\colon \mathbb{R}^d \to \mathbb{R}^\ell$, a fully connected neural network with two hidden layers with 20 nodes in each layer and a ReLU activation function $\mathfrak{R}(x) = \max(0, x)$ acting element-wise is employed. More precisely, the ϕ_k^θ is of the form

$$\phi_k^{\theta_k}(x) = W_k^3\mathfrak{R}(W_k^2\mathfrak{R}(W_k^1 x + b_k^1) + b_k^2) + b_k^3,$$

with weight matrices $W_k^1 \in \mathbb{R}^{20\times d}$, $W_k^2 \in \mathbb{R}^{20\times 20}$, $W_k^3 \in \mathbb{R}^{\ell\times 20}$ and bias vectors $b_k^1, b_k^2 \in \mathbb{R}^{\mathfrak{N}}$, $b_k^3 \in \mathbb{R}^\ell$, and $\theta_k = (W_k^1, W_k^2, W_k^3, b_k^1, b_k^2, b_k^3)$, where the matrices are considered vectorized before concatenation.

4.4 Convergence analysis

In this section, we primarily provide an error analysis for (4.15), i.e., for the semidiscretization in time. In Section 4.4.1, we introduce notation and spaces, and, in Section 4.4.2, we present the setting and some further notation. Two technical results on strong and weak convergence are stated and proved in Section 4.4.3. These two results are used in Section 4.4.4 to prove the error in the objective function, in the initial and terminal value of Y and for the variance of the stochastic cost. The results hold under an assumption on the regularity of the exact continuous and discrete Markov maps. Convergence of the latter to the former is not assumed. Section 4.4.5 contains strong convergence of the FBSDE under either the stronger assumption of small time T or convergence of the discrete Markov maps. A discussion about a full error analysis for the fully implementable scheme (4.16) is presented in Section 4.4.6.

4.4.1 Notation and spaces

For Euclidean spaces $\mathbb{R}^k$, $k \geq 1$, we denote by $\|\cdot\|$ the 2-norm without specifying the dimension. Let $\mathcal{S}^2(\mathbb{R}^k)$ and $\mathcal{H}^2(\mathbb{R}^k)$ be the spaces of all progressively measurable stochastic processes $y, z\colon [0,T]\times\Omega \to \mathbb{R}^k$, for which

$$\|y\|_{\mathcal{S}^2(\mathbb{R}^k)} = \sup_{t\in[0,T]} \Big(\mathbb{E}\big[\|y_t\|^2\big]\Big)^{\frac{1}{2}} < \infty, \quad \text{and} \quad \|z\|_{\mathcal{H}^2(\mathbb{R}^k)} = \left(\mathbb{E}\Big[\int_0^T \|z_t\|^2 \mathrm{d}t\Big]\right)^{\frac{1}{2}} < \infty,$$

respectively. For a discretization, $0 = t_0 < t_1 < \cdots < t_N = T$ with $t_{n+1} - t_n = h$, for all n, the space $\mathcal{S}_h(\mathbb{R}^k)$ is the space of all $\mathcal{F}^h$-adapted, square integrable and discrete stochastic processes $y : \{0, 1, \ldots, N\}\times\Omega \to \mathbb{R}^d$, where $\mathcal{F}_n^h = \sigma(\Delta W_m,\ m = 0, \ldots, n-1)$. For $k_1, k_2, k_3, k_4 \geq 1$, $\ell_1, \ell_2, \ell_3 \geq 0$ and regular $S_i \subseteq \mathbb{R}^{k_i}$, $i = 1, 2, 3, 4$, by $C_{\mathrm{b}}^{\ell_1,\ell_2,\ell_3}(S_1\times S_2\times S_3; S_4)$, we denote the space of all functions $\phi\colon S_1\times S_2\times S_3 \to S_4$, whose derivatives up to orders ℓ_1, ℓ_2, ℓ_3 exist, are continuous and bounded. We equip it with the semi-norms

$$|\phi|_\gamma = \sup_{x\in S_1\times S_2\times S_3} \|\partial^\gamma \phi(x)\|, \quad i \in \{1, \ldots, \ell\},$$

where $\gamma = \{(i_1, i_2, i_3) : i_j \in \{0, , \dots, \ell_j\},\ j = 1, 2, 3\}$ are multi-indices and $\partial^\gamma = \partial_1^{i_1}\partial_2^{i_2}\partial_3^{i_3}$ with ∂_j^i denoting i:th partial derivative in variable j. The set $B(\ell_1, \ell_2, \ell_3)$ denotes all multi-indices of length 3 that have exactly one non-zero index. We only use multi-indices in $B(\ell_1, \ell_2, \ell_3)$ and do therefore not impose restrictions on the cross derivatives, as is otherwise common. For functions with fewer than three variables, we reduce the number of indices accordingly and in the semi-norms we write $|\phi|_{i_1,i_2,i_3} = |\phi|_{(i_1,i_2,i_3)}$. For $\alpha \in (0, 1]$, $k_1, k_2, k_3 \geq 1$, $\ell_1, \ell_2 \geq 0$ and regular $S_i \subseteq \mathbb{R}^{k_i}$, $i = 1, 2, 3$, by $C_{\mathrm{H,b}}^{\alpha,\ell_1,\ell_2}([0, T] \times S_1 \times S_2; S_3)$, we denote the space of all functions $\phi\colon [0, T] \times S_1 \times S_2 \to S_3$ that are α-Hölder continuous in time and whose derivatives of order ℓ_1, ℓ_2 exist, are continuous and bounded and satisfy the property

$$\begin{aligned}\|\!|\!|\phi|\!|\!|_{\alpha,\ell_1,\ell_2} = \sup_{t\in[0,T]} \|\phi(t, 0, 0)\| + \sum_{\gamma\in B} \sup_{t\in[0,T]} |\phi(t, \cdot, \cdot)|_\gamma \\ + \sup_{(x_1,x_2)\in S_1\times S_2} \sup_{t_1,t_2\in[0,T],t_1\neq t_2} \frac{\mathbf{1}_{(0,1]}(\alpha)\|\phi(t_1, x_1, x_2) - \phi(t_2, x_1, x_2)\|}{(1 + \|x_1\| + \|x_2\|)|t_2 - t_1|^\alpha} < \infty.\end{aligned}$$

Again, when there is only one space variable, we reduce the number of indices. When there is no risk of confusion, we write $|\cdot|_{\alpha,0,0}$ to denote the second term defining the $\|\!|\!|\cdot|\!|\!|_{\alpha,\ell_1,\ell_2}$-norm, and let $|\cdot|_{0,i,0}$ and $|\cdot|_{0,0,i}$ coincide with the semi-norms on $C_{\mathrm{b}}^{\alpha,\ell_1,\ell_2}([0, T] \times S_1 \times S_2; S_3)$ with the same notation. For $\alpha = 0$, we let $C_{\mathrm{H,b}}^{0,\ell_1,\ell_2}([0, T] \times S_1 \times S_2; S_3) = C_{\mathrm{b}}^{0,\ell_1,\ell_2}([0, T] \times S_1 \times S_2; S_3)$.

For any function or process R defined on $[0, T]$, we denote by $\check{R}$ the discrete time function or process defined by $\check{R}_n = R_{t_n}$, $n = 0, \dots, N$. For any discrete function or process R_h defined on $0, \dots, N$, we write $\hat{R}_h$ for the continuous time function or process defined by the piecewise constant interpolation $\hat{R}_{h,t} = R_{h,n}$ for $t \in [t_n, t_{n+1})$ and $\hat{R}_{h,T} = R_{h,N}$.

4.4.2 Setting and spaces of Markov maps

Let $(W_t)_{t\in[0,T]}$ be a k-dimensional Brownian motion on a filtered probability space $(\Omega, \mathcal{F}, (\mathcal{F}_t)_{t\in[0,T]}, \mathbb{P})$, and $(\alpha, \beta) \in [0, \frac{1}{2}] \times \{1\}$ or $(\alpha, \beta) = (1, 2)$. The coefficients $b\colon [0, T] \times \mathbb{R}^d \times \mathbb{R}^k \to \mathbb{R}^d$, $\sigma\colon [0, T] \times \mathbb{R}^d \to \mathbb{R}^{d\times k}$, $f\colon [0, T] \times \mathbb{R}^d \times \mathbb{R}^k \to \mathbb{R}$ and $g\colon \mathbb{R}^d \to \mathbb{R}$ satisfy $b \in C_{\mathrm{H,b}}^{\alpha,\beta,\beta}([0, T] \times \mathbb{R}^d \times \mathbb{R}^k; \mathbb{R}^d)$, $\sigma \in C_{\mathrm{H,b}}^{\alpha,\beta}([0, T] \times \mathbb{R}^d; \mathbb{R}^{d\times k})$, $f \in C_{\mathrm{H,b}}^{\alpha,\beta,\beta}([0, T] \times \mathbb{R}^d \times \mathbb{R}^k; \mathbb{R})$, $g \in C_{\mathrm{b}}^1(\mathbb{R}^d; \mathbb{R})$. In the case $\alpha = 1$, we assume that $\mathrm{D}_x\sigma = 0$.

We next introduce families of Markov maps. Let $\mathcal{Z} = \mathcal{Z}(b, \sigma, f, g)$ be the collection of all measurable functions $\zeta : [0, T] \times \mathbb{R}^d \to \mathbb{R}^k$ with the property that the stochastic processes $(X^\zeta, Y^\zeta, Z^\zeta)_{\zeta\in\mathcal{Z}} \subset \mathcal{S}^2(\mathbb{R}^d) \times \mathcal{S}^2(\mathbb{R}) \times \mathcal{H}^2(\mathbb{R}^d)$ satisfying for all $t \in [0, T]$, $\mathbb{P}$-almost surely (4.8), are well defined. We write $\mathcal{Z}^{\alpha,\beta} = \mathcal{Z} \cap C_{\mathrm{H,b}}^{\alpha,\beta}([0, T] \times \mathbb{R}^d; \mathbb{R}^k)$. For the discrete equations, we introduce for every $h \in (0, 1)$ analogously $\mathcal{Z}_h = \mathcal{Z}_h(b, \sigma, f, g)$ to be the collection of all measurable functions $\zeta\colon \{0, \dots, N_h - 1\} \times \mathbb{R}^d \to \mathbb{R}^k$, with the property that $(X^{h,\zeta}, Y^{h,\zeta}, Z^{h,\zeta})_{\zeta\in\mathcal{Z}_h} \subset \mathcal{S}_h^2(\mathbb{R}^d) \times \mathcal{S}_h^2(\mathbb{R}) \times \mathcal{S}_h^2(\mathbb{R})$ satisfying for all $n \in \{0, \dots, N_h\}$, $\mathbb{P}$-almost surely (4.15) are well-defined. We write $\mathcal{Z}_h^\beta = \mathcal{Z}_h \cap (C_{\mathrm{b}}^\beta(\mathbb{R}^d; \mathbb{R}^k))^{N+1}$.

By introducing assumptions on the Markov maps we eliminate the need for assuming smoothness, Lipschitz, polynomial growth, monotonicity, coercivity or other conditions on the coefficients, for the existence and uniqueness of solutions for (4.8) and (4.15). For unfortunate choices of b, σ, f, g, the spaces $\mathcal{Z}$ and $\mathcal{Z}_h$ might be empty and results hold by default, but given regular b, σ, f, g it is not hard, using available solution theory, to find $\zeta \in \mathcal{Z}$ and $\zeta_h \in \mathcal{Z}_h$. Still, the entire spaces might be hard to represent but this is not of central importance.

In Assumption 3 below, the regularities of the optimal ζ^* and ζ_h^*, solving (4.8) and (4.15) are though of importance. Thus classical solution theory for SDE, FBSDE and regularity theory for optimal Markov maps in discrete and continuous time are required for verifying our assumptions for concrete examples. The latter is not well developed, see the discussion prior to Assumption 3 below. Thus part of our assumptions require further theoretical development to be verified, but our results show what is required.

4.4.3 Auxiliary lemmata on strong and weak convergence for SDEs

In the proof of our convergence results in Subsection 4.4.4, we rely on the strong convergence result, stated next. It contains both classical strong convergence of the Euler-Maruyama scheme for Hölder continuous coefficients, including strong order 1 for additive noise, but also a non-standard type of strong convergence result for processes that have drift coefficients that for each step size, agree between the grid points, but whose coefficients do not necessarily converge as the step size tends to zero.

Lemma 1. *Suppose the setting of Subsection 4.4.2 holds. Let $a \in C^{\alpha,\beta}_{\mathrm{H,b}}([0,T]\times\mathbb{R}^d;\mathbb{R}^d)$ and $a_h\colon [0,T]\times\mathbb{R}^d \to \mathbb{R}^d$, $h\in(0,1)$ be a family of functions that are constant on each interval $[t_n, t_{n+1})$, satisfy $a_h([0,T],\cdot)\subset C^\beta(\mathbb{R}^d;\mathbb{R}^d)$ and $\sup_{h\in(0,1)}|||a_h|||_{0,\beta}<\infty$, let $\mathcal{X},\mathcal{X}^{1,h}\in\mathcal{S}^2(\mathbb{R}^d)$, $h\in(0,1)$ be the unique solutions to*

$$\begin{aligned}
\mathrm{d}\mathcal{X}_t &= a(t,\mathcal{X}_t)\mathrm{d}t + \sigma(t,\mathcal{X}_t)\mathrm{d}W_t,\ t\in(0,T]; \quad \mathcal{X}_0 = x_0,\\
\mathrm{d}\mathcal{X}^{1,h}_t &= a_h(t,\mathcal{X}^{1,h}_t)\mathrm{d}t + \sigma(t,\mathcal{X}^{1,h}_t)\mathrm{d}W_t,\ t\in(0,T]; \quad \mathcal{X}^{1,h}_0 = x_0,,
\end{aligned}$$

and $\mathcal{X}^{2,h},\mathcal{X}^{3,h}\in\mathcal{S}^2_h(\mathbb{R}^d)$, $h\in(0,1)$ be the unique solutions to

$$\begin{aligned}
\mathcal{X}^{2,h}_{n+1} &= \mathcal{X}^{2,h}_n + a(t_n,\mathcal{X}^{2,h}_n)h + \sigma(t_n,\mathcal{X}^{2,h}_n)\Delta W_n,\ n\in\{0,\dots,N-1\}; \quad \mathcal{X}^{2,h}_0 = x_0,\\
\mathcal{X}^{3,h}_{n+1} &= \mathcal{X}^{3,h}_n + a_h(t_n,\mathcal{X}^{3,h}_n)h + \sigma(t_n,\mathcal{X}^{3,h}_n)\Delta W_n,\ n\in\{0,\dots,N-1\}; \quad \mathcal{X}^{3,h}_0 = x_0.
\end{aligned}$$

It holds that

$$\sup_{h\in(0,1)}\|\mathcal{X}^{1,h}\|_{\mathcal{S}^2(\mathbb{R}^d)} + \sup_{h\in(0,1)}\|\hat{\mathcal{X}}^{2,h}\|_{\mathcal{S}^2(\mathbb{R}^d)} + \sup_{h\in(0,1)}\|\hat{\mathcal{X}}^{3,h}\|_{\mathcal{S}^2(\mathbb{R}^d)} < \infty, \tag{4.17}$$

and there exists a constant C, such that

$$\lim_{h\to 0} h^{-\alpha}\Big(\|\mathcal{X}-\hat{\mathcal{X}}^{2,h}\|_{\mathcal{S}^2(\mathbb{R}^d)} + \|\mathcal{X}^{1,h}-\hat{\mathcal{X}}^{3,h}\|_{\mathcal{S}^2(\mathbb{R}^d)}\Big) = \begin{cases} 0 & \text{if } \alpha = 0,\\ C & \text{otherwise.}\end{cases}$$

Proof. Due to the assumption on a_h, there exists a finite constant C such that

$$\|a_h(t,x)\| \le \sup_{h\in(0,1)}\sup_{t\in[0,T]}\big(\|a_h(t,0)\| + |a_h(t,\cdot)|_1\big)\|x\| \le C\|x\|.$$

This is enough to show the first assertion, the stability estimate, using a Gronwall argument. For the convergence, we rely on [112], which covers both the cases $\alpha\in[0,\frac{1}{2}]$ and $\alpha=1$. It is based on the fact that bistability and consistency of order α, in the sense of [112], imply convergence of order α. We start with $\mathcal{X}^{1,h}\to\mathcal{X}^{3,h}$ and the convergence $\mathcal{X}^{2,h}\to\mathcal{X}$ is immediate afterwards.

Since the drift coefficients of $\mathcal{X}^{1,h}$ and $\mathcal{X}^{3,h}$ are both h-dependent, are not assumed to converge, what we want to prove, i.e., $\lim_{h\to 0} \|\mathcal{X}_t^{1,h} - \hat{\mathcal{X}}_t^{3,h}\|_{\mathcal{S}^2(\mathbb{R}^d)} = 0$ does not say anything about convergence of $\mathcal{X}^{3,h}$ and $\mathcal{X}^{1,h}$. In particular, $\mathcal{X}_t^{1,h_1} - \hat{\mathcal{X}}_t^{3,h_2}$ is in general not small for $h_1 \neq h_2$. Due to this special setting, we have to verify that the proofs of [112] are still valid.

We start with proving bistability. First, in [112, Lemma 4.1], it is proven that for small enough h the numerical scheme is bijective. For our proof, it is important that the upper bound for this property does not depend on h. From the proof, it is clear that this bound depends on the reciprocal of $\sup_{h\in(0,1)} \sup_{t\in[0,T]} |a_h(t,\cdot)|_1$ and is therefore bounded away from zero by assumption. By following the proof of [112, Lemma 4.2] line by line, the bistability constants are bounded by $\sup_{h\in(0,1)} \sup_{t\in[0,T]} |a_h(t,\cdot)|_1$ and its reciprocal when Assumptions (S1) and (S2) in [112] hold. It remains to prove (S1) and (S2) with uniform constants. First, (S1) is trivially satisfied with $L = 0$ for any explicit scheme. By inspection of the proof of [112, Theorem 3.3], we see that, for $\alpha \in [0, \frac{1}{2}]$, corresponding to the stochastic θ-method with $\theta = 0$, (S2) is satisfied for all h with $L = 3T \sup_{h\in(0,1)} \sup_{t\in[0,T]} |a_h(t,\cdot)|_1^2 + 12d|\sigma|_1^2$. For $\alpha = 1$, (S2) holds for the same L as the schemes are the same. This concludes the proof of bistability.

Consistency for $\alpha \in [0, \frac{1}{2}]$ is obtained by observing that the proof in [112], referring for details to [113], only relies on (4.17) and the $\frac{1}{2}$-Hölder continuity of a_h between the grid points, and this is clearly satisfied since a_h is constant on $[t_n, t_{n+1})$. For consistency in case $\alpha = 1$, we rely on the proof of consistency for the Itô-Taylor scheme of order one, i.e., the Milstein scheme, coinciding with our scheme under the assumption of additive noise. It suffices to note that the analysis is conducted per interval, and to check that f_α is globally Lipschitz continuous for $\alpha \in \mathcal{B}(\mathcal{A}_1)$, in the notation of [112]. For this to hold, it suffices that $\sup_{h\in(0,1)} \sup_{t\in[0,T]} |a_h(t,\cdot)|_2 < \infty$. By assumption, this completes the proof for $\mathcal{X}^{1,h}$ and $\mathcal{X}^{3,h}$. For $\mathcal{X}^{2,h}$ and $\mathcal{X}$, the convergence is included in [112], up to the order of consistency, which in our setting is lower when $\alpha \in [0, \frac{1}{2})$. It is straight forward to use α-Hölder continuity for $\alpha \in (0, \frac{1}{2})$, exactly as in the case $\alpha = \frac{1}{2}$, and this way get consistency of order α. For $\alpha = 0$, one uses dominated convergence instead to obtain the convergence without order. This is valid under our assumptions. □

Our next Lemma is a weak convergence type result. With our assumption of low regularity on the FBSDE coefficients, the weak rate does not exceed the strong rate and therefore the proof relies on the strong convergence results of Lemma 1. Obtaining weak convergence order $\alpha \in (\frac{1}{2}, 1)$ for multiplicative noise, would otherwise require a Hölder condition on the third derivative of b, σ, f, g and the case $\alpha = 1$ requires four derivatives [114, 115].

Lemma 2. *Suppose the setting of Subsection 4.4.2 holds. For all functions $\zeta \in \mathcal{Z}^{\alpha,\beta}$, collection of functions $\zeta_h \in \mathcal{Z}_h^\beta$, $h \in (0,1)$, satisfying $\sup_{h\in(0,1)} |||\hat{\zeta}_h|||_{0,\beta} < \infty$, and $\lambda \geq 0$, there exists a constant C, such that*

$$\lim_{h\to 0} h^{-\alpha}\Big(|\Phi_\lambda(\zeta) - \Phi_{\lambda,h}(\check{\zeta})| + |\Phi_\lambda(\hat{\zeta}_h) - \Phi_{\lambda,h}(\zeta_h)|\Big) = \begin{cases} 0 & \text{if } \alpha = 0, \\ C & \text{otherwise.} \end{cases}$$

Proof. We start with the first term and observe that

$$|\Phi_\lambda(\zeta) - \Phi_{\lambda,h}(\check{\zeta})| \leq |\mathbb{E}[\mathcal{Y}_0^\zeta - \mathcal{Y}_0^{h,\check{\zeta}}]| + \lambda|\mathrm{Var}(\mathcal{Y}_0^\zeta) - \mathrm{Var}(\mathcal{Y}_0^{h,\check{\zeta}})|. \tag{4.18}$$

For the proof of both terms of (4.18), we rely on convergence of $(\mathbb{E}[(\mathcal{Y}_0^{\zeta} - \mathcal{Y}_0^{h,\check{\zeta}})^2])^{\frac{1}{2}}$, which we next prove, beginning with $\alpha > 0$. For $t \in [t_n, t_{n+1})$, denote $\bar{t} = t_n$. We start by observing that by the definition,

$$\begin{aligned}
\mathcal{Y}_0^{\zeta} - \mathcal{Y}_0^{h,\check{\zeta}} &= g(X_T^{\zeta}) - g(X_T^{h,\check{\zeta}}) + \int_0^T \left(f(t, X_t^{\zeta}, \zeta(t, X_t^{\zeta})) - f(\bar{t}, \hat{X}_t^{h,\check{\zeta}}, \zeta(\bar{t}, \hat{X}_t^{h,\check{\zeta}}))\right)\mathrm{d}t \\
&\quad + \int_0^T \langle \zeta(t, X_t^{\zeta}) - \zeta(\bar{t}, \hat{X}_t^{h,\check{\zeta}}), \mathrm{d}W_t \rangle \\
&= g(X_T^{\zeta}) - g(X_T^{h,\check{\zeta}}) + \int_0^T \left(f(t, X_t^{\zeta}, \zeta(t, X_t^{\zeta})) - f(t, \hat{X}_t^{h,\check{\zeta}}, \zeta(t, \hat{X}_t^{h,\check{\zeta}}))\right)\mathrm{d}t \\
&\quad + \int_0^T \left(f(t, \hat{X}_t^{h,\check{\zeta}}, \zeta(t, \hat{X}_t^{h,\check{\zeta}})) - f(\bar{t}, \hat{X}_t^{h,\check{\zeta}}, \zeta(\bar{t}, \hat{X}_t^{h,\check{\zeta}}))\right)\mathrm{d}t \\
&\quad + \int_0^T \langle \zeta(t, X_t^{\zeta}) - \zeta(t, \hat{X}_t^{h,\check{\zeta}}), \mathrm{d}W_t \rangle + \int_0^T \langle \zeta(t, X_t^{\zeta}) - \zeta(t, \hat{X}_t^{h,\check{\zeta}}), \mathrm{d}W_t \rangle \\
&\quad + \int_0^T \langle \zeta(t, \hat{X}_t^{h,\check{\zeta}}) - \zeta(\bar{t}, \hat{X}_t^{h,\check{\zeta}}), \mathrm{d}W_t \rangle
\end{aligned}$$

From the Itô Isometry and the triangle inequality, we have

$$\begin{aligned}
(\mathbb{E}[(\mathcal{Y}_0^{\zeta} - \mathcal{Y}_0^{h,\check{\zeta}})^2])^{\frac{1}{2}} &\leq (\mathbb{E}[(g(X_T^{\zeta}) - g(X_T^{h,\check{\zeta}}))^2])^{\frac{1}{2}} \\
&\quad + \int_0^T (\mathbb{E}[\left(f(t, X_t^{\zeta}, \zeta(t, X_t^{\zeta})) - f(t, \hat{X}_t^{h,\check{\zeta}}, \zeta(t, \hat{X}_t^{h,\check{\zeta}}))\right)^2])^{\frac{1}{2}}\mathrm{d}t \\
&\quad + \int_0^T \mathbb{E}[\left(f(t, \hat{X}_t^{h,\check{\zeta}}, \zeta(t, \hat{X}_t^{h,\check{\zeta}})) - f(\bar{t}, \hat{X}_t^{h,\check{\zeta}}, \zeta(\bar{t}, \hat{X}_t^{h,\check{\zeta}}))\right)^2])^{\frac{1}{2}}\mathrm{d}t \\
&\quad + \Big(\int_0^T \mathbb{E}[\|\zeta(t, X_t^{\zeta}) - \zeta(t, \hat{X}_t^{h,\check{\zeta}})\|^2]\mathrm{d}t\Big)^{\frac{1}{2}} \\
&\quad + \Big(\int_0^T \mathbb{E}[\|\zeta(t, \hat{X}_t^{h,\check{\zeta}}) - \zeta(\bar{t}, \hat{X}_t^{h,\check{\zeta}})\|^2]\mathrm{d}t\Big)^{\frac{1}{2}} \\
&= I_1 + I_2 + I_3 + I_4 + I_5.
\end{aligned}$$

For I_1, we use Lipschitz continuity of g and the Cauchy-Schwarz inequality, to get

$$I_1 \leq |g|_1 \mathbb{E}[\|X_T^{\zeta} - X_T^{h,\check{\zeta}}\|] \leq |g|_1 (\mathbb{E}[\|X_T^{\zeta} - X_T^{h,\check{\zeta}}\|^2])^{\frac{1}{2}} \leq |g|_1 \|X^{\zeta} - \hat{X}^{h,\check{\zeta}}\|_{\mathcal{S}^2(\mathbb{R}^d)}. \tag{4.19}$$

The function $\phi(t,x) = f(t,x,\zeta(t,x))$ is uniformly Lipschitz continuous in x with Lipschitz constant $C_\phi = |f|_{0,1,0} + |f|_{0,0,1}|\zeta|_{0,1}$. This implies, together with the Cauchy-Schwarz inequality,

$$I_2 \leq C_\phi \int_0^T (\mathbb{E}[\|X_t^{\zeta} - \hat{X}_t^{h,\check{\zeta}}\|^2])^{\frac{1}{2}}\mathrm{d}t \leq C_\phi T \|X^{\zeta} - \hat{X}^{h,\check{\zeta}}\|_{\mathcal{S}^2(\mathbb{R}^d)}. \tag{4.20}$$

Using the assumptions of b and ζ, it is easily verified that ϕ belongs to $C^{\alpha,\beta}_{\mathrm{H,b}}([0,T] \times \mathbb{R}^d; \mathbb{R}^d)$ and therefore, by (4.19), (4.20) and Lemma 1, we have $I_1 + I_2 \leq Ch^{\alpha}$. For the term I_3, we see that

$$\begin{aligned}
I_3 &\leq \left(|f|_{\alpha,0,0} + |f|_{0,0,1}|\zeta|_{\alpha,0}\right)\left(1 + \sup_{h \in (0,1)} \|\hat{X}^{h,\check{\zeta}}\|_{\mathcal{S}^2(\mathbb{R}^d)}\right) \sum_{n=0}^{N-1} \int_{t_n}^{t_{n+1}} (t - t_n)^{\alpha}\mathrm{d}t \\
&= \frac{|f|_{\alpha,0,0} + |f|_{0,0,1}|\zeta|_{\alpha,0}}{1+\alpha}\left(1 + \sup_{h \in (0,1)} \|\hat{X}^{h,\check{\zeta}}\|_{\mathcal{S}^2(\mathbb{R}^d)}\right) N h^{1+\alpha} \leq Ch^{\alpha}.
\end{aligned}$$

Similarly, for I_4 and I_5 we have

$$I_4 \leq |\zeta|_{0,1}\Big(\int_0^T \mathbb{E}[\|X_t^\zeta - \hat{X}_t^{h,\check{\zeta}}\|^2]\mathrm{d}t\Big)^{\frac{1}{2}} \leq |\zeta|_{0,1}T^{\frac{1}{2}}\|X^\zeta - \hat{X}^{h,\check{\zeta}}\|_{\mathcal{S}^2(\mathbb{R}^d)} \leq Ch^\alpha,$$

and

$$\begin{aligned} I_5 &\leq |\zeta|_{\alpha,0}\Big(1 + \sup_{h\in(0,1)}\|\hat{X}^{h,\check{\zeta}}\|_{\mathcal{S}^2(\mathbb{R}^d)}\Big)\left(\sum_{n=0}^{N-1}\int_{t_n}^{t_{n+1}}|t-t_n|^{2\alpha}\mathrm{d}t\right)^{\frac{1}{2}} \\ &= \frac{|\zeta|_{\alpha,0}}{1+2\alpha}\Big(1 + \sup_{h\in(0,1)}\|\hat{X}^{h,\check{\zeta}}\|_{\mathcal{S}^2(\mathbb{R}^d)}\Big)(Nh^{1+2\alpha})^{\frac{1}{2}} \leq Ch^\alpha. \end{aligned}$$

For $\alpha = 0$, the terms I_1, I_2, I_4 need no special treatment. For I_3, we notice that the function ϕ is only continuous in t and convergence without rate holds by the Dominated Convergence Theorem. This is verified by noting that

$$\begin{aligned} &\sup_{t\in[0,T]}\Big(\mathbb{E}\Big[\Big(f(t,\hat{X}_t^{h,\check{\zeta}},\zeta(t,\hat{X}_t^{h,\check{\zeta}})) - f(\bar{t},\hat{X}_t^{h,\check{\zeta}},\zeta(\bar{t},\hat{X}_t^{h,\check{\zeta}}))\Big)^2\Big]\Big)^{\frac{1}{2}} \\ &\quad\leq 2\sup_{t\in[0,T]}\Big(\mathbb{E}\Big[\big(f(t,\hat{X}_t^{h,\check{\zeta}},\zeta(t,\hat{X}_t^{h,\check{\zeta}}))\big)^2\Big]\Big)^{\frac{1}{2}} \\ &\quad\leq 2\sup_{t\in[0,T]}\Big(\mathbb{E}\Big[\big(f(t,\hat{X}_t^{h,\check{\zeta}},\zeta(t,\hat{X}_t^{h,\check{\zeta}}) - f(t,0,0)\big)^2\Big]\Big)^{\frac{1}{2}} + 2\sup_{t\in[0,T]}|f(t,0,0)| \\ &\quad\leq 2\sup_{t\in[0,T]}\big(|f(t,\cdot,\cdot)|_{1,0} + |f(t,\cdot,\cdot)|_{0,1}\big(|\zeta(t,\cdot)|_1 + \|\zeta(t,0)\|\big)\big)\sup_{h\in(0,1)}\|\hat{X}^{h,\check{\zeta}}\|_{\mathcal{S}^2(\mathbb{R}^d)} \\ &\qquad+ 2\sup_{t\in[0,T]}\big|\mathbb{E}[f(t,0,0)\big|. \end{aligned}$$

Due to the assumptions, the right-hand side is finite. The term I_5 admits a similar treatment and we refrain from giving details. This proves that

$$\lim_{h\to 0}\big(\mathbb{E}[(\mathcal{Y}_0^\zeta - \mathcal{Y}_0^{h,\check{\zeta}})^2]\big)^{\frac{1}{2}} = \begin{cases} 0 & \text{if } \alpha = 0, \\ C & \text{otherwise.} \end{cases} \tag{4.21}$$

For the first term in (4.18), this and the fact that $|\mathbb{E}[\mathcal{Y}_0^\zeta - \mathcal{Y}_0^{h,\check{\zeta}}]| \leq (\mathbb{E}[(\mathcal{Y}_0^\zeta - \mathcal{Y}_0^{h,\check{\zeta}})^2])^{\frac{1}{2}}$ prove the convergence.

For the second term in (4.18), we use the conjugate rule, Cauchy-Schwarz' inequality, the triangle inequality, to get

$$\begin{aligned} &|\mathrm{Var}(\mathcal{Y}_0^\gamma) - \mathrm{Var}(\mathcal{Y}_0^{h,\check{\gamma}})| \\ &\quad= \Big|\mathbb{E}\Big[|\mathbb{E}[\mathcal{Y}_0^\gamma] - \mathcal{Y}_0^\gamma|^2 - |\mathbb{E}[\mathcal{Y}_0^{h,\check{\gamma}}] - \mathcal{Y}_0^{h,\check{\gamma}}|^2\Big]\Big| \\ &\quad= |\mathbb{E}[(\mathbb{E}[\mathcal{Y}_0^\gamma - \mathcal{Y}_0^{h,\check{\gamma}}] + \mathcal{Y}_0^{h,\check{\gamma}} - \mathcal{Y}_0^\gamma)(\mathbb{E}[\mathcal{Y}_0^\gamma + \mathcal{Y}_0^{h,\check{\gamma}}] - \mathcal{Y}_0^{h,\check{\gamma}} - \mathcal{Y}_0^\gamma)]| \\ &\quad\leq \Big(\mathbb{E}\Big[\big(\mathbb{E}[\mathcal{Y}_0^\gamma - \mathcal{Y}_0^{h,\check{\gamma}}] + \mathcal{Y}_0^{h,\check{\gamma}} - \mathcal{Y}_0^\gamma\big)^2\Big]\Big)^{\frac{1}{2}}\Big(\mathbb{E}\Big[\big(\mathbb{E}[\mathcal{Y}_0^\gamma + \mathcal{Y}_0^{h,\check{\gamma}}] - \mathcal{Y}_0^{h,\check{\gamma}} - \mathcal{Y}_0^\gamma\big)^2\Big]\Big)^{\frac{1}{2}} \\ &\quad\leq \Big(|\mathbb{E}[\mathcal{Y}_0^\gamma - \mathcal{Y}_0^{h,\check{\gamma}}]| + (\mathbb{E}[(\mathcal{Y}_0^{h,\check{\gamma}} - \mathcal{Y}_0^\gamma)^2])^{\frac{1}{2}}\Big)\Big(|\mathbb{E}[\mathcal{Y}_0^\gamma + \mathcal{Y}_0^{h,\check{\gamma}}]| + (\mathbb{E}[(\mathcal{Y}_0^{h,\check{\gamma}} + \mathcal{Y}_0^\gamma)^2])^{\frac{1}{2}}\Big) \\ &\quad\leq 4\Big((\mathbb{E}[(\mathcal{Y}_0^\zeta)^2])^{\frac{1}{2}} + \sup_{h\in(0,1)}(\mathbb{E}[(\mathcal{Y}_0^{h,\check{\zeta}})^2])^{\frac{1}{2}}\Big)(\mathbb{E}[(\mathcal{Y}_0^{h,\check{\gamma}} - \mathcal{Y}_0^\gamma)^2])^{\frac{1}{2}}. \end{aligned}$$

Together with (4.21), this completes the proof for the term $|\Phi_\lambda(\zeta) - \Phi_{\lambda,h}(\check{\zeta})|$.

For the second term, $|\Phi_\lambda(\hat{\zeta}_h) - \Phi_{\lambda,h}(\zeta_h)|$, we define the functions $a_h \colon [0,T] \times \mathbb{R}^d \to \mathbb{R}^d$, $h \in (0,1)$, by $a_h = \hat{c}_h$, where $c_h \colon \{0,\dots,N\} \times \mathbb{R}^d \to \mathbb{R}^d$ are given by $c_{h,n}(x) = b(t_n, x, \zeta_{h,n}(x))$. By the assumptions on b and ζ_h, we have that a_h satisfy the assumption of Lemma 1. This implies that $\lim_{h\to 0} \|X^{\zeta_h} - \hat{X}^{h,\zeta_h}\|_{\mathcal{S}^2(\mathbb{R}^d)} = 0$ and $\|X^{\zeta_h} - \hat{X}^{h,\zeta_h}\|_{\mathcal{S}^2(\mathbb{R}^d)} \leq Ch^\alpha$ for $\alpha > 0$. This fact, together with the same calculations as for I_1, I_2, I_4, I_5, and noting that the analogous for I_3 is zero, completes the proof. □

4.4.4 Time discretization error of the initial and terminal values

This and the next section contain our main results. The results are based on an assumption of regularity of the Markov maps $\zeta^* \in \mathcal{Z}$ and $\zeta^*_{\lambda,h} \in \mathcal{Z}_h$, $h \in (0,1)$, $\lambda \geq 0$, of the FBSDE and its discretizations, respectively. The regularity has to be verified for specific problems and this, in particular for the discrete problem, is a non-trivial and also not a well studied problem, see [116, 117, 118]. For the continuous problem, it breaks down into existence and uniqueness of a solution of the FBSDE and regularity of the solution to the HJB equation. In this chapter, we provide only the abstract error analysis. Our assumption is next stated.

Assumption 3. *There exists a unique minimizer $\zeta^* \in \mathcal{Z}$ of $\zeta \mapsto \Phi_1(\zeta)$, it belongs to $\mathcal{Z}^{\alpha,\beta}$ and satisfies* $\mathrm{Var}(\mathcal{Y}_0^{\zeta^*}) = 0$. *Moreover, for all $\lambda \geq 0$, there exists a collection of minimizers $\zeta^*_{\lambda,h} \in \mathcal{Z}_h$, $h \in (0,1)$, of $\zeta \mapsto \Phi_{\lambda,h}(\zeta)$ that for each h belongs to $\mathcal{Z}_h^\beta$ and satisfies* $\sup_{h\in(0,1)} |||\hat{\zeta}^*_{\lambda,h}|||_{0,\beta} < \infty$.

Our first result concerns the convergence of the objective function. The proof relies on a split of the error into two error terms, one containing only ζ^* and one containing only $\zeta^*_{\lambda,h}$. This way, we avoid getting an error bound containing the error between the continuous and discrete Markov maps. The result is used throughout our proofs, except in Proposition 4, where we have the error of the Markov map in the bound.

Theorem 5. *Suppose the setting of Subsection 4.4.2, let Assumption 3 hold and $\lambda \geq 0$. Then, $|\Phi^* - \Phi^*_{\lambda,h}| \to 0$ as $h \to 0$, and if $\alpha > 0$, then there exists a constant C, independent of h, such that*

$$|\Phi^* - \Phi^*_{\lambda,h}| \leq Ch^\alpha.$$

Proof. We start by noting that, since $\mathrm{Var}(\mathcal{Y}_0^{\zeta^*}) = 0$, it holds that ζ^* is optimal for all Φ_λ, $\lambda \geq 0$, and $\Phi^* = \Phi^*_\lambda := \Phi_\lambda(\zeta^*)$. Because of optimality of $\zeta^* \in \mathcal{Z}$ and since $\hat{\zeta}^*_h \in \mathcal{Z}$, it holds that $\Phi^* = \Phi^*_\lambda \leq \Phi_\lambda(\hat{\zeta}^*_h)$. This implies

$$\Phi^* \leq \Phi_\lambda(\hat{\zeta}^*_{\lambda,h}) = \Phi^*_{\lambda,h} + \Phi_\lambda(\hat{\zeta}^*_{\lambda,h}) - \Phi^*_{\lambda,h} \leq \Phi^*_{\lambda,h} + |\Phi_\lambda(\hat{\zeta}^*_{\lambda,h}) - \Phi^*_{\lambda,h}|.$$

Similarly, from the optimality of $\zeta^*_{\lambda,h} \in \mathcal{Z}_h$ and since $\check{\zeta}^* \in \mathcal{Z}_h$, it holds $\Phi^*_{\lambda,h} \leq \Phi_{\lambda,h}(\check{\zeta}^*)$. We have

$$\Phi^*_{\lambda,h} \leq \Phi_{\lambda,h}(\check{\zeta}^*) = \Phi^* + \Phi_{\lambda,h}(\check{\zeta}^*) - \Phi^* \leq \Phi^* + |\Phi_{\lambda,h}(\check{\zeta}^*) - \Phi^*|.$$

The two inequalities equivalently read $\Phi^* - \Phi^*_{\lambda,h} \leq |\Phi(\hat{\zeta}^*_{\lambda,h}) - \Phi^*_{\lambda,h}|$ and $\Phi^*_{\lambda,h} - \Phi^* \leq |\Phi_{\lambda,h}(\check{\zeta}^*) - \Phi^*|$, and thus

$$|\Phi^* - \Phi^*_{\lambda,h}| \leq \max(|\Phi(\hat{\zeta}^*_{\lambda,h}) - \Phi^*_{\lambda,h}|, |\Phi_{\lambda,h}(\check{\zeta}^*) - \Phi^*|) \leq |\Phi(\hat{\zeta}^*_{\lambda,h}) - \Phi^*_{\lambda,h}| + |\Phi_{\lambda,h}(\check{\zeta}^*) - \Phi^*|.$$

The convergence is given by our assumption and Lemma 2. This completes the proof. □

A consequence of the convergence of the objective function is the convergence of the two components of the objective, given that $\lambda > 0$. This is stated in our next theorem.

Theorem 6. *Suppose the setting of Subsection 4.4.2, let Assumption 3 hold, $\lambda > 0$, $\mathcal{Y}_0^{h,\lambda} := \mathcal{Y}_0^{h,\zeta^*_{h,\lambda}}$ and $Y_0^{h,\lambda} := \mathbb{E}[\mathcal{Y}_0^{h,\lambda}]$. Then, $|Y_0 - Y_0^{h,\lambda}| + \mathrm{Var}(\mathcal{Y}_0^{h,\lambda}) \to 0$ as $h \to 0$, and if $\alpha > 0$, there exists a constant C, independent of h, such that*

$$|Y_0 - Y_0^{h,\lambda}| + \mathrm{Var}(\mathcal{Y}_0^{h,\lambda}) \leq Ch^\alpha.$$

Proof. We start with the proof for the variance and use both the sequences $(\zeta^*_{0,h})_{h\in(0,1)}$ and $(\zeta_{\lambda,h})_{h\in(0,1)}$. The proof relies on a squeezing argument. Adding and subtracting terms and using the triangle inequality, yields

$$\lambda\mathrm{Var}(\mathcal{Y}_0^{h,\lambda}) \leq |Y_0 - Y_0^{h,\lambda} - \lambda\mathrm{Var}(\mathcal{Y}_0^{h,\lambda})| + |Y_0 - Y_0^{h,0}| + |Y_0^{h,0} - Y_0^{h,\lambda}|. \tag{4.22}$$

From the assumption $\mathrm{Var}(\mathcal{Y}_0) = 0$, it holds that $\Phi^* = Y_0$ and Theorem 5 gives

$$|Y_0 - Y_0^{h,\lambda} - \lambda\mathrm{Var}(\mathcal{Y}_0^{h,\lambda})| + |Y_0 - Y_0^{h,0}| = |\Phi^* - \Phi^*_{\lambda,h}| + |\Phi^* - \Phi^*_{0,h}| \leq Ch^\alpha. \tag{4.23}$$

For the third term on the right-hand side of (4.22), we first notice that, since $\zeta_{0,h}$ is a minimizer of $\zeta \mapsto Y_0^{h,0}$, it holds that $Y_0^{h,\lambda} - Y_0^{h,0} \geq 0$. This fact, adding $\lambda\mathrm{Var}(\mathcal{Y}_0^{h,\lambda})$, adding and subtracting Y_0, using the triangle inequality and using (4.23), gives us

$$\begin{aligned} |Y_0^{h,0} - Y_0^{h,\lambda}| = Y_0^{h,\lambda} - Y_0^{h,0} &\leq Y_0^{h,\lambda} + \lambda\mathrm{Var}(\mathcal{Y}_0^{h,\lambda}) - Y_0 - Y_0^{h,0} + Y_0 \\ &\leq |Y_0 - Y_0^{h,\lambda} - \lambda\mathrm{Var}(\mathcal{Y}_0^{h,\lambda})| + |Y_0 - Y_0^{h,\lambda}| \leq Ch^\alpha. \end{aligned} \tag{4.24}$$

Now, (4.22)–(4.24) complete the proof for the variance. For the convergence of $Y_0^{h,\lambda}$, we conclude

$$|Y_0 - Y_0^{h,\lambda}| \leq |Y_0 - Y_0^{h,\lambda} - \lambda\mathrm{Var}(\mathcal{Y}_0^{h,\lambda})| + \lambda\mathrm{Var}(\mathcal{Y}_0^{h,\lambda}) \leq Ch^\alpha.$$

□

From (4.9) and Theorem 6, we directly get convergence in the terminal value and we can also conclude strong convergence of $y_0^{h,\lambda}$. This is stated in the following two corollaries.

Corollary 4.4.1. Suppose the setting of Subsection 4.4.2, let Assumption 3 hold and $\lambda > 0$. Then, $\mathbb{E}[(g(X_N^{h,\lambda}) - Y_N^{h,\lambda})^2] \to 0$ as $h \to 0$, and if $\alpha > 0$, then there exists a constant C, independent of h, such that

$$\Big(\mathbb{E}\Big[\big(g(X_N^{h,\lambda}) - Y_N^{h,\lambda}\big)^2\Big]\Big)^{\frac{1}{2}} \leq Ch^{\frac{\alpha}{2}}.$$

Corollary 4.4.2. Suppose the setting of Subsection 4.4.2, let Assumption 3 hold and $\lambda > 0$. Then, $(\mathbb{E}[(\mathcal{Y}_0 - \mathcal{Y}_0^{h,\lambda})^2])^{\frac{1}{2}} \to 0$ as $h \to 0$, and if $\alpha > 0$, then there exists a constant C,

independent of h, such that

$$\Big(\mathbb{E}\Big[(\mathcal{Y}_0 - \mathcal{Y}_0^{h,\lambda})^2\Big]\Big)^{\frac{1}{2}} \leq Ch^{\frac{\alpha}{2}}.$$

Proof. From Theorem 6 and the triangle inequality, it holds

$$\begin{aligned}\Big(\mathbb{E}\Big[(\mathcal{Y}_0 - \mathcal{Y}_0^{h,\lambda})^2\Big]\Big)^{\frac{1}{2}} &\leq \Big(\mathbb{E}\Big[(Y_0 - Y_0^{h,\lambda} + \mathcal{Y}_0 - \mathcal{Y}_0^{h,\lambda})^2\Big]\Big)^{\frac{1}{2}} + |Y_0 - Y_0^{h,\lambda}| \\ &= \Big(\mathrm{Var}(\mathcal{Y}_0^{h,\lambda})\Big)^{\frac{1}{2}} + |Y_0 - Y_0^{h,\lambda}| \leq Ch^{\frac{\alpha}{2}}.\end{aligned}$$

This proves the corollary. □

4.4.5 Time discretization error of the FBSDE

While the error analysis for the initial and terminal values in the previous subsection required only Assumption 3, the strong error analysis of (X, Y, Z) requires more. In Theorem 7, we prove strong convergence for small horizon T. This should be compared with the convergence result in [87], that also has a very restrictive assumption on T. Compared to [87], whose bound contains the error in the terminal value, we have no such term. In Proposition 4, we prove the strong convergence without a restriction on T with the cost of having a bound containing the error between the Markov maps ζ^* and $\zeta^*_{\lambda,h}$.

Theorem 7. *Suppose the setting of Subsection 4.4.2, let Assumption 3 hold, $\alpha, \lambda > 0$, and*

$$\max\left(T^{\frac{1}{2}}|f|_{0,0,1}, \frac{5T(T|f|_{0,1,0} + |g|_1)|b|^2_{0,0,1}\exp(5T(|b|_{0,1,0}T + |\sigma|_{0,1}))}{1 - T^{\frac{1}{2}}|f|_{0,0,1}}\right) < 1.$$

Then, there exists a constant C, independent of h, such that

$$\|X - \hat{X}^{h,\lambda}\|_{\mathcal{S}^2(\mathbb{R}^d)} + \|Y - \hat{Y}^{h,\lambda}\|_{\mathcal{S}^2(\mathbb{R}^d)} + \|Z - \hat{Z}^{h,\lambda}\|_{\mathcal{H}^2(\mathbb{R}^k)} \leq Ch^{\frac{\alpha}{2}}.$$

Proof. We start by noting that for $t \in [0, T]$, it holds

$$Y_t - Y_t^{h,\lambda} = Y_0 - Y_0^{h,\lambda} - \int_0^t (f(s, X_s, Z_s) - f(\bar{s}, \hat{X}_s^{h,\lambda}, \hat{Z}_s^{h,\lambda}))\mathrm{d}s + \int_0^t (Z_s - \hat{Z}_s^{h,\lambda})\mathrm{d}W_s. \quad (4.25)$$

By the Itô Isometry, substitution of (4.25) with $t = T$, and the triangle inequalities, we have

$$\begin{aligned}\|Z - \hat{Z}^{h,\lambda}\|_{\mathcal{H}^2(\mathbb{R}^k)} &= \Big(\mathbb{E}\Big[\int_0^T \|Z_t - \hat{Z}_t^{h,\lambda}\|^2\mathrm{d}t\Big]\Big)^{\frac{1}{2}} = \Big(\mathbb{E}\Big[\Big(\int_0^T (Z_t - \hat{Z}_t^{h,\lambda})\mathrm{d}W_t\Big)^2\Big]\Big)^{\frac{1}{2}} \\ &= \left(\mathbb{E}\left[\left(Y_0 - Y_0^{h,\lambda} + g(X_T) - Y_N^{h,\lambda} - \int_0^T (f(t, X_t, Z_t) - f(\bar{t}, \hat{X}_t^{h,\lambda}, \hat{Z}_t^{h,\lambda}))\mathrm{d}t\right)^2\right]\right)^{\frac{1}{2}} \\ &\leq |Y_0 - Y_0^{h,\lambda}| + \left(\mathbb{E}\left[\left(g(X_T) - g(X_N^{h,\lambda})\right)^2\right]\right)^{\frac{1}{2}} + \left(\mathbb{E}\left[\left(g(X_N^{h,\lambda}) - Y_N^{h,\lambda}\right)^2\right]\right)^{\frac{1}{2}} \\ &\quad + \Big(\mathbb{E}\Big[\Big(\int_0^T (f(t, X_t, Z_t) - f(\bar{t}, \hat{X}_t^{h,\lambda}, \hat{Z}_t^{h,\lambda}))\mathrm{d}t\Big)^2\Big]\Big)^{\frac{1}{2}}.\end{aligned}$$

The first three terms are by Theorem 6 and Corollary 4.4.1, bounded from below by $Ch^{\frac{\alpha}{2}} + |g|_1\|X - \hat{X}^{h,\lambda}\|_{\mathcal{S}^2(\mathbb{R}^d)}$. By similar arguments as those for I_2, I_3 in the proof of Lemma 2, it

holds

$$\begin{aligned}&\Big(\mathbb{E}\Big[\Big(\int_0^T (f(t,X_t,Z_t)-f(\bar{t},\hat{X}_t^{h,\lambda},\hat{Z}_t^{h,\lambda}))\mathrm{d}t\Big)^2\Big]\Big)^{\frac{1}{2}}\\&\quad\le T|f|_{\alpha,0,0}(1+\alpha)^{-1}h^\alpha+T|f|_{0,1,0}\|X-\hat{X}^{h,\lambda}\|_{\mathcal{S}^2(\mathbb{R}^d)}+T^{\frac{1}{2}}|f|_{0,0,1}\big\|\hat{Z}^{h,\lambda}-Z\big\|_{\mathcal{H}^2(\mathbb{R}^k)}.\end{aligned}\tag{4.26}$$

By a kickback argument and by assumption, it holds

$$\big\|Z-\hat{Z}^{h,\lambda}\big\|_{\mathcal{H}^2(\mathbb{R}^k)}\le Ch^\alpha+\frac{T|f|_{0,1,0}+|g|_1}{1-T^{\frac{1}{2}}|f|_{0,0,1}}\|X-\hat{X}^{h,\lambda}\|_{\mathcal{S}^2(\mathbb{R}^d)}.\tag{4.27}$$

We next approach the error $\|X-\hat{X}^{h,\lambda}\|_{\mathcal{S}^2(\mathbb{R}^d)}$. Let $\Xi^{h,\lambda}\in\mathcal{S}^2(\mathbb{R}^d)$, $h\in(0,1)$, be the family of stochastic processes that for all $t\in[0,T]$, $\mathbb{P}$-a.s., satisfy

$$\Xi_t^{h,\lambda}=x_0+\int_0^t b(\bar{s},\Xi_s^{h,\lambda},Z_s^{h,\lambda})\mathrm{d}s+\int_0^t\sigma(\bar{s},\Xi_s^{h,\lambda})\mathrm{d}W_s.$$

Using the triangle inequality, we get

$$\|X-X^{h,\lambda}\|_{\mathcal{S}^2(\mathbb{R}^d)}\le\|X-\Xi^{h,\lambda}\|_{\mathcal{S}^2(\mathbb{R}^d)}+\|\Xi^{h,\lambda}-\hat{X}^{h,\lambda}\|_{\mathcal{S}^2(\mathbb{R}^d)}.$$

By the arguments used repeatedly in the proof of Lemma 1 and by assumptions, it holds

$$\begin{aligned}&\mathbb{E}\big[\|X_t-\Xi_t^{h,\lambda}\|^2\big]\\&\qquad\le Ch^\alpha+5|b|_{0,0,1}^2T\|Z-Z^{h,\lambda}\|_{\mathcal{H}^2(\mathbb{R}^k)}^2+5\big(|b|_{0,1,0}T+|\sigma|_{0,1}\big)\int_0^t\mathbb{E}\big[\|X_s-\Xi_s^{h,\lambda}\|^2\big]\mathrm{d}s.\end{aligned}$$

From this, it follows by the Gronwall lemma that

$$\mathbb{E}\big[\|X_t-\Xi_t^{h,\lambda}\|^2\big]\le\exp\Big(5T\big(|b|_{0,1,0}T+|\sigma|_{0,1}\big)\Big)\Big(Ch^\alpha+5|b|_{0,0,1}^2T\|Z-Z^{h,\lambda}\|_{\mathcal{H}^2(\mathbb{R}^k)}^2\Big).$$

A use of Lemma 1 yields $\|\Xi^{h,\lambda}-\hat{X}^{h,\lambda}\|_{\mathcal{S}^2(\mathbb{R}^d)}\le Ch^\alpha$. We conclude that

$$\|X-\hat{X}^{h,\lambda}\|_{\mathcal{S}^2(\mathbb{R}^d)}\le Ch^\alpha+5T|b|_{0,0,1}^2\exp(5T(|b|_{0,1,0}T+|\sigma|_{0,1}))\|Z-\hat{Z}^{h,\lambda}\|_{\mathcal{H}^2(\mathbb{R}^k)}.\tag{4.28}$$

Using (4.28) in (4.27) gives, after a kickback argument, the desired bound $\|Z-\hat{Z}^{h,\lambda}\|_{\mathcal{H}^2(\mathbb{R}^k)}\le Ch^\alpha$.

For $\|Y-\hat{Y}^{h,\lambda}\|_{\mathcal{S}^2(\mathbb{R}^d)}$, we use (4.25) and the standard arguments to get

$$\begin{aligned}&\Big(\mathbb{E}\big[\|Y_t-Y_t^{h,\lambda}\|^2\big]\Big)^{\frac{1}{2}}\\&\qquad\le|Y_0-Y_0^{h,\lambda}|+\left(\int_0^T\mathbb{E}\Big[\big\|f(s,X_s,Z_s)-f(\bar{s},\hat{X}_s^{h,\lambda},\hat{Z}_s^{h,\lambda})\big\|^2\Big]\mathrm{d}s\right)^{\frac{1}{2}}+\big\|Z_s-\hat{Z}_s^{h,\lambda}\big\|_{\mathcal{H}^2(\mathbb{R}^k)}.\end{aligned}$$

Applying Theorem 5, (4.26) and the obtained results for $\|X-\hat{X}^{h,\lambda}\|_{\mathcal{S}^2(\mathbb{R}^d)}$ and $\|Z-\hat{Z}^{h,\lambda}\|_{\mathcal{H}^2(\mathbb{R}^k)}$ complete the proof. □

Proposition 4. *Suppose the setting of Subsection 4.4.2, let Assumption 3 hold, $\alpha, \lambda > 0$. Then, there exists a constant C, independent of h, such that*

$$\begin{aligned}&\|X - \hat{X}^{h,\lambda}\|_{\mathcal{S}^2(\mathbb{R}^d)} + \|Y - \hat{Y}^{h,\lambda}\|_{\mathcal{S}^2(\mathbb{R}^d)} + \|Z - \hat{Z}^{h,\lambda}\|_{\mathcal{H}^2(\mathbb{R}^k)} \\ &\qquad \leq C\left(h^{\frac{\alpha}{2}} + \max_{0,\dots,N_h}\left(\mathbb{E}\Big[\|\zeta^*(t_n, X_n^{\lambda,h}) - \hat{\zeta}^*_{h,\lambda}(t_n, X_n^{\lambda,h})\|^2\Big]\right)^{\frac{1}{2}}\right).\end{aligned}$$

Proof. This is proved similarly to Theorem 7 without the kick-back argument and instead of (4.26), using

$$\begin{aligned}&\Big(\mathbb{E}\Big[\Big(\int_0^T (f(t, X_t, Z_t) - f(\bar{t}, \hat{X}_t^h, \hat{Z}_t^h))\mathrm{d}t\Big)^2\Big]\Big)^{\frac{1}{2}} \\ &\quad \leq C\Big(h^{\frac{\alpha}{2}} + \|X - \hat{X}^{h,\lambda}\|_{\mathcal{S}^2(\mathbb{R}^d)} + \max_{0,\dots,N_h}\left(\mathbb{E}\Big[\|\zeta^*(t_n, X_n^{\lambda,h}) - \hat{\zeta}^*_{h,\lambda}(t_n, X_n^{\lambda,h})\|^2\Big]\right)^{\frac{1}{2}}\Big).\end{aligned}$$

□

4.4.6 A discussion on the full error analysis of the robust deep FBSDE method

In Subsections 4.4.4 and 4.4.5, only the time discretization error is considered, i.e., the error between (4.8) and (4.15). For a full error analysis, the error between the fully implementable scheme (4.16) and (4.8) must be considered. Besides the time discretization error, there are three other sources of error: The first is the error induced by optimizing over the parameters of a neural network, instead of over the vast set $\mathcal{Z}_h$. By the Universal Approximation Theorem [111], this error can be made arbitrarily small, but this theorem gives no help with suggesting the network architecture that can guarantee a maximal error of desired size. The second error is the Monte-Carlo error induced from approximating the expectation in Y_0 by a sample mean. This error allows for a simple error analysis and the Monte Carlo error is of the order $\mathcal{O}(M_{\text{batch}}^{-1/2})$. The final error is the error induced from the inexact optimization procedure of (4.16).

4.5 Numerical experiments

In this section, we evaluate our algorithm on three different problems. The first two are of LQ type, for which we have access to a semi-analytic solution for comparison. The third example uses nonlinear terms, both in the drift and diffusion coefficients in the forward equation, and we no longer have access to a reference solution. In the first example, there is a one-to-one map between the feedback control and the $Z-$process, and we can set $\lambda = 0$ in the loss function. In the second and third examples, this is not the case, and $\lambda > 0$ is necessary for uniqueness of the minimizer to our discrete problem, and in turn convergence to the continuous FBSDE.

In the experimental convergence studies, we approximate $\|\cdot\|_{\mathcal{S}(\mathbb{R}^q)}$ and $\|\cdot\|_{\mathcal{H}(\mathbb{R}^q)}$, with

$$\|A\|_{\mathcal{S}^2_{h,M}(\mathbb{R}^q)} = \max_{n\in\{0,1,\dots,N\}} \left(\frac{1}{M}\sum_{m=1}^{M} \|A_n(m)\|^2\right)^{\frac{1}{2}},$$
$$\|A\|_{\mathcal{H}^2_{h,M}(\mathbb{R}^q)} = \frac{1}{N}\sum_{N=0}^{N-1} \left(\frac{1}{M}\sum_{m=1}^{M} \|A_n(m)\|^2\right)^{\frac{1}{2}}.$$

Here, $A(m) = \{A_1(m), A_2(m), \cdot, A_N(m)\}$, $m = 1, 2, \dots, M$, are *i.i.d.* realizations of some adapted stochastic processes A on the grid. The norm $\|\cdot\|_{L^2(\Omega;\mathbb{R}^q)}$ is approximated with a sample mean, denoted $\|\cdot\|_{L^2_{h,M}(\mathbb{R}^q)}$. For the convergence study, the Experimental Order of Convergence (EOC) is used. It is defined as

$$\mathrm{EOC}(h_i) = \frac{\log\left(\mathrm{error}(h_{i+1})\right) - \log\left(\mathrm{error}(h_i)\right)}{\log\left(h_{i+1}\right) - \log\left(h_i\right)}.$$

In all examples, we use the neural network architecture in Section 4.3.2. We use $M_{\text{train}} = 2^{22}$ training data points and batch size $M_{\text{batch}} = 2^9$ with $K_{\text{epoch}} = 15$ epochs. This gives $K_{\text{batch}} = 2^{12} = 4096$ updates per epoch. For the optimization, the Adam optimizer [119] is used with learning rate 0.1 for the first three epochs, which, after that, is multiplied by a factor of $\mathrm{e}^{-0.5}$ for each new epoch. For our use, it was important to choose M_{train} large, since in our empirical convergence results we want to isolate the time discretization error. In practice, the method generates acceptable solutions with significantly smaller M_{train}.

4.5.1 Linear quadratic control problems

Among all stochastic control problems, the LQ control problem is the most studied and that with the most structure, see *e.g.,* [120]. For our purposes, it has a closed-form analytic solution, with which we can compare our numerical approximations.

Let $k = d$, $x_0 \in \mathbb{R}^d$, $A, \sigma, \in \mathbb{R}^{d\times d}$, $R_x, G \in \mathbb{S}^d_+$, $R_u \in \mathbb{S}^\ell_+$ and $B \in \mathbb{R}^{d\times\ell}$ be of full rank and $C \in \mathbb{R}^d$. The state equation and cost functional of a linear-quadratic-Gaussian control problem are given by

$$\begin{cases} X_t = x_0 + \displaystyle\int_0^t (A(C - X_s) + Bu_s)\mathrm{d}s + \int_0^t \sigma \mathrm{d}W_s, \\ J^u(t,x) = \mathbb{E}^{t,x}\Big[\displaystyle\int_t^T (\langle R_x X_s, X_s\rangle + \langle R_u u_s, u_s\rangle)\mathrm{d}s + \langle GX_T, X_T\rangle\Big], \quad t \in [0,T]. \end{cases}$$

With the minimizer v^* of the corresponding Hamiltonian, $\inf_{u\in U}\{\langle \mathrm{D}_x V, Bu\rangle + \langle R_u u, u\rangle\}$, we have the optimal feedback control

$$u^*_t = -\frac{1}{2} R_u^{-1} B^T \mathrm{D}_x V(t, X_t). \tag{4.29}$$

Here, we recall that V is the solution to the associated HJB-equation. Its solution is given by

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

where (P, Q, R) solves the system of ordinary differential equations,

$$\begin{cases} \dot{P}(t) - A^T P(t) - P(t)A - P(t)BR_u^{-1}B^T P(t) + R_x = \mathbf{0}_{d\times d}, \\ \dot{Q}(t) + 2P(t)AC - A^T Q(t) - P(t)BR_u^{-1}B^T Q(t) = \mathbf{0}_d, \\ \dot{R}(t) + \mathrm{Tr}\{\sigma\sigma^T P(t)\} + Q(t)^T AC - \frac{1}{4}Q(t)^T BR_u^{-1}B^T Q(t) = 0, \quad t \in [0,T], \\ P(T) = G; \quad Q(T) = \mathbf{0}_d; \quad R(T) = 0. \end{cases}$$

The first equation is a matrix Riccati equation, and we refer to the whole system, slightly inaccurately, as the Riccati equation. The gradient of V satisfies $\mathrm{D}_x V(t,x) = 2P(t)x + Q(t)$. The related FBSDE reads:

$$\begin{cases} X_t = x_0 + \int_0^t \left[A(C - X_s) - \frac{1}{2}BR_u^{-1}B^\top Z_s\right]\mathrm{d}s + \int_0^t \sigma \mathrm{d}W_s, \\ Y_t = \langle GX_T, X_T\rangle - \int_t^T \left(\langle R_x X_s, X_s\rangle - \frac{1}{4}\langle R_u^{-1}B^\top Z_s, B^\top Z_s\rangle\right)\mathrm{d}s + \int_t^T \langle Z_s, \sigma \mathrm{d}W_s\rangle, \quad t \in [0,T]. \end{cases} \tag{4.30}$$

The solution to (4.30) is then given by

$$Y_t = X_t^T P(t) X_t + X_t^T Q(t) + R(t); \quad Z_t = 2P(t)X_t + Q(t). \tag{4.31}$$

The Riccati equation has an analytic solution in closed-form, only in one dimension. As benchmark solution in our experiments, we use the Euler approximation of the Riccati equation with 160×2^7 time steps and with 160 time steps for for X. The processes (Y, Z) are approximated by (4.31).

4.5.1.1 Example with state and control of the same dimension

Our first example concerns a two-dimensional LQ control problem with two-dimensional control. The matrices for the forward equation are given by

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

and the penalty matrices for the control problem by

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

In Figure 4.3, the approximation of (X, Y, Z) is compared to the analytic solution in mean, an empirical credible interval (again, defined as the area between the 5:th and 95:th percentiles at each time point) as well as for a single path. We see that the largest error comes from the approximation of Y. The reason for this is the error accumulation stemming from our time discretization. It is not due to the neural network approximation. This can be verified by using the baseline for Z, from the Riccati equation, and use an Euler-Maruyama scheme to generate the same error. This suggests that a more suitable choice of numerical schemes for Y should be used.

In Table 4.1, we see the convergence rates from the experiment. The regime of the LQ control problem, with, e.g., quadratic dependence in f does not satisfy the assumptions made in Section 4.4 and a direct comparison cannot be made. Still, we see, for instance, that Y_0^h converges empirically with order 1, while the error in the terminal condition reaches 0.69 and is likely to continue to decrease. In Theorem 6 and Corollary 4.4.1, there is a difference of a factor two between these two errors, which roughly appears to be in line with the rates obtained.

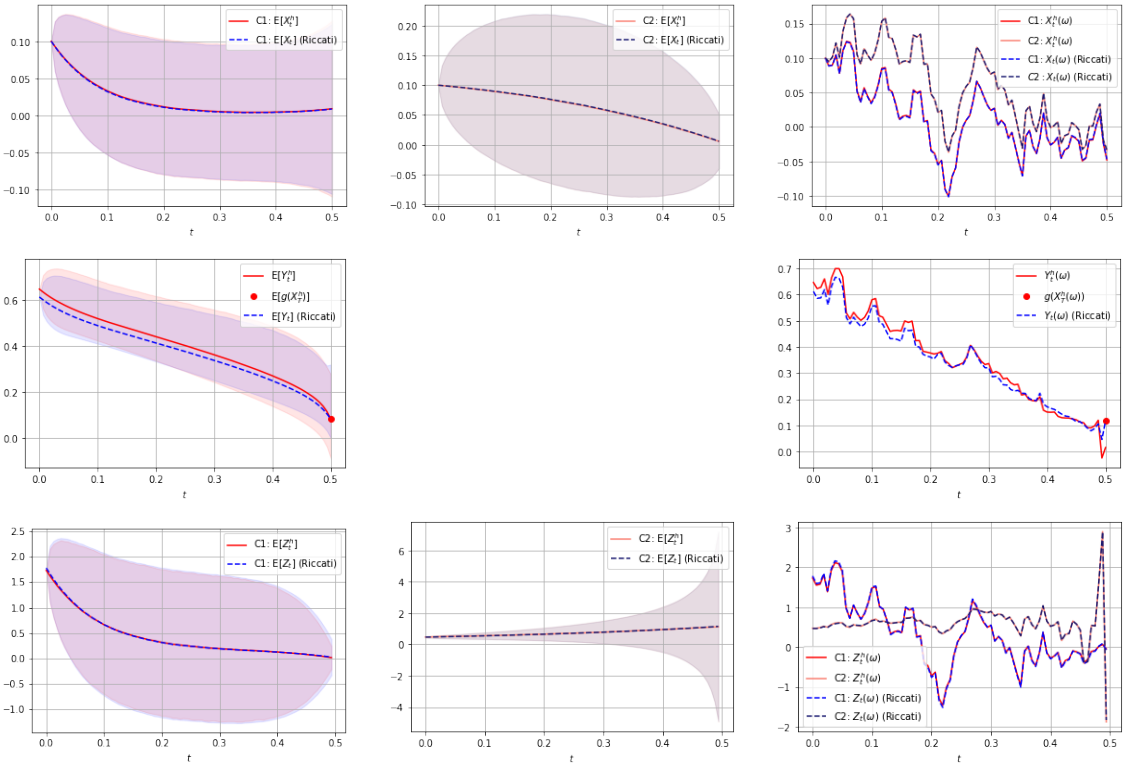

Figure 4.3: Average of solutions and a single solution path compared to their analytic counterparts for the LQ control problem from Section 4.5.1.1. The shaded areas represent empirical credible intervals, defined as the areas between the 5:th and the 95:th percentiles at each time point.

4.5.1.2 Example with control in lower dimensions than the state

Our second example concerns a six-dimensional problem with a two-dimensional control. The matrices used for the state equation are given by

$$A = \operatorname{diag}([1,2,3,1,2,3]), \quad B = \begin{pmatrix} 1 & -1 \\ 1 & 1 \\ 0.5 & 1 \\ 1 & -1 \\ 0 & -1 \\ 0 & 1 \end{pmatrix}, \quad C = \operatorname{diag}([-0.2,-0.1,0,0,0.1,0.2]),$$

$$\sigma = \operatorname{diag}([0.05,0.25,0.05,0.25,0.05,0.25]), \quad x_0 = (0.1,0.1,0.1,0.1,0.1,0.1)^\top, \quad T = 0.5.$$

	$\|X - X^h\|_{\mathcal{S}_h^2}$		$\|Y - Y^h\|_{\mathcal{S}_h^2}$		$\|Z - Z^h\|_{\mathcal{H}_h^2}$		$\|Y_N^h - g(X_N^h)\|_{L_h^2}$		$\|Y_0 - Y_0^h\|$		Y_0^h
N	Error	EOC	Error	EOC	Error	EOC	Error	EOC	Error	EOC	Value
Problem 1 with $d = 2$ and $\ell = 2$ with analytic initial value $Y_0 = 0.6122$											.
5	6.90e-2	1.28	1.22	1.03	5.66e-1	1.34	9.91e-1	0.98	7.10e-1	1.13	1.32
10	2.85e-2	1.11	6.02e-1	0.98	2.22e-1	1.16	5.04e-1	0.92	3.26e-1	1.16	0.937
20	1.32e-2	1.09	3.05e-1	0.86	1.00e-1	1.08	2.67e-1	0.80	1.47e-1	1.04	0.759
40	6.16e-3	1.00	1.68e-1	0.74	4.72e-2	0.98	1.53e-1	0.69	7.01e-2	1.01	0.683
80	3.07e-3		1.01e-1		2.39e-2		9.46e-3		3.47e-2		0.645
Problem 2 with $d = 6$ and $\ell = 2$ with analytic initial value $Y_0 = 1.4599$.											
5	7.25e-2	1.23	5.65e-1	0.90	1.20	0.90	4.43e-1	0.80	3.51e-1	1.10	1.80
10	3.10e-2	1.10	3.02e-1	0.76	7.41e-1	0.49	2.54e-1	0.69	1.63e-1	0.94	1.55
20	1.45e-2	0.87	1.79e-1	0.63	5.26e-1	0.28	1.57e-1	0.60	8.51e-2	0.82	1.55
40	7.96e-3	0.35	1.15e-1	0.53	4.34e-1	0.18	1.04e-1	0.54	4.81e-2	0.69	1.51
80	6.24e-3		7.96e-2		3.84e-1		7.15e-2		3.00e-2		1.49
Problem 3 with $d = 25$ and $\ell = 1$ with analytic initial value $Y_0 = 11.348$.											
5	2.25e-1	1.86	1.93	0.68	3.40	-	1.29	0.50	1.43	0.99	12.78
10	6.19e-2	0.97	1.21	0.42	2.54	-	9.15e-1	0.31	0.72	0.47	12.07
20	5.66e-2	0.57	9.00e-1	0.29	2.73	-	7.40e-1	0.19	0.52	0.53	11.87
40	2.48e-2	0.25	7.37e-1	0.069	2.72	-	6.47e-1	0.023	0.36	0.53	11.71
80	2.09e-2		7.03e-1		3.06		6.37e-1		0.25		11.60

Table 4.1: Errors and experimental order of convergence for LQ control problems described in Sections 4.5.1.1 and 4.5.1.2.

The penalty matrices of the control problem are given by

$$R_x = \mathrm{diag}([25, 1, 25, 1, 25, 1]), \quad R_u = \mathrm{diag}([1, 1]), \quad G = \mathrm{diag}([1, 25, 1, 25, 1, 25]).$$

Before we discuss our results, recall that the optimal feedback control at time t is given by $u_t^* = -\frac{1}{2}R_u^{-1}B^T Z_t$. Since u_t^* takes on values in $\mathbb{R}^\ell$ and $R_u^{-1}B^T$ is of rank $\ell < d$ at most, we can conclude that there exists infinitely many processes ζ_t, such that $u_t^* = -\frac{1}{2}R_u^{-1}B^T\zeta_t$. To obtain uniqueness of the control component, we set $\lambda = 1 > 0$.

In Figure 4.4, the approximations are compared with semi-analytic solutions in empirical mean, credible interval and for a representative path realization of X, Y and Z. Visually, the approximations capture (X, Y, Z) well. The convergence is shown in the middle part of Table 4.1 and we note that the experimental orders decrease below the orders of the previous example (top part of Table 4.1). To investigate whether this is the true convergence order, or if other errors are dominating for small time steps, we have done some hyperparameter optimization with different training data and batch sizes, learning rates and neural network architectures, without being able to improve these rates.

The third example aims to demonstrate our methods' ability to deal with high-dimensional problems. Most high-dimensional PDE and BSDE problems in the literature are symmetric (solutions are permutation invariant), and in some cases the solutions can be represented by a one-dimensional BSDE [121, Example 1]. From the parameters below, it becomes evident that the 25-dimensional problem that we choose is highly non-symmetric and therefore very challenging (arguably more challenging than a similar, but 100-dimensional symmetric problem). Non-symmetric problems in the literature are [103, 83, 102, 101, 86], and the dimensions are 4, 5, 3, 4, and 2, respectively. A symmetric problem in 100-dimensions is found in [83].

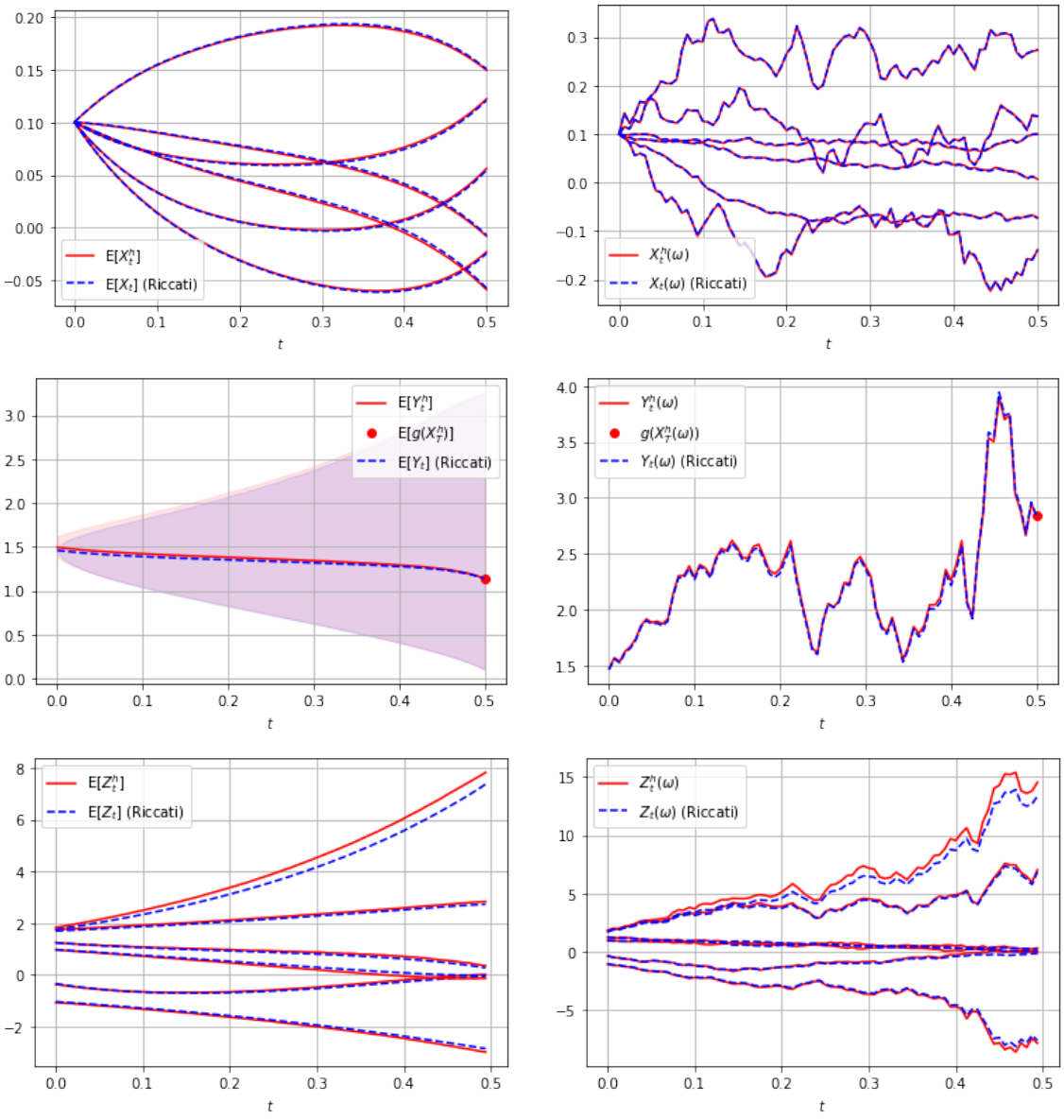

Figure 4.4: Average of solutions and a single solution path compared to their analytic counterparts for the LQ control problem from Section 4.5.1.2. The shaded area represents an empirical credible interval for Y, defined as the area between the 5:th and the 95:th percentiles at each time point. We do not include credible intervals for X and Z in this figure to facilitate visualization. For X and Z, we see one realization of each of the six components.

Despite the challenging nature of the problem and its relatively high-dimension, we achieve acceptable results, which is displayed in the bottom third of Table 4.1. It should however be pointed out that the error source induced by the time discretization is no longer dominating. This means that we do not see a convergence with the number of time steps for the approximation of the $Z-$process. All the other discretization errors decreases with the step size, but it is clear that we have other significant error sources. Figure 4.5 shows that visually the performance for 40 time steps is acceptable, even though some of the components of the Z process oscillates close to the terminal time. The phenomena of accurate X and Y processes and less accuracy in some of the components of the Z process could, at least heuristically, be explained by the mapping $\mathbb{R}^{25} \ni Z_t \mapsto u_t \in \mathbb{R}$. It is reasonable to assume that some of the components of the Z process are more influential in the above mentioned mapping, which is what we have seen empirically in our experiments. Moreover, we have noticed that the components of the Z process with the lowest magnitudes are less accurately approximated (relatively), which by the form of the feedback control, also justifies the above reasoning.

We use the following parameters:

$$
\begin{aligned}
&T = 0.5, \quad d = 25, \quad l = 1, \qquad x_0 = (0, 1, 0.1, \dots, 0.1), \quad A = \operatorname{diag}([1, 2, 3, \dots, 1, 2, 3, 1]),\\
&B = (1, 1, 0.5, 1, 0, 0, 1, 1, 0.5, 1, 0, 0, 1, 1, 0.5, 1, 0, 0, 1, 1, 0.5, 1, 0, 0, 1)\\
&C = (-0.2, -0.1, 0, 0, 0.1, 0.2, -0.2, -0.1, 0, 0, 0.1, 0.2, -0.2, -0.1, 0, 0, 0.1, 0.2, -0.2, -0.1, 0, 0,\\
&\qquad 0.1, 0.2, -0.2), \quad \sigma = \operatorname{diag}([0.15, 0.15, 0.25, 0.25, 0.25, 0.25, 0.25, 0.25, 0.25, 0.25, 0.25, 0.25,\\
&\qquad 0.15, 0.15, 0.25, 0.25, 0.25, 0.25, 0.25, 0.25, 0.25, 0.25, 0.25, 0.25]),
\end{aligned}
$$

and

$$
\begin{aligned}
&R_x = \operatorname{diag}([25, 1, 25, 1, 25, 1, 25, 1, 25, 1, 25, 1, 25, 1, 25, 1, 25, 1, 25, 1, 25, 1, 25, 1, 25]), \quad R_u = 1,\\
&G = \operatorname{diag}([25, 25, 25, 25, 25, 25, 1, 25, 1, 25, 1, 25, 25, 25, 25, 25, 25, 25, 25, 1, 25, 1, 25, 1, 25, 1]).
\end{aligned}
$$

4.5.2 Non-linear quadratic control problems

Finally, we consider a control problem with non-linear coefficients in the state equation and quadratic coefficients in the cost functional. It has stable and unstable equilibrium points at the odd and even integers, respectively. The problem has been chosen to mimic the unstable problems that are commonly considered in control, such as inverted pendulums.

Let $x_0 \in \mathbb{R}^d$, $A, \Sigma, \in \mathbb{R}^{d\times d}$, $R_x, G \in \mathbb{S}^d_+$, $R_u \in \mathbb{S}^\ell_+$ and $B \in \mathbb{R}^{d\times \ell}$ be of full rank and $C \in \mathbb{R}^d$. The state equation and cost functional of a non-linear quadratic control problem are given by

$$
\begin{cases}
X_t = x_0 + \int_0^t \left(A \sin(\pi C X_s) + B u_s\right) \mathrm{d}s + \int_0^t \Sigma(1_d + X_s X_s^\top) \mathrm{d}W_s, \\
J^u(t,x) = \mathbb{E}^{t,x}\Big[\int_t^T \left(\langle R_x X_s, X_s\rangle + \langle R_u u_s, u_s\rangle\right) \mathrm{d}s + \langle G X_T, X_T\rangle \Big].
\end{cases}
\tag{4.32}
$$

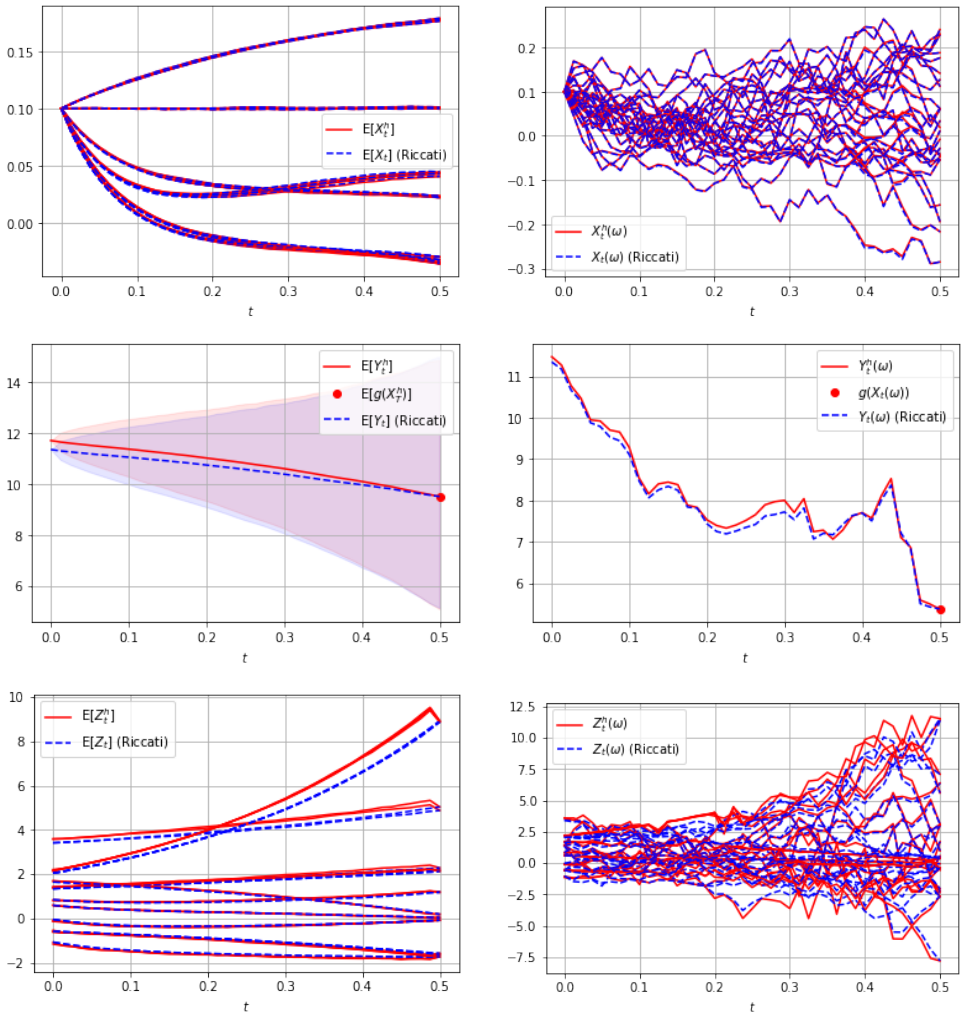

Figure 4.5: Average of solutions and a single solution path compared to their analytic counterparts for the second LQ control problem from Section 4.5.1.2, *i.e.*, the problem with $d = 25$ and $\ell = 1$. The shaded area represents an empirical credible interval for Y, defined as the area between the 5:th and the 95:th percentiles at each time point. We do not include credible intervals for X and Z in this figure to facilitate visualization. For X and Z, we see one realization of each of the 25 components.

	$\|Y_N^h - g(X_N^h)\|_{L^2(\Omega)}$		$\|Y_0 - Y_0^h\|$		Y_0^h
N	Error	EOC	Error	EOC	Value
5	2.69e-2	0.59	9.80e-3	1.01	0.2297
10	1.79e-2	0.53	4.85e-3	1.03	0.2241
20	1.24e-2	0.50	2.38e-3	0.99	0.2219
40	8.76e-3	0.49	1.20e-3	0.98	0.2207
80	6.27e-3		6.07e-4		0.2200

Table 4.2: Errors and experimental order of convergence for the nonlinear control problem in Section 4.5.2. A reference solution of $Y_0 = 0.2194$ is computed with the same method on a fine grid with $N = 160$ time points.

Due to the linear dependence of the control and the quadratic cost functional, the optimal feedback control is again given by

$$u_t^* = -\frac{1}{2} R_u^{-1} B^T \mathrm{D}_x V(t, X_t). \tag{4.33}$$

Similar to above, V is the solution to the associated HJB-equation. Again, by setting $Y_t = V(t, X_t)$ and $Z_t = \mathrm{D}_x V(t, X_t)$, we obtain the FBSDE

$$\begin{cases} X_t = x_0 + \int_0^t \left[A \sin(\pi C X_s) - \frac{1}{2} B R_u^{-1} R_u^{-1} B^\top Z_s\right] \mathrm{d}s + \int_0^t \Sigma(1_d + X_s X_s^\top) \mathrm{d}W_s, \\ Y_t = g(X_T) - \int_t^T \left(\langle R_x X_s, X_s \rangle - \frac{1}{4} \langle R_u^{-1} B^\top Z_s, B^\top Z_s \rangle\right) \mathrm{d}s + \int_0^t Z_s^\top \Sigma(1_d + X_s X_s^\top) \mathrm{d}W_s. \end{cases} \tag{4.34}$$

Particularly, we consider a three-dimensional problem with control in two dimensions, *i.e.,* $d = 3$ and $\ell = 2$ and use the following matrices for the state

$$A = \mathrm{diag}([1, 1, 1]), \quad B = \begin{pmatrix} 1 & 0 \\ 0 & 1 \\ 1 & 1 \end{pmatrix}, \quad C = \mathrm{diag}([1, 1, 1]),$$
$$\Sigma = \mathrm{diag}([0.1, 0.1, 0.1]), \quad x_0 = (0.1, 0.1, 0.1)^\top, \quad T = 0.25.$$

For the cost functional, we have the matrices

$$R_x = \mathrm{diag}([5, 1, 1]), \quad R_u = \mathrm{diag}([1, 1]), \quad G = \mathrm{diag}([1, 5, 1]).$$

Table 4.2 shows the experimental order of convergence of the terminal condition and the initial value of the BSDE. The factor two between them is again consistent with Theorem 6 and Corollary 4.4.1, even though the problem does not fall under the assumptions of these results.

5

D-TIPO: Deep time-inconsistent portfolio optimization with stocks and options

In this chapter, we propose a machine learning algorithm for time-inconsistent portfolio optimization. The proposed algorithm builds upon neural network-based trading schemes, in which the asset allocation at each time point is determined by a neural network. The loss function is given by an empirical version of the objective function of the portfolio optimization problem. Moreover, various trading constraints are naturally fulfilled by choosing appropriate activation functions in the output layers of the neural networks. Besides this, our main contribution is to add options to the portfolio of risky assets and a risk-free bond and using additional neural networks to determine the amount allocated into the options as well as their strike prices.

We consider objective functions more in line with the rational preference of an investor than the classical mean-variance, apply realistic trading constraints and model the assets with a correlated jump-diffusion SDE. With an incomplete market and a more involved objective function, we show that it is beneficial to add options to the portfolio. Moreover, it is shown that adding options leads to a more constant stock allocation with less demand for drastic re-allocations.

Keywords *- Portfolio optimization, time-inconsistent, neural networks, options*

This chapter is based on the paper with the same title, which is submitted for publication.

5.1 Introduction

Mean-variance (MV) portfolio optimization, originally proposed in [122], in 1955, has been a cornerstone of modern portfolio selection. The popularity for practitioners as well as researchers can on the one hand be explained by its simplicity with an intuitive trade-off between reward (mean) and risk (variance) and, on the other hand, by some delicate mathematical and philosophical properties. In the original version of MV optimization, the problem was static in the sense that the allocation between a set of assets and a riskfree bond was determined at $t = 0$ and held until terminal time T. Moreover, the assets were described by an arithmetic Brownian motion and, all together, the problem offered a closed-form allocation which was optimal with respect to the MV objective. Ever since, many extensions and generalizations have been proposed under the MV umbrella, such as static multi-period trading with discrete reallocation, see *e.g.,* [123, 124, 125, 126] and continuous trading, see *e.g.,*. [127, 128, 128, 129]. In most of the later applications, the arithmetic Brownian motion has been replaced by geometric Brownian motion or by more general processes describing the asset dynamics.

Time-inconsistency

Despite the straight forward formulation of the MV optimization, there are several difficulties making standard approximation tools from stochastic optimal control theory intractable. The main problem is that the MV objective function does not satisfy the law of iterated expectations and hence, the dynamic programming principle (DPP) is not satisfied. From an intuitive perspective, this means that a strategy which is optimal at time t is in general not optimal at $t + h$. From a practical perspective, this implies that the standard toolbox of computational methods for problems satisfying the DPP (i.e., optimization and approximations of conditional expectations backward in time) is not applicable. There are many attempts to circumvent this issue and they can mainly be categorised into two classes; Despite the straightforward formulation of the MV optimization, there are several difficulties making standard approximation tools from stochastic optimal control theory intractable. The main problem is that the MV objective function does not satisfy the law of iterated expectations and hence, the dynamic programming principle (DPP) is not satisfied. From an intuitive perspective, this means that a strategy which is optimal at time t is in general not optimal at $t + h$. These strategies are referred to as *pre-commitment* strategies, since an investor commits to the strategy at $t = 0$, and follows this strategy until $t = T$. From a practical perspective, this implies that the standard toolbox of computational methods for problems satisfying the DPP (i.e., optimization and approximations of conditional expectations backward in time) is not applicable. There are many attempts to circumvent this issue and they can mainly be categorised into two classes; *i*) *Induced time-consistent controls*, in which an embedding technique is used to transform the problem into an equivalent time-consistent problem. For example, consider the mean-variance optimzation problem with objective function

$$-\mathbb{E}[X_T] + \lambda \mathrm{Var}[X_T],$$

for some risk parameter $\lambda > 0$. This objective function can be re-written in the time-consistent form,

$$\mathbb{E}[\lambda X_T^2 - \mu X_T],$$

with $\mu = 1 + 2\lambda\mathbb{E}[X_T]$. If we treat μ as a constant, the optimization problem is time-consistent and can be solved by means of dynamic programming. The problem with this formulation is that μ depends on the expected value of the terminal wealth, which is a priori, unknown. On the other hand, we can solve the problem above for different values of μ and pick the unique μ (under certain concavity conditions), which satisfies $\mathbb{E}[X_T] = \frac{\mu-1}{2\lambda}$. For a detailed presentation, see [7]. Similar reformulations can be used for the mean-expected shortfall objective functions, see *e.g.,* [130].

An alternative is using *ii*) *Enforced time-consistent controls* in which the optimization is performed over the subset of time-consistent allocation strategies. This means that a constraint is added, which forces the control to be time-consistent. However, this constraint changes the problem formulation, and, as noted in [131, 132], the obtained strategy compares unfavorably with the pure pre-commitment strategy. Both methods have their own advantages and disadvantages and can both be motivated from different philosophical perspectives. For a more detailed discussion, we refer to [133, 134, 135].

The two classes above have in common that they transform the problem into a time-consistent problem, in which the DPP holds. In turn, this implies that the problem can be solved recursively, backward in time, one sub-problem at the time. By solving many sub-problems, classical methods to approximate conditional expectations can be used. Typically, one can resort to Monte–Carlo methods or approximation methods for PDEs or FBSDEs, see *e.g.,* [123, 129, 83, 87, 3] and [136, 137], respectively.

It should also be noted that there are certain ways to reformulate a time-inconsistent control problem either using a probabilistic approach, to obtain a McKean–Vlasov FBSDE or a PDE approach to obtain a so-called master equation, see *e.g.,* [8]. Even though these reformulations are mathematically possible, the resulting problems are inherently difficult to (numerically) solve, sometimes even harder than the original problem formulation.

In this chapter, instead of a problem specific embedding technique, we follow *e.g.,* [138, 135, 139, 140] and employ a machine learning algorithm to approximate the allocation strategy. Our approach is to take a step back and view the portfolio optimization problem as a stochastic control problem. After a discretization in time, the optimization problem can be approximated by representing the allocation strategy with a sequence of neural networks and letting the loss function be an empirical version of the objective function. This was originally proposed in [6], to approximate general stochastic control problems and in *e.g.,* [135, 139, 127], specifically for portfolio optimization. It should however be emphasized that these ideas were by no means new in a broader context, for instance, [141, 143, 144, 145, 146, 147, 148, 149, 150, 142], proposed machine learning methods for solving a wide variety of control problems. On the other hand, the specific application to finite horizon problems in which it is the control policy, and not the value function itself, should, to the best of our knowledge, largely be attributed to the authors in [6]. These algorithms have a clear advantage for time-inconsistent problems since the optimization is performed only once. Moreover, in contrast to the classical dynamic programming approach, the problem is solved forward in time, and as a consequence, the

algorithm does not rely on the DPP and hence, time-consistent and time-inconsistent problems are treated similarly.

Adding options and gain flexibility

Due to the ambiguity of measuring risk with variance of the terminal wealth, many alternative ways to measure risk have been proposed. As the most common alternatives, we mention the semi-variance and expected shortfall, which are explored in *e.g.,* [151, 152, 153, 154]. As stated above, it is clear that the MV objective function is not completely in line with the rational preference of an investor, since it treats downside risk and upside potential equivalently. On the other hand, it is not clear whether optimizing with respect to another objective function will result in strategies that are more in line with the investors' preferences. As an example, assume that the asset returns are normally distributed, then the MV objective is, objectively, the best objective function since the law of the normal distribution is completely determined by its mean and variance. The question is then whether or not asset returns are normally distributed, and the answer is, in general, a clear "no". On the other hand, asset returns are usually close to symmetric and a symmetric distribution with reasonable tails can be relatively well approximated by a normal distribution. Therefore, it is not clear that we can create a terminal wealth with a distribution that is flexible enough to benefit from the more complex objective functions. Fortunately, there are several financial products with asymmetric returns, which can be added to a portfolio. For instance, a plain European option has a highly asymmetric return and is a good candidate to add to the portfolio.

Another interesting aspect of adding options to the portfolio is the fact that many pension funds are prohibited from using leverage in their portfolios. This can be circumvented by using options, which offer a similar structure as a leveraged position, however, without the downside risk. This comes at the price of the option premium.

Although a straight forward extension to a stock and bond portfolio, portfolio optimization with equity options has not been a major topic in the scientific literature. Some attempts are found in [155, 156, 157, 158, 159], but the focus was mainly on static portfolios consisting of only options. The focus in this chapter is, instead, to combine the two approaches and trade in both the assets themselves and European options written on the same assets, and have a machine learning algorithm determine the optimal allocation. We name the proposed algorithm *Deep Time-Inconsistent Portfolio optimization with stocks and Options*, with acronym "D-TIPO". When the options are not included in the allocation learning, we use the abbreviation "D-TIP".

Structure of the chapter

This chapter is organized as follows. In Section 5.2, the framework is introduced. We outline trading constraints, objective functions and aim to provide some heuristic motivation for the objective functions used. In Section 5.3, the methodology including discretization of the continuous optimization problem, as well as the neural networks used, are outlined. Moreover, we explain how the trading constraints can be built into the neural network structure and provide pseudo-code for the full algorithm. Section 5.4 contains numerical experiments in which we first confirm the accuracy of the algorithm for an example with a reference solution and

then the algorithm is validated and compared with some other allocation strategies. Finally, in section 5.5, we conclude the findings of the chapter.

5.2 Problem formulation

In this chapter, we take the perspective of a *trader*, who is allowed to trade in one risk-free bond, in $N^{\text{stocks}} \in \mathbb{N}$ stocks, and $N^{\text{options}} \in \mathbb{N}$ options. We let $S = (S_t)_{t\in[0,T]}$ be an $\mathbb{R}^{N^{\text{stocks}}}-$valued time-continuous Markov process on a complete probability space $(\Omega, \mathcal{F}, \mathbb{A})$. We consider a *trading period* $\mathcal{T} := [0, T]$, where $T \in \mathbb{R}_+$ is referred to as the *terminal time*. The outcome set Ω is the set of all possible realizations of the stochastic economy, $\mathcal{F}$ is a $\sigma-$algebra on Ω and we define $\mathcal{F}_t$ as the sub-σ-algebra generated by $(S_s)_{s\in[0,t]}$. The probability measure $\mathbb{A}$ is a generic notation, representing either the real world measure, or the risk-neutral measure, denoted $\mathbb{P}$ and $\mathbb{Q}$, respectively. The bond is denoted by $B = (B_t)_{t\in[0,T]}$ and for $i \in \{1, 2, \ldots, N^{\text{options}}\}$, an option with S as underlying asset (could be a single stock or a basket of stocks), at time $t \in [0, T]$ and terminating at T is denoted by $V^i(t, S_t\,; K)$, respectively, where $K \in \mathbb{R}$ is the strike price. Without loss of generality, we set the initial values of all stocks, options and the bond to unity at $t = 0$, *i.e.,* for $j \in \{1, 2, \ldots, N^{\text{stocks}}\}$ and $i \in \{1, 2, \ldots, N^{\text{options}}\}$, we set $S_0^j = 1$, $V^i(0, S_0) = 1$ and $B_0 = 1$.

The trader is allowed to trade the stocks and the bond at a set of *trading dates*, denoted by $\mathcal{T} \subseteq [0, T]$. If $\mathcal{T} = [0, T]$, then trading is allowed continuously in the entire trading period but a more realistic scenario is that trading is allowed only on a discrete set of dates within the trading period. The options, on the other hand, can only be traded at *initial time* of the trading period, *i.e.,* at time 0 and are held until the terminal time. It should however be pointed out that our framework would also allow for trading in the options during the entire time period $\mathcal{T}$, but since options are less liquid and transaction costs usually are significantly higher than for stocks, we restrict ourselves to trade options only at $t = 0$.

Denote by $\alpha^k = (\alpha_t^k)_{t\in[0,T]}$ the piecewise constant process describing the amount the trader holds in stock k and for $k = 0$, the amount the trader holds in the bond. We can describe the total wealth of the portfolio stemming from the stock and the bond holdings at time t by

$$x_t = \alpha_t^0 B_t + \sum_{k=1}^{N^{\text{stocks}}} \alpha_t^k S_t^k. \tag{5.1}$$

Since the portfolio is self-financing,

$$\alpha_t^0 = \frac{1}{B_{\tau(t)}}\left(x_{\tau(t)} - \sum_{k=1}^{N^{\text{stocks}}} \alpha_{\tau(t)}^k S_{\tau(t)}^k\right), \tag{5.2}$$

where $\tau(t) = \max_s\{s \in \mathcal{T} \mid s \le t\}$, *i.e.,* the most recent trading date. Sometimes, it is convenient to work with the total amount of cash.We therefore denote by $A^k = (A_t^k)_{t\in[0,T]}$, the amount of cash invested in asset k and define for $k > 0$, $A_t^k = \alpha_t^k S_t^k$ and for $k = 0$, $A_t^0 = \alpha_t^0 B_t^0$. The return on investment from trading in the stocks and the bond is then given by

$$R_{\text{SB}}(S; \alpha) = x_T - x_0. \tag{5.3}$$

Let β^i denote the amount of option i in the portfolio. The total amount invested in the options at time t is then given by

$$y_t = \sum_{i=1}^{N^{\text{options}}} \beta^i V^i(t, S_t; K^i),$$

where K^i is the strike price for option i. We can define the return on the investment from the static option position by

$$R_{\text{O}}(S; \beta) = y_T - y_0. \tag{5.4}$$

Summing up (5.3) and (5.4), we obtain the total return

$$R(S; \alpha, \beta) = R_{\text{SB}}(S; \alpha) + R_{\text{O}}(S; \beta). \tag{5.5}$$

In order to evaluate the satisfaction of the investor with the return, we use an objective function. A good objective function should be able to numerically represent the investor's view of risk. In this section, the only restriction we put on the objective function is that it is a deterministic function, taking the trading strategies α and β as inputs. The objective function is of the form

$$U(\alpha, \beta) = u\big(\mathcal{L}[R(S; \alpha, \beta)]\big), \tag{5.6}$$

where $\mathcal{L}(\cdot)$ denotes the probabilistic law. Our objective is then to find a trading strategy α, β, such that the objective function is maximized. Note that we do not specify the trading strategies that are allowed. Moreover, we assume that the maximum of the objective function above is attainable.

Example 5.2.1. *Suppose we consider a portfolio consisting of only stocks, modeled with a multi-dimensional Geometric Brownian Motion and a bond with deterministic interest rate and that continuous trading on $[0, T]$ is allowed. If the aim is to maximize the expectation and minimize the variance of $R_{\text{SB}}(S; \alpha)$, i.e., using an objective function of the form $U^{\text{DMVO}}(\alpha; \lambda) = \mathbb{E}[R_{\text{SB}}(S; \alpha)] - \lambda Var[R_{\text{SB}}(S; \alpha)]$ ($\lambda \in \mathbb{R}_+$ is a risk parameter), we are in the classical* Dynamic Mean-Variance Optimization *framework. It is well-known, see e.g., [7], that there is an analytic strategy α^* such that*

$$U^{\text{DMVO}}(\alpha^*; \lambda) = \text{Max}_\alpha U^{\text{DMVO}}(\alpha; \lambda).$$

Now, if we would also allow for trading in vanilla call and put options on each of the underlying stocks, this would not increase the utility, since in the Black–Scholes framework, the options would be fully replicable by the stocks and the bond. This implies that any strategy involving stocks, the bond and options can be replicated by a strategy only involving stocks and the bond.

The example above indicates that adding options to a specific portfolio does not increase utility for the trader, but is that statement true in general? The answer is that if we are able to trade continuously in a complete market, then the European option is replicable, and hence it is not possible to increase the utility by adding options to the portfolio. On the other hand, if the market is incomplete, or when we are not able to trade continuously, then there is a possibility that adding options to a portfolio can increase the utility. It is a well-known fact that the real world financial market is incomplete, and in the subsections below, we introduce

some aspects which make the model of the market incomplete. It should however be pointed out that in general, a Lévy process modelling the stocks, (*e.g.,* Geometric Brownian Motion with jumps) would lead to an incomplete market, even without adding any other market frictions.

5.2.1 Market frictions

In this section, we introduce the market frictions and trading constraints considered in this chapter. Below, we extend the framework to allow for transaction costs, non-bankruptcy constraint and leverage constraints for the trading strategies α and β.

From now on, unless otherwise is stated, we assume that $\mathcal{T}$ is a set of finite trading dates[1], *i.e.,* $\mathcal{T} = \{t_0, t_1, \ldots, t_{N-1}\}$, such that $t_0 = 0$ and $t_i < t_{i+1}$. We can now rewrite the value of the stock and the bond, given in (5.1), in terms the initial value plus the accumulated increments between the finite trading dates, i.e.,

$$x_{t_{n+1}} = x_{t_n} + \alpha^0_{t_n}(B_{t_{n+1}} - B_{t_n}) + \sum_{k=1}^{N^{\text{stocks}}} \alpha^k_{t_n}(S^k_{t_{n+1}} - S^k_{t_n}). \tag{5.7}$$

Transaction costs: Assuming a proportional transaction cost we can write the sum of the transaction costs for stock k as

$$\text{TC}^k = \sum_{n=1}^{N} C\mathrm{e}^{r(T-t_n)}|\alpha^k_{t_n} - \alpha^k_{t_{n-1}}|S^k_{t_n}. \tag{5.8}$$

Here $100 \times C \in \mathbb{R}_+$ represents the transaction costs as a percentage of the size of the transaction. Finally, the total transaction cost is given by

$$\text{TC} = \sum_{k=1}^{N^{\text{stocks}}} \text{TC}^k.$$

In fact, the implication of the above is that we do not pay transaction costs immediately, but instead at the end of the trading period, with appropriate interest rate.

Non-bankruptcy constraint: The non-bankruptcy constraint is formulated in a way such that the first time the sum of the values of the stocks and the bond is non-positive, the portfolio is liquidated. This implies that, in the presence of a non-bankruptcy constraint, (5.7) is replaced by

$$x_{t_{n+1}} = x_{t_n} + \mathbb{I}_{\{x>0\}}(x_{t_n})\left(\alpha^0_{t_n}(B_{t_{n+1}} - B_{t_n}) + \sum_{k=1}^{N^{\text{stocks}}} \alpha^k_{t_n}(S^k_{t_{n+1}} - S^k_{t_n})\right), \tag{5.9}$$

where $\mathbb{I}_{\{x>0\}}(\cdot)$ is the indicator function.

[1] Or a discrete approximation of continuous trading. In practice, all trading is carried out in a time discrete fashion, but in some cases a closed-form optimal strategy can be achieved in the continuous case, which is why we allow for such trading in our framework.

The above implies that a liquidation of x caused by a bankruptcy can only occur at a trading date. In reality, it would be reasonable to liquidate the portfolio as soon as the value touches zero. On the other hand, immediate liquidation may not be possible, due to low liquidity. It should also be noted that the underlying methodology does not rely on this specific modelling choice.

Trading constraint: There are several different trading constraints and below, we mention a few of them.

- **No short-selling constraint** - No short-selling of the stocks, implying that for $t \in [0, T]$ and $1 \geq k \geq N^{\text{stocks}}$, $\alpha_t^k \geq 0$.
- **No leverage constraint** - Implying that we cannot short sell the bond, *i.e.,* for $t \in [0, T]$, $\alpha_t^0 \geq 0$.
- **No bankruptcy constraint** - If $x_{t_n} \leq 0$, then all positions are liquidated and for $t \geq t_n$, $x_t = x_{t_n}$.
- **No short selling of the options**, which is equivalent to ensuring that for $1 \geq i \geq N^{\text{options}}$ $\beta^i \geq 0$.
- **Positivity of the bond and the stocks part of the portfolio** - If $y_0 \leq x_0^{\text{IC}}$ is ensured, then from the definition $x_0 \geq 0$.

The aim in this chapter is to apply a neural network-based strategy to approximate the allocations at each trading date. The constraints described above are built into the neural networks, which is specified in full detail in sections below.

5.2.2 Objective functions

There are several different ways of measuring the performance of a trading strategy. What all measures have in common is that they attempt to numerically represent the risk aversion of the trader so that a strategy optimal for the trader's preferences can be adopted. Below, we introduce a few different objective functions and present their individual motivations.

Utility functions: A utility function measures the so-called marginal happiness of wealth, *i.e.,* how happy is the investor with one extra unit of wealth, given the current level of wealth. These functions are typically concave since one extra unit adds more happiness when the investor's wealth is low than when it is high. The objective function used is usually the expected utility, which leads to a mathematically nice framework from a modeling perspective since the problem becomes time-consistent and, hence, the dynamic programming principle holds. A disadvantage with these types of objective functions is that they are not particularly intuitive and it becomes therefore difficult for managers to determine the risk parameters. Another disadvantage is that the expected utility is a narrow measure of the utility and does not say anything about the variability of the utility outcomes.

Mean-Variance objective: Under the assumption that asset returns are normally distributed,

	Volatility strategy	Equities
Mean	9.9%	9.7%
Standard deviation	15.2%	15.1%
Skewness	-8.3	-0.6
Kurtosis	104.4	4.0

Table 5.1: Comparison of returns of a volatility strategy and a pure equity strategy, in terms of mean, standard deviation, skewness and kurtosis. If returns would have been normally distributed, the skewness and the kurtosis would have been 0 and 3, respectively.

the mean-variance objective function is the optimal choice, since the normal distribution is completely determined by its mean and variance. It is however a well-known fact that asset returns are not normally distributed, but have a slightly fatter tail than the one of a normal distribution. This phenomenon is even more pronounced for volatility strategies (strategies implemented in the derivatives market using options and other financial products), see *e.g.,* [160, page 40], in which these two strategies are compared in terms of their first four moments. It is shown that while the first two moments are almost identical, the third and fourth moments differ significantly (the third and fourth moments for asset returns are closer to being normal while these moments for the returns of a volatility strategy exhibit significant leptokurtic behaviour). The numbers are displayed in Table 5.1. It should be emphasized that by using the mean-variance objective function, we do not inherently assume normally distributed returns but optimality in terms of the trading strategy related to a mean-variance objective function could be ambiguous as can be demonstrated with the example above. Since the mean values (9.9% and 9.7%) and the variances (15.2% and 15.1%) are similar in a mean-variance objective function, one could conclude that the strategies are equally good, even though the distributions of the returns are very different. The question is then which of the two strategies a trader should apply? If the trader manages a pension fund, it is reasonable to believe that a high negative skewness and a large kurtosis are highly undesirable, while for a hedge fund with short time horizon the reasoning could be the other way around. The challenge for us is then to construct an objective function which better represents the true objective compared to the mean-variance objective.

As a final note on when it makes sense to consider another objective, we claim that when the returns deviate from normality and in particular if the distribution is asymmetric, it is natural to ask why a trader would want to penalize not only downside risk but also the upside potential, which is the direct consequence of using the variance as a measure of risk.

Non-symmetric objectives: Since we deal with trading strategies that are able to create highly non-symmetrical distributions of the terminal wealth, we want to explore objective functions suitable for this. The first important aspect is to model downside risk and upside potential differently. One way to assess this problem is to use *expected shortfall*, which is defined as the mean of the tail of a distribution. For example, if we want to penalize downside risk, we may try to maximize the average of the 10% worst outcomes or if we want to encourage upside potential, we could try to maximize the average of the 10% best outcomes.

The expected shortfall can be defined using the *Value at Risk* in the following way

$$\begin{aligned}\mathrm{VaR}_p(R) &= \inf\{P \in \mathbb{R} \,|\, \mathbb{A}(R \leq P) \geq p\},\\ \mathrm{ES}_p^-(R) &= \mathbb{E}[R \,|\, R \leq \mathrm{VaR}_p(R)], \quad \mathrm{ES}_p^+(R) = \mathbb{E}[R \,|\, R \geq \mathrm{VaR}_p(R)].\end{aligned}$$

For notational convenience, we omit the dependency on the trading strategy in the notation below. A typical objective function would then be

$$U = u(\mathcal{L}(S)) = \mathbb{E}[R] - \lambda_1 \mathrm{Var}[R] + \lambda_2 \mathrm{ES}_{p_1}^-(R) + \lambda_3 \mathrm{ES}_{p_2}^+(R), \tag{5.10}$$

where $\lambda_1, \lambda_2, \lambda_3 \in \mathbb{R}_+$ are parameters describing the risk preference and $p_1, p_2 \in (0, 1)$ are the parameters controlling the sizes of the left and right tails we want to address. For instance, one could choose $p_1 = 0.1$ and $p_2 = 0.9$ to control the upper and lower tails (10% on each side). In the formulation above, we view R as a generic return of some strategy, however, in our case U and R would depend on the trading strategy and R would in addition also depend on a random realization of the stochastic economy, as in (5.5) and (5.6). In Figure 5.1, the objective function given in (5.10) is illustrated for a toy example with a returns following a skew-normal distribution.

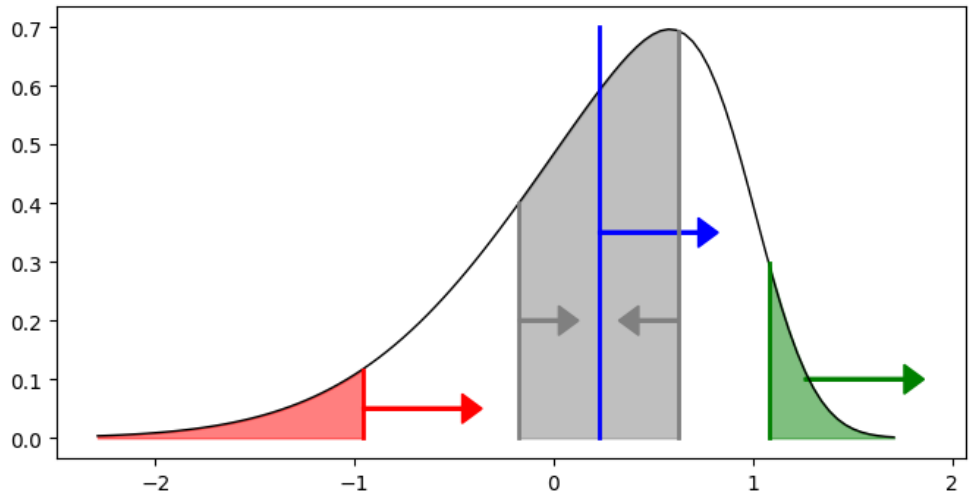

Figure 5.1: Example of a probability density function for the terminal wealth and the terms in (5.10) are illustrated. Red, blue and green represent the lower expected shortfall, mean and higher expected shortfall, respectively, and the arrows represent the preferred direction. The gray area represents the mean plus/minus the variance and the arrows represent the preference to decrease variance.

5.2.3 Full optimization problem

The aim in the previous sections was to explain and motivate each component of the problem statement. Here, we summarize the above and formulate the optimization problem.

In the sections above, we referred to α and β as the trading strategies. From here on, we also consider the set of strike prices, $K = (K^1, K^2, \ldots, K^{\text{options}})$, as part of the trading strategy which is denoted by $\pi = (\alpha, \beta, K)$. The following example aims to motivate why we also include strike prices.

Example 5.2.2 (Bull-call-spread strategy)**.** *Consider a situation where a trader believes that a stock will either increase in value, or drastically decrease, for instance, when the company is about to report. Then, a so-called bull-call-spread can be used. This is the position where the trader both buys and sells a European call option but with different strike prices. If executed correctly, this can create a situation where the trader only risks to loose the difference in the premiums between the bought and sold option with an upside potential proportional to the stock value but bounded from above. This strategy is arguably best explained with a specific example and illustrative figures. Therefore, we consider a stock which today, at* $t = 0$*, has value* $S_0 = 1.0$*. Moreover, we have two European call options with values at maturity* T*,* $V^1(T, S_T; K^1) = max(S_T - K^1, 0)$ *and* $V^2(T, S_T; K^2) = max(S_T - K^2, 0)$*, respectively. We set* $K^1 = 1.1$ *and* $K^2 = 1.3$ *and in Figure 5.2 (left-side), we display the return, at maturity* T*, for the stock itself, buying one unit of option 1 and selling one unit of option 2, respectively. In the right-side figure, we compare returns of three different strategies and clearly the return is dependent not only on how much of each of the three products the trader invests in, but also on the strike prices* K^1 *and* K^2*.*

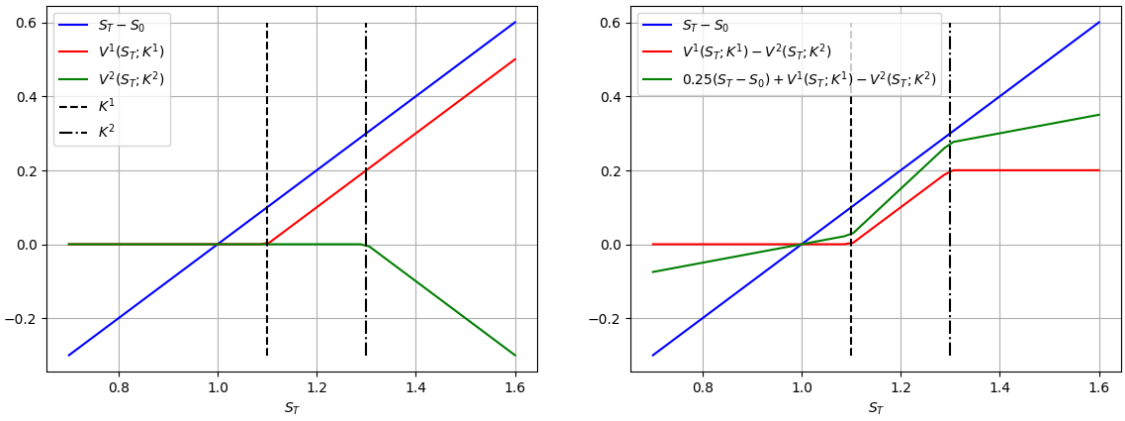

Figure 5.2: Returns plotted against the stock value at the terminal time T. **Left:** Returns for investing in a stock, buying one unit of option 1 and selling one unit of option 2, respectively. **Right:** Returns for three different combinations of the products, where the red line is the classical bull-call spread.

As becomes clear in the example above, it is not easy to have an intuitive overview on which strike prices to choose, especially not when a complex objective function is used. Therefore, we consider also the strike prices to be part of the optimization problem.

We assume an objective function $U(\pi) = u(\mathcal{L}[R(S;\pi)])$, an initial wealth $x_0^{\text{IC}} \in \mathbb{R}_+$ (usually set to 1) and we denote by Π the set of *admissible* (allowed) trading strategies, *i.e.* the set of executable strategies taking all the trading constraints into account. All equations below are collected from the sections above, but the dependency on the trading strategy π is specified

more carefully to make the optimization problem more clear.

$$\begin{cases} \underset{\pi\in\Pi}{\text{maximize}} = U(\pi) = u\big(\mathcal{L}(R(S\,;\pi))\big), \quad \text{where,} \\ R(S\,;\pi) = R_{\text{SB}}(S\,;\pi) + R_{\text{O}}(S\,;\pi) \\ R_{\text{SB}}(S\,;\pi) = x_T(S;\pi) - x_0(\pi) - \sum_{k=1}^{N^{\text{stocks}}} \text{TC}^k, \quad R_{\text{O}}(S\,;\pi) = y_T(S\,;\pi) - y_0(\pi), \\ x_T(S;\pi) = x_0 + \sum_{n=0}^{N} \mathbb{I}_{\{x>0\}}\big(x_{t_n}(S;\pi)\big)\big[\alpha^0_{t_n}(B_{t_{n+1}} - B_{t_n}) + \sum_{k=1}^{N^{\text{stocks}}} \alpha^k_{t_n}(S^k_{t_{n+1}} - S^k_{t_n})\big], \\ \quad x_0 = x_0^{\text{IC}} - y_0(\pi),\ y_T(S\,;\pi) = \sum_{i=1}^{N^{\text{options}}} \beta^i V^i(T, S_T\,;\, K^i), \quad y_0(\pi) = \sum_{i=1}^{N^{\text{options}}} \beta^i V^i(0, S_0\,;\, K^i). \end{cases} \tag{5.11}$$

This optimization problem admits to closed-form strategies α and β (for fixed K) only in rare special cases and we therefore have to rely on numerical approximations. When it is required, we use a time discretization scheme for S and we use empirical distributions as approximations for $\mathcal{L}[R(S;\pi)]$ in a Monte–Carlo fashion. The discrete counterpart (in time and/or in probability space) of (5.11) is then approximated by letting deep neural networks represent the trading strategies and optimizing with a gradient decent type algorithm. This is explained in more detail in sections below.

5.3 Methodology

As briefly mentioned in the previous section, there are two entities that often need to be discretized, namely the asset price process and the objective function. For the asset process, we need to resort to a discretization scheme when there are no closed-form expressions available. The objective function usually not only depends on the returns but also on the probability distribution of the returns. Since this is unknown, despite for rare special cases, we need an approximation. This is simply done by using empirical distributions as approximations.

5.3.1 Discretized asset process and empirical distribution

Let $\mathcal{T}^N = \{t_0, t_1, \ldots, t_{N-1}\}$ be the set of distinct trading dates, where either, $\mathcal{T}^N = \mathcal{T}$, or, if $\mathcal{T}$ is an infinite set (continuous trading), $\mathcal{T}^N$ is a discrete approximation of $\mathcal{T}$. We let $t_N = T$ and for $0 \leq i \leq N-1$, $t_i < t_{i+1}$. We assume that we are able to generate $M \in \mathbb{N}_+$ samples of the $N^{\text{stocks}}-$dimensional asset process S. We denote asset k, realization m, at time $t_n \in \mathcal{T}_N$ by $S^k_{t_n}(m)$, realization m at time $t_n \in \mathcal{T}_N$ by $S_{t_n}(m) = \{S^1_{t_n}(m), \ldots S^{N^{\text{stocks}}}_{t_n}(m)\}$ and realization m by $S(m) = \{S_{t_0}(m), \ldots S_{t_N}(m)\}$.

Example 5.3.1 (A one-dimensional Geometric Brownian motion)**.** *Given the initial value S_0 and assuming we have access to the drift and diffusion parameters μ and σ, a closed-form solution for the asset process is given by*

$$S_t = S_0 e^{(\mu - \frac{\sigma^2}{2})t + \sigma(W_t - W_0)}.$$

A model like this can easily be sampled from for any $t \in \mathcal{T}^N$ and by the Markovianity of S_t no information is lost by only sampling discrete values.

Example 5.3.2 (A general jump-diffusion SDE)**.** *Given the initial value S_0 and assuming we have access to the drift, diffusion and jump coefficients μ, σ and J, we employ,*

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

where X_t represents a jump process which we do not further specify in this example. In general, we do not have a closed-form solution for S_t and therefore we need to rely on a discrete approximation scheme such as the Euler–Maruyama scheme. Similar to the Geometric Brownian motion, the jump-diffusion SDE above is Markovian, but one needs to be careful with the time discretization in order to ensure convergence.

We denote by $\bar{U}^M(\pi)$ the objective function when the distributions of the returns are approximated by an empirical counterpart with M samples and by $\mathcal{L}^M[\,\cdot\,]$ the empirical probability law.

5.3.2 Neural network approximation

In this section, we describe how the trading strategy π is represented by a sequence of neural networks and simple parameters. Optimality in terms of the objective function is then sought with a gradient descent type algorithm. We sometimes use the machine learning terminology *loss function* and *training* for the objective function and optimization procedure, respectively.

5.3.2.1 General notation for a neural network

Below, we introduce some machine learning notation, which is subsequently put into the context of this chapter. A neural network, which is nothing but a parametrized mapping, is here denoted by $\phi(\,\cdot\,;\theta^{\mathrm{NN}})\colon \mathbb{R}^{\mathfrak{D}^{\mathrm{input}}} \to \mathbb{R}^{\mathfrak{D}^{\mathrm{output}}}$, where $\mathfrak{D}^{\mathrm{input}}$ and $\mathfrak{D}^{\mathrm{output}}$ are the input and output dimensions, respectively. Here, θ^{NN} is a parameter containing all the trainable (optimizeable) parameters of the network.

- Denote the number of layers (input and output layers included) by $\mathfrak{L} \in \mathbb{N}$, and for layer $\ell \in \{1, 2, \ldots, \mathfrak{L}\}$, the number of nodes by $\mathfrak{N}_\ell \in \mathbb{N}$. Note that $\mathfrak{N}_1 = \mathfrak{D}^{\mathrm{input}}$;
- For layer $\ell \in \{2, 3, \ldots, \mathfrak{L}\}$, we denote the weight matrix, acting between layers $\ell - 1$ and ℓ, by $w_\ell \in \mathbb{R}^{\mathfrak{N}_{\ell-1} \times \mathfrak{N}_\ell}$, and the bias vector by $b_\ell \in \mathbb{R}^\ell$;
- For layer $\ell \in \{2, 3, \ldots, \mathfrak{L}\}$, we denote the (scalar) activation function by $a_\ell\colon \mathbb{R} \to \mathbb{R}$ and the vector activation function by $\boldsymbol{a}_\ell\colon \mathbb{R}^{\mathfrak{N}_\ell} \to \mathbb{R}^{\mathfrak{N}_\ell}$, which, for $x = (x_1, x_2, \ldots, x_{\mathfrak{N}_\ell})$, is defined by
$$\boldsymbol{a}_\ell(x) = \begin{pmatrix} a_\ell(x_1) \\ \vdots \\ a_\ell(x_{\mathfrak{N}_\ell}) \end{pmatrix};$$
- The output of the network should obey the trading constraints and this is managed by choosing an appropriate activation function in the output layer.

The neural network is then defined by

$$\phi(\cdot\,;\theta) = L_{\mathfrak{L}} \circ L_{\mathfrak{L}-1} \circ \cdots \circ L_1(\cdot), \tag{5.12}$$

where for $x \in \mathbb{R}^{\mathfrak{L}_{\ell-1}}$, the layers are defined as

$$L_\ell(x) = \begin{cases} x, & \text{for } \ell = 1, \\ \boldsymbol{a}_\ell(w_\ell^\top x + b_\ell), & \text{for } \ell \geq 2, \end{cases}$$

with $w_\ell^\top$ the matrix transpose of w_ℓ. The trainable parameters are then given by the list

$$\theta^{\text{NN}} = \{w_2, b_2, w_3, b_3, \ldots, w_{\mathfrak{L}}, b_{\mathfrak{L}}\}.$$

Finally, denote by $\mathfrak{D}^{\theta^{\text{NN}}}$ the number of trainable parameters, *i.e.,* $\mathfrak{D}^{\theta^{\text{NN}}} = \sum_{k=2}^{\mathfrak{L}} \dim(w_k) + \dim(b_k)$.

5.3.2.2 Neural networks representing the trading strategy

As mentioned above, the trading strategy π consists of three parts; *i*) the static amount invested in each option β, *ii*) the static strike prices of the options K, and *iii*) the dynamic amount invested in each stock α. Recall that β and K are decided at $t = 0$, and, as long as we have a deterministic initial wealth x_0^{IC}, we want to have a deterministic representation for β and K (this is what is meant by a static strategy above). The third component, α, on the other hand, does not have to be deterministic since it may depend on previous performance, which is affected by randomness through the stock process (this is what is meant by a dynamic strategy above). Below, we describe how each of these three components is represented, one by one, and give examples of how some trading constraints can be modeled.

The difference between the static and the dynamic parts requires different modeling strategies. For the static strategies, the trader simply needs to decide an allocation or a strike price. This can be modelled with a set of trainable parameters, which does not take an input, *i.e.,* these allocations are not functions of some input but instead constant vectors. For the dynamic trading strategy, which, at each trading date, is a function of the current wealth, we use a deep neural network taking the current wealth as input and outputs the stock allocation. The set of admissible trading strategies $\Pi = \{\Pi^\beta, \Pi^K, \Pi^{\alpha_0}, \Pi^{\alpha_1}, \ldots, \Pi^{\alpha_{N-1}}\}$, where we note that $\Pi^{\alpha_1}, \ldots, \Pi^{\alpha_{N-1}}$, may also depend on the stock process. For instance, there may be a constraint on how high the exposure can be in a single stock which also depends on the evolution of that specific stock value.

Static option strategy (β):
The number of trainable parameters needed to represent β is the same as the number of options, *i.e.,* N^{options}, implying that $\theta^\beta \in \mathbb{R}^{N^{\text{options}}}$. Moreover, an activation function $\boldsymbol{a}^\beta \colon \mathbb{R}^{N^{\text{options}}} \to \Pi^\beta$ is applied to ensure that the trading constraints are satisfied. The obtained $\beta-$strategy is then given by

$$\hat{\beta} = \boldsymbol{a}^\beta(\theta^\beta).$$

For instance, if there are no constraints on how we are allowed to trade in the options, then $\Pi^\beta = \mathbb{R}^{N^{\text{options}}}$ and $\boldsymbol{a}^\beta$ would be the component-wise identity function. A more realistic constraint would be to not allow short selling and a maximum total amount of $\beta^{\max}$ plus a maximum amount of $\frac{\beta^{\max}}{N^{\text{options}}}$ in each option, which would give $\Pi^\beta = [\mathbf{0}^{N^{\text{options}}}, \mathbf{1}^{N^{\text{options}}} \times \frac{\beta^{\max}}{N^{\text{options}}}]$, where $\mathbf{0}^{N^{\text{options}}}$ and $\mathbf{1}^{N^{\text{options}}}$ are vectors of zeros and ones, respectively. A natural way to achieve this is to set

$$\boldsymbol{a}^\beta(\theta^\beta) = \frac{\beta^{\max}}{N^{\text{options}}} \times \text{sigmoid}(\theta^\beta), \tag{5.13}$$

where sigmoid$(\cdot)$ is the component-wise sigmoid function, which in each component is given by $(1 - \mathrm{e}^{-x})^{-1}$.

Static strike price strategy (K):
Similar to β above, the number of trainable parameters for the strike prices is N^{options} and we use the notation $\boldsymbol{a}^K \colon \mathbb{R}^{N^{\text{options}}} \to \Pi^K$ with the K-strategy

$$\hat{K} = \boldsymbol{a}^K(\theta^K).$$

In practice, there is a discrete set of strike prices available in the market, and a reasonable model is to consider each option strike price in some interval $[K^{\text{low}}, K^{\text{high}}]$, implying that $\Pi^K = [K^{\text{low}} \times \mathbf{1}^{N^{\text{options}}}, K^{\text{high}} \times \mathbf{1}^{N^{\text{options}}}]$. A natural choice for such an activation function is given by

$$\boldsymbol{a}^K(\theta^K) = (K^{\text{high}} - K^{\text{low}}) \odot \text{sigmoid}(\theta^K) \oplus K^{\text{low}}, \tag{5.14}$$

where $\odot$ and $\oplus$ denote component-wise multiplication and addition, respectively.

Dynamic stock and bond strategy (α):
Since the allocation into the bond and the stocks changes at each trading date, we need to approximate the sequence $\alpha = \{\alpha_0, \ldots, \alpha_{N-1}\}$. At t_0, the allocation is deterministic, since we assume a fixed initial wealth, therefore α_0 needs to be treated differently from α_n, for $n \in \{1, \ldots, N-1\}$. The number of trainable parameters in the representation of α_0 is N^{stocks} (the amount traded in the bond is given to maintain the self-financing constraint) and we use the notation $\boldsymbol{a}^{\alpha_0} \colon \mathbb{R}^{N^{\text{stocks}}} \to \Pi^{\alpha_0}$, with α_0-strategy

$$(\hat{\alpha}_0^1, \ldots, \hat{\alpha}_0^{N^{\text{stocks}}})^\top = \boldsymbol{a}^{\alpha_0}(\theta^{\alpha_0}).$$

A typical setup at t_0 can be that the trader is allowed to allocate a specific amount within a specified interval $[A_0^{\text{low}}, A_0^{\text{high}}]$, which gives $\Pi^{\alpha_0} = [A_0^{\text{low}} \times \mathbf{1}^{N^{\text{stocks}}}, A_0^{\text{high}} \times \mathbf{1}^{N^{\text{stocks}}}]$. Note that α_0^k is the amount of stock k in the portfolio and not the amount of cash allocated into stock k. Therefore, it is A_0^k that is supposed to be within $[A_0^{\text{low}}, A_0^{\text{high}}]$. On the other hand, since we have assumed that the initial values of all assets equal 1, the definition still holds. Similar to above, we can then use the activation function

$$\boldsymbol{a}^{\alpha_0}(\theta^{\alpha_0}) = (A_0^{\text{high}} - A_0^{\text{low}}) \odot \text{sigmoid}(\theta^{\alpha_0}) \oplus A_0^{\text{low}}. \tag{5.15}$$

Finally, $\hat{\alpha}_0$ (which also includes the initial bond allocation) is given by

$$\hat{\alpha}_0 = \left(\hat{\alpha}_0^0, \boldsymbol{a}^{\alpha_0}(\theta^{\alpha_0})^\top\right)^\top,$$

where $\hat{\alpha}_0^0 = \hat{x}_0 - \sum_{k=1}^{N^{\text{stocks}}} \hat{\alpha}_0^k$, with $\hat{x}_0 = x_0^{\text{IC}} - \hat{y}_0$ and $\hat{y}_0 = \sum_{k=1}^{N^{\text{options}}} \hat{\beta}^k$.

The above trading strategies all have in common that they are decided at time t_0 before any randomness enters the system, and hence, they do not depend on previous performance. For trading after t_0, on the other hand, we need to take previous performance, as well as random realization of the asset processes, into account. Therefore, we use fully connected neural networks of the form from Section 5.3.2.1

$$(\hat{\alpha}_n^1, \dots, \hat{\alpha}_n^{N^{\text{stocks}}})^\top = \phi(S_{t_n}; \theta^{\alpha_n}).$$

Given a parametrization θ^{α_n}, the above is a mapping from S_{t_n}, which represents the value of all the stocks at t_n, to the new stock allocation, or α_n-strategy. On the other hand, given S_{t_n}, the neural network is a mapping from the parameter space to the admissible strategy space, *i.e.*, $\phi(S_{t_n}; \cdot)\colon \mathbb{R}^{\mathfrak{D}^{\theta_n}} \to \Pi^{\alpha_n}$, where $\mathfrak{D}^{\theta_n}$ is the number of trainable parameters, as specified in Section 5.3.2.1. The activation functions in the interior layers can be chosen arbitrarily, however, as for the strategies above, the activation function in the output layer should be chosen such that the trading constraints are satisfied. With the same trading constraints as for t_0, we have $\Pi^{\alpha_n} = [A_n^{\text{low}} \times \mathbf{1}^{N^{\text{stocks}}} \oslash S_{t_n}, A_n^{\text{high}} \times \mathbf{1}^{N^{\text{stocks}}} \oslash S_{t_n}]$, where $\oslash$ denotes component-wise division. The reason for dividing by the stock process above is that it is usually not the amount of each stock that matters from a constraint perspective, but rather the value of the traders position in each stock. We specify a possible activation function in the output layer, which guarantees that the trading constraints are obeyed:

$$\boldsymbol{a}^{\alpha_n}(\chi) = \left[(A_n^{\text{high}} - A_n^{\text{low}}) \odot \text{sigmoid}(\chi) \oplus A_n^{\text{low}}\right] \oslash S_{t_n}, \tag{5.16}$$

where $\chi = w_{\mathfrak{L}}^\top L_{\mathfrak{L}-1} + b_{\mathfrak{L}}$, with $L_{\mathfrak{L}-1}$ is the vector from layer $\mathfrak{L}-1$ (with slight abuse of notation) and $w_{\mathfrak{L}}$ and $b_{\mathfrak{L}}$ are the weight and bias vectors, respectively. Similar to above, we have $\hat{\alpha}^n$, which is given by

$$\hat{\alpha}_n = \left(\hat{\alpha}_0^n, \phi(S_{t_n}; \theta^{\alpha_n})^\top\right)^\top,$$

where $\hat{\alpha}_n^0 = (B_{t_n})^{-1}(\hat{x}_{t_n} - \sum_{k=1}^{N^{\text{stocks}}} \hat{\alpha}_n^k S_{t_n}^k)$, with $\hat{x}_{t_n} = \hat{\alpha}_n^0 B_n + \sum_{k=1}^{N^{\text{stocks}}} \hat{\alpha}_{n-1}^k S_{t_n}^k$.

5.3.2.3 Optimization problem with neural networks

Summarizing, all the trainable parameters from above, we conclude that our neural network-based trading strategy $(\hat{\beta}, \hat{K}, \hat{\alpha})$ is determined by the parameters $\theta = (\theta^\beta, \theta^K, \theta^{\alpha_0}, \theta^{\alpha_1} \dots, \theta^{\alpha_{N-1}})$, where each component is vectorized before concatenation. Note that θ as given above contains all the trainable parameters for a sequence of networks which together control the trading strategy, while in (5.11), θ denotes the trading strategy itself. Even though this constitutes a minor abuse of notation, the simplicity weighs in favor of a simpler notation.

Below, a list of the strategy evaluated at one random realization of the underlying asset process is given by

$\{\boldsymbol{a}^K(\theta^K), \boldsymbol{a}^\beta(\theta^\beta), \boldsymbol{a}^{\alpha_0}(\theta^{\alpha_0}), \phi_1(\hat{x}_{t_1};\theta^{\alpha_1}), \dots, \phi_{N-1}(\hat{x}_{t_{N-1}};\theta^{\alpha_{N-1}})\}$, and the full neural network-based optimization problem is defined by

$$\begin{cases} \underset{\theta\in\Theta^{\mathrm{NN}}}{\text{maximize}} = \bar{U}^M(\theta), \quad \text{where the } M \textit{ i.i.d.} \text{ random variables follow,} \\ R(S;\theta) = R_{\mathrm{SB}}(S;\theta) + R_{\mathrm{O}}(S;\theta) \\ R_{\mathrm{SB}}(S;\theta) = \hat{x}_{t_N} - \hat{x}_0 - \sum_{k=1}^{N^{\mathrm{stocks}}} \mathrm{TC}^k, \quad R_{\mathrm{O}}(S;\theta) = \hat{y}_{t_N} - \hat{y}_0, \\ \hat{x}_{t_N} = \hat{x}_0 + \sum_{n=0}^{N} \mathbb{I}_{\{x>0\}}(\hat{x}_{t_n})\big[\hat{\alpha}_n^0(B_{t_{n+1}} - B_{t_n}) + \sum_{k=1}^{N^{\mathrm{stocks}}} \hat{\alpha}_n^k(S_{t_{n+1}} - S_{t_n})\big], \quad \hat{x}_0 = \hat{x}_0^{\mathrm{IC}} - \hat{y}_0, \\ \hat{y}_{t_N} = \sum_{i=1}^{N^{\mathrm{options}}} \hat{\beta}^i V^i(T, S_{t_N};\hat{K}^i), \quad \hat{y}_0 = \sum_{i=1}^{N^{\mathrm{options}}} \hat{\beta}^i V^i(0, S_0;\hat{K}^i), \\ \hat{\alpha}_0^0 = \hat{x}_0 - \sum_{k=1}^{N^{\mathrm{stocks}}} \hat{\alpha}_0^k, \quad (\hat{\alpha}_0^1, \dots, \hat{\alpha}_0^{N^{\mathrm{stocks}}})^\top = \boldsymbol{a}^{\alpha_0}(\theta^{\alpha_0}), \quad \hat{\beta} = \boldsymbol{a}^\beta(\theta^\beta), \quad \hat{K} = \boldsymbol{a}^K(\theta^K), \\ \hat{\alpha}_n^0 = \dfrac{1}{B_{t_n}}(\hat{x}_{t_n} - \sum_{k=1}^{N^{\mathrm{stocks}}} \hat{\alpha}_n^k S_{t_n}^k), \quad (\hat{\alpha}_n^1, \dots, \hat{\alpha}_n^{N^{\mathrm{stocks}}})^\top = \phi(\hat{x}_{t_n};\theta^{\alpha_n}). \end{cases} \tag{5.17}$$

Note that in the above, $\bar{U}^M(\theta) = u(\bar{R}^M(\theta))$, where $\bar{R}^M(\theta) = \{R(S(1);\theta), \dots, R(S(M);\theta)\}$ is the empirical distribution from M samples distributed as described in (5.17) above.

5.3.2.4 Pseudo-code

Denote the number of training samples, batch size and the number of epochs by M_{train}, M_{batch}, $M_{\mathrm{epoch}} \in \mathbb{N}_+$, respectively. Pseudo-code aiming to describe the proposed algorithm is given below. For notational convenience, the pseudo-code describes the algorithm in the special case when $M_{\mathrm{epoch}} = 1$, but the extension to $M_{\mathrm{epoch}} > 1$ can simply be achieved by repeating the procedure M_{epoch} times.

5.3.2.5 Pre-commitment strategy

The allocation strategy obtained by our algorithm is by design of pre-commitment type. The reason for this is that the cost is computed only once and the optimization is performed globally. If the DPP would be satisfied for the problem at hand then the obtained pre-commitment strategy would also be time-consistent.

5.4 Numerical experiments

In this section, we aim to demonstrate the effectiveness and flexibility of the proposed algorithm. In previous sections, the asset process has not been fully specified and the reason for this is that any process, computer generated or real world data, can be used. In this section, on the other hand, we need to decide how to generate the training data and we use the jump-diffusion SDE. Moreover, the fully implementable model includes the market frictions as well as the European call and put options, which are introduced below. A special case, which fits into this framework, is the mean-variance (MV) portfolio optimization problem, which admits

Input: Initialization of neural network parameters, $\{\theta^\beta(1), \theta^K(1), \theta_0^\alpha(1), \dots, \theta_{N-1}^\alpha(1)\}$, initial state S_0, initial cash $\hat{x}^{\mathrm{IC}}$, and for $0 \le m \le M_{\mathrm{train}}$ and $1 \le n \le N-1$, asset state $S_{t_n}(m)$.

Output: Approximation of the trading strategy, determined by β, K and $(\alpha_t)_{t\in\mathcal{T}^N}$.

for $k = 1, 2, \dots, K_{\mathrm{batch}} - 1$, *where* $K_{\mathrm{batch}} = M_{\mathrm{train}}/M_{\mathrm{batch}}$ is the number of batches. (should be carried out sequentially) **do**

  for $m \in \{1, \dots, M_{\mathrm{batch}}\}$ (may be carried out in parallel) **do**

    $\hat{\beta} = \boldsymbol{a}^\beta(\theta^\beta(k)), \quad \hat{K} = \boldsymbol{a}^K(\theta^K(k)), \quad \hat{y}_0 = \sum_{i=1}^{N^{\mathrm{options}}} \hat{\beta}^i V^i(0, S_0\,;\, \hat{K}^i), \quad \hat{x}_0 = \hat{x}_0^{\mathrm{IC}} - \hat{y}_0, \quad (\hat{\alpha}_0^1, \dots, \hat{\alpha}_0^{N^{\mathrm{stocks}}})^\top = \boldsymbol{a}^{\alpha_0}(\theta^{\alpha_0}(k)), \quad \hat{\alpha}_0^0 = \hat{x}_0 - \sum_{k=1}^{N^{\mathrm{stocks}}} \hat{\alpha}_0^k,$
    $\hat{x}_{t_1}(m) = \hat{x}_{t_0} + \mathbb{I}_{\{x>0\}}(\hat{x}_{t_0})\big[\hat{\alpha}_0^0(B_{t_1} - B_{t_0}) + \hat{\alpha}_0^k(S_{t_1}(m) - S_0)\big]$

    for $n = 1, \dots, N-1$ (should be carried out sequentially) **do**

      $(\hat{\alpha}_n^1(m), \dots, \hat{\alpha}_n^{N^{\mathrm{stocks}}})^\top = \phi(S_{t_n}(m)\,;\theta^{\alpha_n}(k)), \quad \hat{\alpha}_n^0(m) = \frac{1}{B_{t_n}}\big(\hat{x}_{t_n} - \sum_{k=1}^{N^{\mathrm{stocks}}} \hat{\alpha}_n^k S_{t_n}^k(m)\big), \; \hat{x}_{t_{n+1}}(m) = \hat{x}_{t_n}(m) + \mathbb{I}_{\{x>0\}}(\hat{x}_{t_n}(m))\big[\hat{\alpha}_n^0(m)(B_{t_{n+1}} - B_{t_n}) + \hat{\alpha}_n^k(m)(S_{t_{n+1}}(m) - S_{t_n}(m))\big]$

    end

    $R(S(m)\,;\theta(k)) = R_{\mathrm{SB}}(S(m)\,;\theta(k)) + R_{\mathrm{O}}(S(m)\,;\theta(k))$
    $R_{\mathrm{SB}}(S(m)\,;\theta(k)) = \hat{x}_{t_N}(m) - \hat{x}_0 - \sum_{k=1}^{N^{\mathrm{stocks}}} \mathrm{TC}^k(m), \quad R_{\mathrm{O}}(S(m)\,;\theta(k)) = \hat{y}_{t_N}(m) - \hat{y}_0,$

  end

  $\theta(k) = \{\theta^\beta(k), \theta_K(k), \theta_{\alpha_0}(k), \theta_{\alpha_1}(k) \dots, \theta^{\alpha_{N-1}}(k)\}$, (trainable parameters)
  $\bar{R}^{M_{\mathrm{batch}}}(\theta(k)) = \{R(S(1)\,;\theta(k)), \dots, R(S(M_{\mathrm{batch}})\,;\theta(k))\}$ (empirical distribution).
  $\bar{U}^{M_{\mathrm{batch}}}(\theta(k)) = u(\bar{R}^{M_{\mathrm{batch}}}(\theta(k)))$ (Loss-function)
  $\theta(k+1) \leftarrow \arg\min_\theta u(\bar{R}^{M_{\mathrm{batch}}}(\theta))$ (some optimization algorithm, usually of gradient ascent type).

end

Algorithm 2: Pseudo-code of one epoch of the neural network training.

to a closed-form analytic solution (an admissible trading strategy in closed-form, which is optimal in the sense of the objective function). This solution is used here as a reference for our approximate trading strategy for the MV portfolio optimization problem. In subsequent sections, we go beyond classical MV problems and consider market frictions, other objective functions as well as more general asset dynamics. This leads to interesting investment strategies in which combinations of a bond, stocks and options seem optimal.

The asset model used in this chapter is a Geometric Brownian motion with multiplicative jumps, which is given by

$$\begin{cases} \mathrm{d}S_t = b \odot S_t \mathrm{d}t + \sigma S_t \odot \mathrm{d}W_t + J \odot \mathrm{d}X_t; \quad S_0 = (1, \dots, 1)^\top, & (5.18) \\ \mathrm{d}B_t = rB_t \mathrm{d}t; \quad B_0 = 1. & (5.19) \end{cases}$$

Here, $T \in [0, \infty)$, $N^{\text{stocks}} \in \mathbb{N}$, $(W_t)_{t\in[0,T]}$ is an N^{stocks}-dimensional standard Brownian motion, J is an N^{stocks}-dimensional multivariate normally distributed random variable with mean $\mu_J \in \mathbb{R}^{N^{\text{stocks}}}$ and covariance $\Sigma_J \in \mathbb{R}^{N^{\text{stocks}} \times N^{\text{stocks}}}$, $(X_t)_{t\in[0,T]}$ is an N^{stocks}-dimensional Poisson process parametrized by $\xi_p \in \mathbb{R}_+^{N^{\text{stocks}}}$ and model parameters $b \in \mathbb{R}^{N^{\text{stocks}}}$ and $\sigma \in \mathbb{R}^{N^{\text{stocks}} \times N^{\text{stocks}}}$. The initial values S_0 and B_0 are scaled to unity for the purpose of simplicity.

Setting $x_0 = 1$ and using a binary parameter for the no-bankrupcy-constraint $\text{NB} \in \{0, 1\}$, we have

$$\begin{aligned} \mathrm{d}x_t =& (1 - \text{NB} \times \mathbb{I}_{\{x\leq 0\}}(x_t)) \Big[(r\alpha_t^0 B_t + \sum_{i=1}^{N^{\text{stocks}}} b_i \alpha_t^i S_t^i)\mathrm{d}t \\ &+ \sum_{i=1}^{N^{\text{stocks}}} \sum_{j=1}^{N^{\text{stocks}}} \sigma_{ij} \alpha_t^i S_t^i \mathrm{d}W_t^j + \sum_{i=1}^{N^{\text{stocks}}} \alpha_t^i S_t J_t^i \mathrm{d}X_t^i \Big] \\ =& (1 - \text{NB} \times \mathbb{I}_{\{x\leq 0\}}(x_t)) \Big[(rx_t + \sum_{i=1}^{N^{\text{stocks}}} (b_i - x_t) N_t^i)\mathrm{d}t \\ &+ \sum_{i=1}^{N^{\text{stocks}}} \sum_{j=1}^{N^{\text{stocks}}} \sigma_{ij} N_t^i \mathrm{d}W_t^j + \sum_{i=1}^{N^{\text{stocks}}} N_t J_t^i \mathrm{d}X_t^i \Big], \end{aligned} \tag{5.20}$$

where $N_t^i = \alpha_t^i S_t^i$. Transaction costs are expressed according to (5.8). Note that the above formulation is expressed in a time-continuous fashion, while in Section 5.2, the allocation process, as well as x, are expressed in a time discrete way. The reason for this is that the classical MV-optimization problem admits to a closed-form solution for the allocation process which offers the opportunity to compare with a reference solution in this special case. Moreover, by considering a piecewise continuous allocation process, the formulations coincide (up to a discrete approximation of the geometric Brownian motion with jumps).

In all experiments, the objective function is approximated in the simplest possible way, by means of its empirical counterpart. For instance, the mean and variance are approximated by the built in Numpy-functions (or, when appropriate, Tensorflow-functions). The expected shortfall is approximated as the average of the empirical tail of a sample. However, it should be pointed out that for these specific examples (variance and expected shortfall), we could instead employ the embedding techniques described in [7, 130]. This would potentially result in simpler optimization and, in turn, in the possibility to use smaller training batches. On the other hand, we want to emphasize the generality of our method and demonstrate that it does

not rely on problem-specific reformulations. Therefore, we keep the empirical loss functions as close as possible to their original forms.

General neural network settings Although any function approximator can be used in (5.17), we choose a sequence of neural networks (as described in the pseudo-code in Algorithm 2). In this chapter, the neural networks are seen merely as tools to solve the problem. Therefore, the focus is not placed on optimizing the structure or parameter choices but rather on finding a setting which is not too much problem dependent. We therefore use the same settings, which are given below, for all problems considered in this chapter. Following the notation in Sections 5.3.2.1 and 5.3.2.4, the total number of training samples is set to $M_{\text{train}} = 2^{22}$, the batch size to $M_{\text{batch}} = 2^{12}$, the number of epochs to $M_{\text{epoch}} = 10$ and the number of layers to $\mathfrak{L} = 4$. For the interior layers, *i.e.,* $\ell \in \{2, 3\}$, we set the number of nodes to $\mathfrak{N}_\ell = 20$ and the activation functions $\mathbf{a}_\ell(\cdot) = \text{ReLU}(\cdot)$. The input dimension $\mathfrak{D}_{\text{input}} = 1$ and the output dimension $\mathfrak{D}_{\text{output}}$, as well as the activation function in the output layer depend on the trading constraints and are specified for each specific problem below. We use 0.01 as initial learning rate and after two batches it decreases with a factor $\exp(-0.5)$ for each new batch. We do not apply regularization or normalization techniques.

5.4.1 Classical continuous mean-variance optimization

As a first example, we consider the classical mean-variance optimization problem. This means that the asset process (5.18) is a pure geometric Brownian motion. This can fit our framework, for instance, by setting the μ_J and Σ_J to a zero vector and a zero matrix, respectively. Moreover, trading should be carried out without transaction costs and no-bankruptcy constraint, *i.e.,* setting the courtage, $C = 0$ and $\text{NB} = 0$ and there should be no constraints for short selling, leverage or bankruptcy. Here trading in the options is not allowed and the algorithm used is D-TIP. The MV objective function is given by

$$U(\theta) = E[x_T] - \lambda \text{Var}[x_T],$$

where $\lambda > 0$ controls the risk aversion. From [7], we know that there exists an analytic solution to this problem in terms of a closed-form expression for the optimal allocation as well as a theoretical optimal mean and variance of the terminal wealth, *i.e.,* an infimum of the objective function.

In the example below, the following parameter values are used

$$T = 2, N = 20, r = 0.06, b = \begin{pmatrix} 0.08 \\ 0.07 \\ 0.06 \\ 0.05 \\ 0.04 \end{pmatrix}, \sigma = \begin{pmatrix} 0.23 & 0.05 & -0.05 & 0.05 & 0.05 \\ 0.05 & 0.215 & 0.05 & 0.05 & 0.05 \\ -0.05 & 0.05 & 0.2 & 0.05 & 0.05 \\ 0.05 & 0.05 & 0.05 & 0.185 & 0.05 \\ 0.05 & 0.05 & 0.05 & 0.05 & 0.17 \end{pmatrix}, \lambda = 1.10. \tag{5.21}$$

The parameter setting as given above leads to a non-symmetric problem with asset dynamic with correlated components, *i.e.,* each stock is correlated to the other stocks. Moreover, we vary between positive correlation and negative correlation, which typically is the case in factor

investing [160].

Problem specific neural network settings In addition to the general neural network settings from Section 5.3.2 and the introduction of Section 5.4, we here give the finer, problem specific, details. Since D-TIP does not allow options in the portfolio, we only consider the networks that compute the dynamic stock and bond strategy.

We have one neural network per allocation point and the requirements are *i*) the output dimension should coincide with the number of stocks, N^{stocks}, and *ii*) since we do not apply any trading constraints, the range for each individual stock allocation is $\mathbb{R}$ and hence the activation function in the output layer should have $\mathbb{R}$ as its range for each component. This is achieved by adjusting (5.15) and (5.16) to

$$\begin{aligned} \boldsymbol{a}^{\alpha_0}(\theta^{\alpha_0}) &= \theta^{\alpha_0}, \quad \text{with } \theta^{\alpha_0} \in \mathbb{R}^{N^{\text{stocks}}}, \\ \boldsymbol{a}^{\alpha_n}(\chi) &= \chi, \quad \text{with } \chi \in \mathbb{R}^{N^{\text{stocks}}}, \quad \text{for } n \in \{1, 2, \dots, N-1\}, \end{aligned}$$

where χ is the output of layer 3, *i.e.,* the output of last hidden layer in our neural networks.

The optimal value of the objective function is approximately 1.1637 and in Figure 5.3, top left, we see the convergence of our approximation to the true value with respect to the number of epochs[2]. It should be emphasized that the implementation results in a discrete approximation of a continuous problem. By increasing N the approximation should converge with respect to the step size $h = T/N$, but to investigate this is beyond the scope of this chapter. The top right plot shows the expected value as well as the 5:th and 95:th percentiles of the wealth as a function of time. We see that the approximation corresponds well to the analytic counterpart. The plots at the bottom display a comparison of the empirical pdfs and CDFs, respectively, in which it is clear that our approximation corresponds well to the empirical distributions obtained from the analytic allocation strategy.

5.4.2 Beyond MV, with market frictions and jumps

In this subsection, we consider the full generality offered by the asset model from (5.18)-(5.19), as well as transaction costs, no bankruptcy constraint and we allow for trading in European call and put options. When also options are allowed to trade in, the algorithm used is D-TIPO. The parameter values from (5.21) are reused and we set $\lambda_J = 0.05$, $\mu_J = (0, \dots, 0)^\top$, $\Sigma_J = \text{diag}(0.2, \dots, 0.2)$, NB $= 1$ and $C = 0.005$. This means that we have an asset price process with symmetric jumps around zero for each individual asset and a courtage of 0.5% for each trade. This could seem high, but an alternative interpretation of C is as a penalizing term for too heavy reallocation (which is something that for instance pension funds want to avoid). With a slight abuse of notation, we denote the discretized version of (5.18) by S but keep in mind that we must rely on an approximation in this section. In all our examples, we use N (the number of allocation dates) as time steps in an Euler–Maruyama scheme, however, if necessary, a finer mesh can be used. In turn, this also implies that (5.20) has to be approximated and this is done with the Euler–Maruyama scheme as well. When there is no risk of confusion, we

[2]Since our data is synthetic we could have generated new data instead of reusing data in each epoch. On the other hand, since our algorithm is data-driven, it would be possible to use real world data which makes it desirable to be able to reuse data.

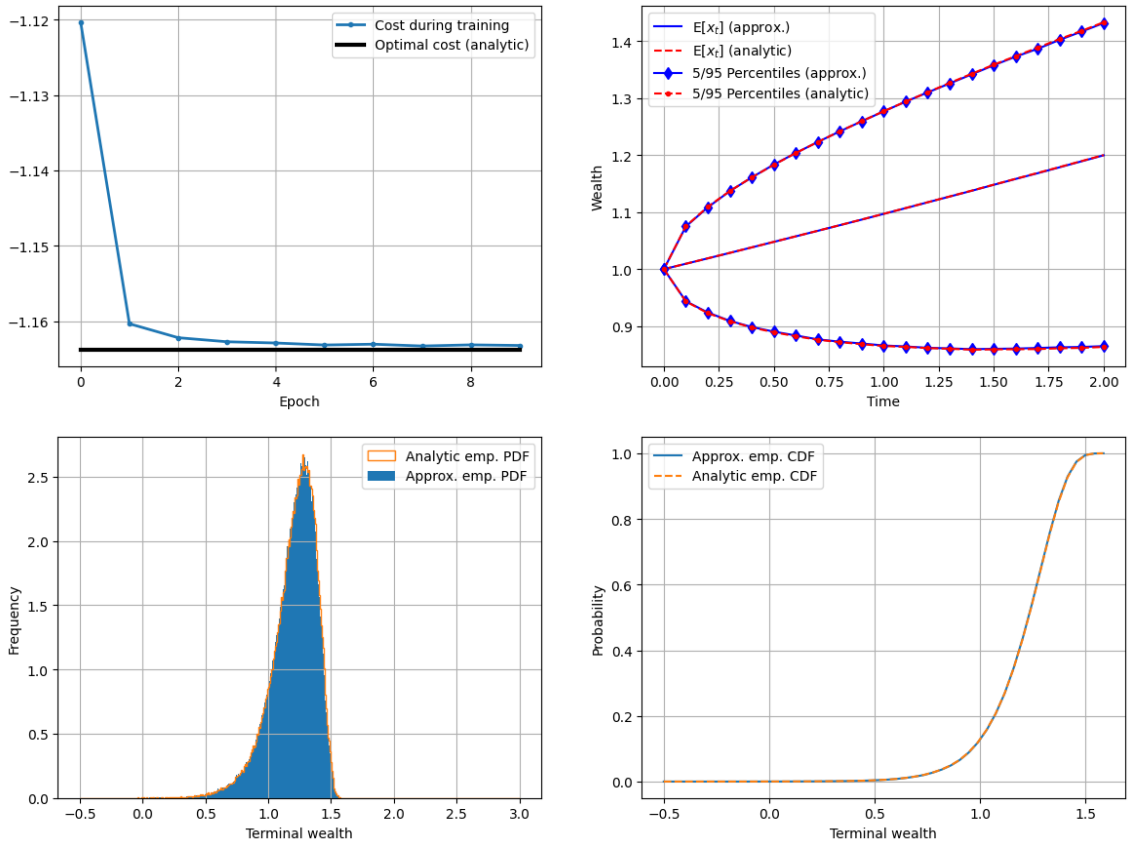

Figure 5.3: Upper left: Convergence of the loss to the analytic counterpart with respect to the number of training epochs. **Upper right:** Comparison of our approximation and the reference solution. Upper lines, middle lines and lower lines represent the 95th percentiles, the mean values and the 5th percentiles, respectively. **Lower left:** Comparison of the empirical pdfs of our approximation and the reference solution. **Lower right:** Comparison of the empirical CDFs of our approximation and the reference solution.

simply denote the discretized versions of S and x by

$$S_n = S_{t_n}, \quad x_n = x_{t_n},$$

where t_n is the n:th allocation date.

For each stock, S^i, $i \in \{1, \dots, N^{\text{stocks}}\}$, we allow trading in one European call option and one European put option with terminal time T and strike K_i. For $i \in \{1, \dots, N^{\text{stocks}}\}$, denoted by $\bar{V}^i(t, S_t^i; K^i)$, the risk-neutral price of a call option, and for $i \in \{\dots, N^{\text{stocks}} + 1, \dots, 2 \dots, N^{\text{stocks}}\}$, we denote by $\bar{V}^{N^{\text{stocks}}+i}(t, S_t^i; K^{N^{\text{stocks}}+i})$ a put option with S^i as underlying. Note that a drift correction term needs to be added to the asset dynamics to obtain a risk-neutral measure, which is necessary in order to make the option values martingales under the jump-diffusion SDE, see *e.g.,* [161]. We then have that $\bar{V}^i(T, S_T^i; K^i) = \max(0, S_T^i - K^i)$ and $\bar{V}^{N^{\text{stocks}}+i}(T, S_T^i; K^{N^{\text{stocks}}+i}) = \max(0, K^{N^{\text{stocks}}+i} - S_T^i)$. We want to work with normalized entities and define

$$V^i(0, S_0^i; K^i) = 1, \; V^i(T, S_T^i; K^i) = \frac{\max(0, S_T^i - K^i)}{\bar{V}^i(0, S_0^i; K^i)},$$

$$V^{N^{\text{stocks}}+i}(0, S_0^i; K^{N^{\text{stocks}}+i}) = 1, \; V^{N^{\text{stocks}}+i}(T, S_T^i; K^{N^{\text{stocks}}+i}) = \frac{\max(0, K^{N^{\text{stocks}}+i} - S_T^i)}{\bar{V}^{N^{\text{stocks}}+i}(0, S_0^i; K^{N^{\text{stocks}}+i})}.$$

In this section, we use the objective function from (5.10). The main reason is that this extended version of the mean-variance objective function allows us to better control the tails of the final wealth distribution (or in fact, of the return R). We want to use an objective function which has similarities to mean-variance since it is desirable to achieve a high mean and a low variance of the terminal wealth. On the other hand, we want to be able to better control the tail distribution. With this in mind, we add two terms aiming to increase the expected value in the tails. Below we give the theoretical version of the used objective function, but when implemented, we always resort to $\bar{U}^M(\theta)$, which is the M-sample empirical counterpart to $U(\theta)$.

$$U(\theta) = \mathbb{E}[R] - \lambda_1 \text{Var}[R] + \lambda_2 \text{ES}_{p_1}^-(R) + \lambda_3 \text{ES}_{p_2}^+(R).$$

We use $p_1 = 0.01$ which means that we penalize low values of the expected return of the worst 1% performance of the portfolio. For the upper tail, we want to maximize the expected return of the 5% best outcomes and therefore set $p_2 = 0.95$. The weights are set to $\lambda_1 = 0.552$, $\lambda_2 = 0.276$ and $\lambda_3 = 0.110$.

Problem specific neural network settings In addition to the general neural network settings from Section 5.3.2 and the introduction of Section 5.4, we here give the finer details which are problem specific. The D-TIPO algorithm does not only return a dynamic trading strategy for the bond and the stocks, as in classical MV-optimization, but also static strategies for allocations into a set of options as well as a strike price for each option. For the option allocation, we follow (5.13) and set $\beta_{\max} = 1$. This implies that the maximum total amount of allocation into the options is 100% of the initial wealth. One problem with (5.13) is that when the number of stocks increases, the possible allocation into each option decreases. Since $N_{\text{options}} = 10$ in the example considered in this section, the maximum amount allocated into

each specific option is 0.1. We therefore use a slight modification of (5.13), given by

$$\boldsymbol{a}^{\beta}(\theta^{\beta}) = \frac{1}{\max\{1, \hat{\beta}\}} \times \hat{\boldsymbol{a}}^{\beta}(\theta^{\beta}),$$

where $\hat{\boldsymbol{a}}^{\beta}(\theta^{\beta}) = \frac{\beta^{\max}}{N^{\text{options}}} \times \text{sigmoid}(\theta^{\beta})$ and $\hat{\beta} = \sum_{i=1}^{N_{\text{options}}} [\hat{\boldsymbol{a}}^{\beta}(\theta^{\beta})]_i$. In this way, the allocation range into each option is $[0, \beta^{\max}]$ while keeping the total allocation range (the sum of the allocations into each option) to $[0, \beta^{\max}]$ as well.

For the strike prices, we follow (5.14) and use $K^{\text{low}} = (0.75, 0.75 \ldots, 0.75)^{\top}$ and $K^{\text{high}} = (1.25, 1.25 \ldots, 1.25)^{\top}$, *i.e.,* setting the range for strike prices to between 75% to 125% of the stock price at the initial time.

Finally, for $n \in \{0, 1, \ldots, N-1\}$, (5.13) is used and we set $\alpha_n^{\text{low}} = -2x_n$ and $\alpha_n^{\text{high}} = 2x_n$ implying that we can allocate into each stock between -200% and 200% of the total value of the stocks and the bond.

5.4.2.1 Evaluation of the results

In this section, we aim to show that for a more sophisticated objective function, it can be beneficial to include options to a portfolio of stocks and a bond.

To begin with, we display the kind of strategy D-TIPO generates. In Figure 5.4 to the left, the average total allocation into the stocks (the sum of all stock positions) and the bond over time as well as the total allocation to the options (the sum of all option positions) is displayed. We see that approximately 35% of the initial wealth is allocated to the options and the remainder to the stocks and the bond. In Figure 5.4 to the right, the average allocations are displayed for each individual stock and we note that the main allocation is into stock 1, for the stock itself as well as for the call and put options. The purpose of Figure 5.5 is to gain

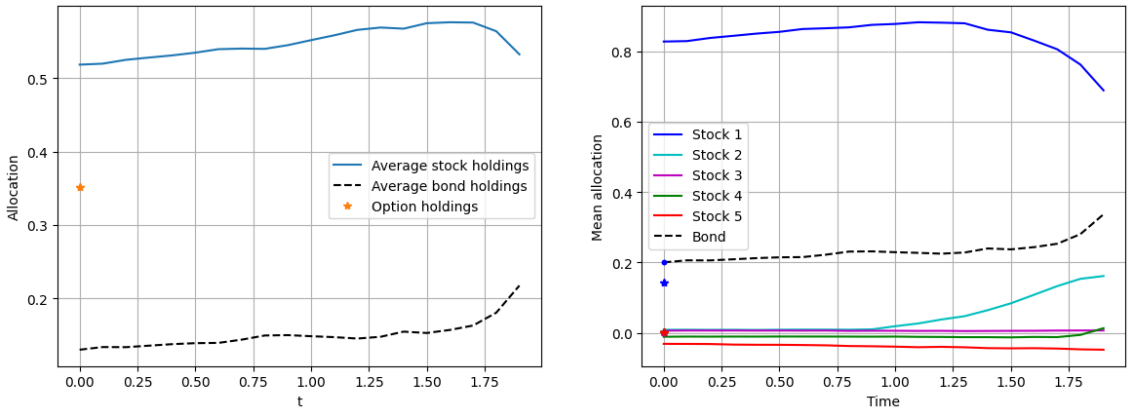

Figure 5.4: Left: Average allocation to stocks, bond and options over time. **Right:** Average allocation to stocks, bond and options over time for each stock. Asterisks and bullets represent call and put option holdings, respectively.

insight into differences between portfolios with different levels of performance, *e.g.,* is there a difference in the contribution from the different asset classes (stocks, bonds and options) for the best contra worst performing portfolios? In Figure 5.5 the average contributions of the stocks and the bond, the call options and put options are displayed for portfolios in three different performance ranges. From left to right, we see portfolios with terminal wealth less

than 1.03, between 1.03 and 1.12, and above 1.12, respectively. The strike prices are optimized by the neural networks to 0.75 for all call options and 1.25 for all put options, *i.e.,* deep in the money. This enhances the viewpoint that the main purpose of the options in the portfolio is to cover up for tail events of the stocks rather than to leverage potential upside. We also note that for the best performing outcomes, the main option contribution comes from the call options and for the worst performing outcomes from the put options.

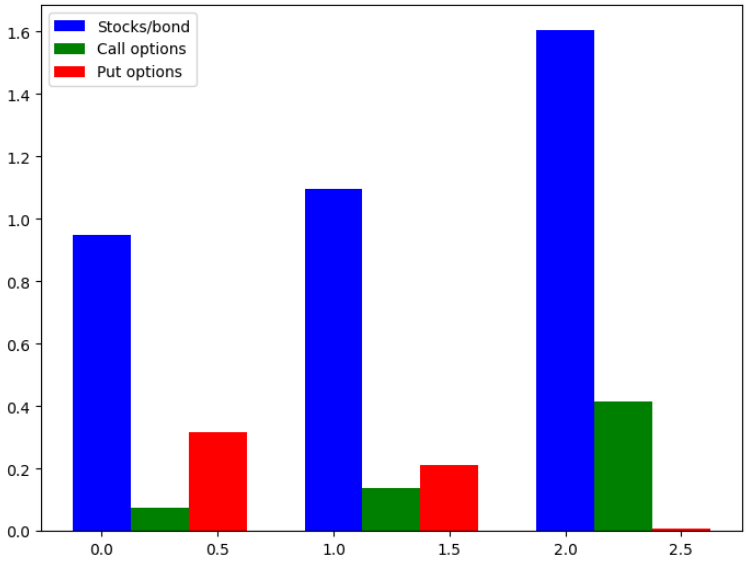

Figure 5.5: Contribution to the portfolios for terminal wealth less than 1.03 (33% of the outcomes), between 1.03 and 1.12 (41%), and above 1.12 (26%).

In Section 5.4.1, we had a closed-form solution as a reference, which made it easy to evaluate the algorithm. Unfortunately, we do not have such a reference in this section. We therefore use two different allocation strategies to compare our results with (for a problem with the same asset process and trading costs):

1. The same algorithm but with a portfolio consisting of only stocks and a bond, *i.e.,* D-TIP;

2. The MV-strategy from Section 5.4.1 which generates the same mean as D-TIPO does (1.14). This means that we compare with a strategy that is optimal for classical MV-optimization problem and apply it to the setting in this section.

For the first comparison, we use the same objective function, but a more restrictive investment directive in the sense that we are not allowed to allocate into the options. Therefore, a reasonable comparison is the optimal (up to the optimization algorithm) value of objective function. If the objective function converges to the same value during optimization with and without options in the portfolio, this would imply that it is not beneficial to add options to the portfolio, *i.e.,* D-TIPO performs in line with D-TIP. If, on the other hand, it converges to a higher value with options in the portfolio, we conclude that it adds value to include options in the portfolio, *i.e.,* D-TIPO performs better than D-TIP. In Figure 5.6, it is clear that the loss function converges to a lower value when options are allowed (we are here using the machine learning convention of a loss function, which is defined by multiplying the objective function by

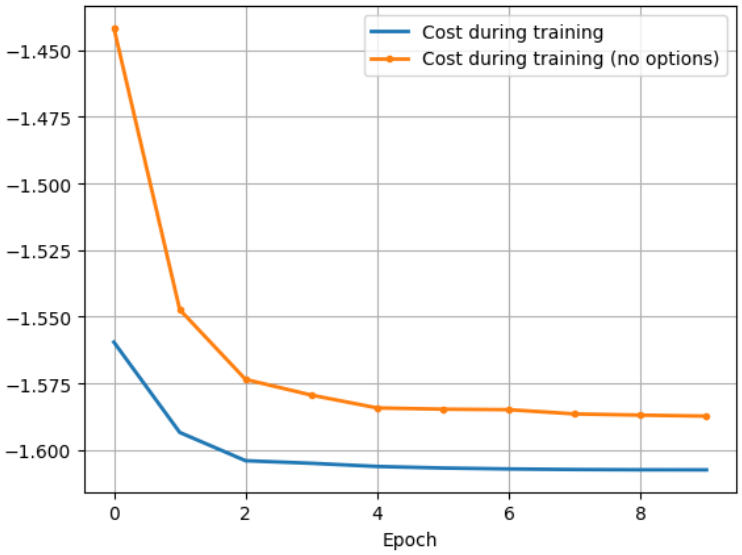

Figure 5.6: Convergence of the loss functions for D-TIPO in blue and D-TIP in orange as a function of the number of training epochs.

-1). In Figure 5.7, top figures, the allocation strategies with and without options are compared in terms of the distribution of the terminal wealth. From left to right, the empirical pdfs and the empirical CDFs are compared. We note that with options we observe *i*) a thinner left tail, *ii*) a higher density around the expected terminal wealth, and *iii*) a fatter right tail. At least the first and the last features are beneficial since the objective function aims to prevent large losses (by the lower expected shortfall term) and encourage large gains (by the upper expected shortfall term). The reason for this is the additional flexibility to shape the distribution, which is offered by adding options to the portfolio.

For the second comparison, in which we compare the D-TIPO strategy with the MV-strategy, it is no longer suitable to compare the objective functions. The reason for this is that the MV-strategy is obtained as the optimal allocation strategy for a different problem, with other asset dynamics as well as another objective function. The best we can do is to compare the distributions of the terminal wealth, which is also done in Figure 5.7. We note the same differences as in the comparison above which is not surprising since the MV-strategy only allocates into stocks and the bond. More interesting is that, in contrast to our strategies both with and without options, we encounter a fatter left tail than the right tail, which is non-desirable from a practical perspective. The reason for this is the limitation of the MV-objective function distinguish between downside risk and upside potential, *i.e.,* it does not only penalize downside risk, but also upside potential.

Similar conclusions can be drawn from Table 5.2, in which the empirical mean values, variances and expected shortfalls of the terminal wealth from the three algorithms are compared. For all measures but the variance the portfolio with options performs best. The reason for the variance to be lower without options is simply because the right tail, with a possibility of very high returns. From a practical perspective, this is clearly not a problem. In the rightmost column of Table 5.2 the transaction cost as a percentage of the initial wealth is expressed. By using the D-TIPO strategy, we see that the transaction cost decreases by more than 60% compared to the D-TIP strategy and by more than 90% compared to the MV-strategy. This is

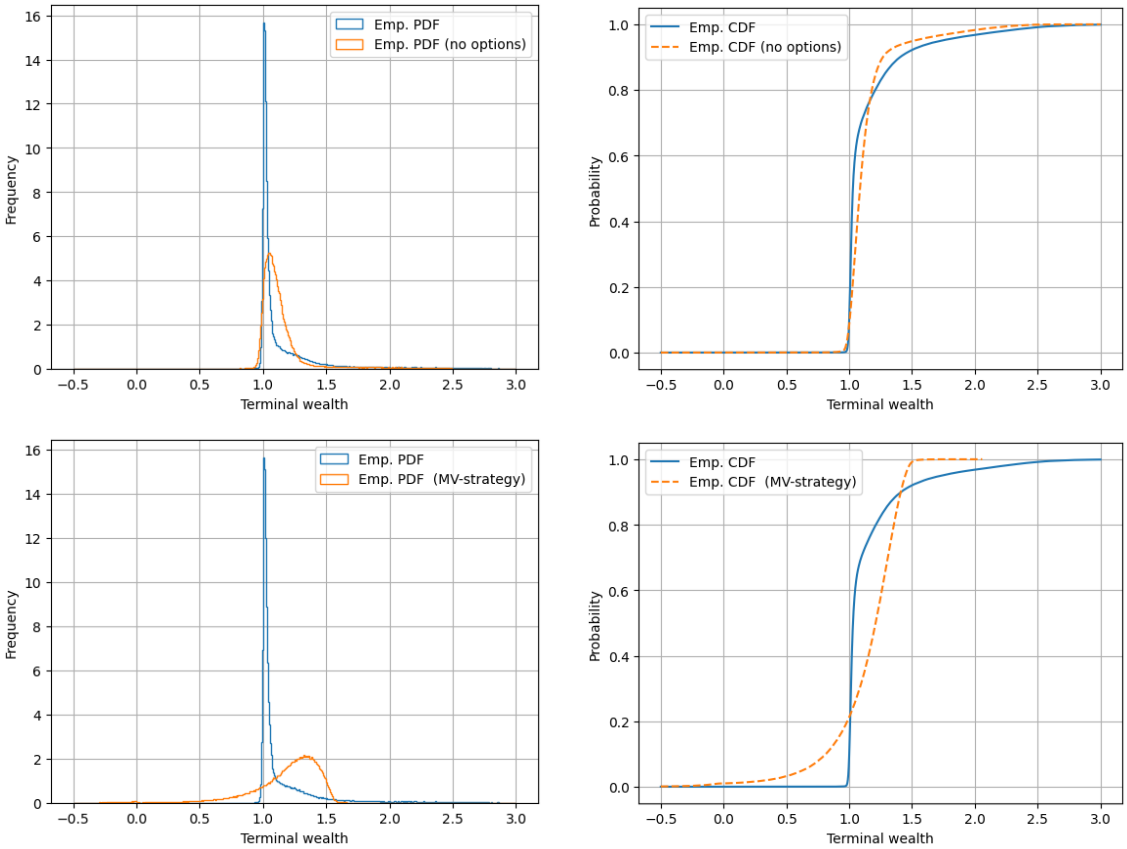

Figure 5.7: Comparison of empirical pdfs (left) and CDFs (right) for the D-TIPO and D-TIP strategies (top) and the MV-strategy (bottom).

	$\mathbb{E}[R]$	$\text{Var}[R]$	$\text{ES}^-_{p_1}(R)$	$\text{ES}^+_{p_2}(R)$	$U(\theta^*)$	Trading cost
Strategy with options	1.146	0.0806	0.971	2.176	1.609	0.386%
Strategy without options	1.140	0.0449	0.931	1.933	1.583	1.01%
Mean-variance strategy	1.146	0.0769	-0.208	1.479	1.209	3.98%

Table 5.2: Comparison of our three strategies in terms of empirical mean values, variances, expected shortfalls and the full loss function. Note that for the MV-strategy, the risk parameter λ, is set to make the mean coincide with the mean obtained from the strategy with options. The trading cost is calculated as a percentage of the initial wealth and reflects how volatile the portfolio re-allocations are.

beneficial not only from the perspective of lower transaction costs but also since less aggressive re-allocation is desirable for most investors.

5.4.2.2 Testing for robustness

In this chapter, we model the market with a jump-diffusion SDE, but as already stated, any model that can generate enough samples would fit the framework. On the other hand, whatever model is used, the only thing that we can be sure of is that real world market does not behave exactly as the model. Therefore, we aim to test the robustness of the algorithms for model miss-specification. This is carried out by applying the strategies from above with either significantly higher or significantly lower volatility of the underlying asset process. In the high volatility case, we multiply σ from (5.21) by two and in the low volatility case, we divide σ by two.

In Table 5.3, we see that the D-TIPO and D-TIP strategies significantly outperform the MV-strategy both with increased and decreased volatility. This is due to the fact that the volatility is a sensitive parameter in the closed-form strategy implying that if the volatility is either larger or smaller, then the strategy deviates significantly from optimality.[3] Most notable is the lower expected shortfall, which expresses a loss of 172.8% and 98.4% and the trading costs at 10.1% and 2.74% for the higher and lower volatilities, respectively. In the comparison between the strategies, it does not come as a surprise that options in the portfolio are particularly beneficial when the volatility increases and are less beneficial when the volatility decreases. Moreover, we note that the variance is larger for the strategy with options than without options, especially in the case with higher volatility. As described above, this is due to the fatter right tail of the distribution of terminal wealth and should not be viewed as a problem.

5.5 Conclusions

The main conclusions drawn from this chapter can be summarized in three parts: 1) It is important to be careful with the choice of objective function, in order to reflect the true incentives of a rational trader; 2) Adding options makes shaping the distribution of the terminal wealth more flexible, due to the asymmetric distribution of option returns. Moreover, adding options may significantly reduce the re-allocation and in turn the trading costs; 3) A

[3]Bear in mind that for the MV-strategy optimality is only in the sense of mean and variance since the expected shortfall terms do not enter the objective function.

	$\mathbb{E}[R]$	$\mathrm{Var}[R]$	$\mathrm{ES}^{-}_{p_1}(R)$	$\mathrm{ES}^{+}_{p_2}(R)$	$U(\theta^*)$	Trading cost
Evaluation with increased volatility for the underlying assets ($\sigma \mapsto 2 \times \sigma$).						
Strategy with options	1.318	0.660	0.763	3.268	1.540	0.838%
Strategy without options	1.350	0.175	0.734	2.635	1.532	1.23%
Mean-variance strategy	1.081	0.674	-0.728	1.460	0.668	10.1%
Evaluation with decreased volatility for the underlying assets ($\sigma \mapsto 0.5 \times \sigma$).						
Strategy with options	1.074	0.0262	0.969	1.620	1.506	0.261%
Strategy without options	1.143	0.0160	0.957	1.548	1.569	0.636%
Mean-variance strategy	1.163	0.0569	0.0160	1.487	1.301	2.74%

Table 5.3: Comparison of our three strategies in terms of empirical mean values, variances, expected shortfalls and the full loss function. Note that for the MV-strategy, the risk parameter λ, is set to make the mean coincide with the mean obtained from the D-TIPO strategy. The trading cost is calculated as a percentage of the initial wealth and reflects how volatile the portfolio re-allocations are.

sequence of neural networks can produce a high quality allocation strategy in high dimensions (many assets, options and also approximating strike prices for each option).

The extension to trading options in a dynamic setting is straight forward, as long as we have access to an efficient method for option valuation along stochastic asset trajectories. To approximate other options parameters, such as terminal time etc., is a straight forward extension.

6

Conclusions and outlook

The main topic of this thesis is the usage of neural networks to find approximate solutions to problems in stochastic control with applications in mathematical finance. For some problems, we considered a classical problem formulation and simply replaced the standard function approximator with a neural network. For instance, in Chapters 2 and 3, when pathwise option values were approximated, we relied on projections of conditional expectations on a finite-dimensional function space. A classical approach would be to use the function space constructed by a set of countable basis functions and then find the analytic expression for the minimizer of the L_2-risk, *i.e.,* to use a least-squares method. Our approach was instead to consider the same problem formulation, but to optimize over the set of neural network parameters. This implied that the function space we were searching in was constructed from the neural network structure. In this way, we could gain flexibility, since this space is richer. On the other hand, we lost some explainability and introduced an additional source of error since our problem does not offer a closed-form optimal solution, but instead has to rely on numerical optimization. The aim was that the richer function space could compensate for the added optimization error.

For other problems, we made use of mathematical reformulations to arrive at a problem formulation specifically tailored to neural networks. More precisely, the reformulations resulted in a loss function, which, if minimized, also solved the original stochastic control problem. The loss functions considered in this thesis usually had too complex structures to be solved analytically or even with classical numerical optimization schemes. In those scenarios, we relied on the stochastic gradient descent optimization machinery which is offered by modern software packages, such as Tensorflow and Pytorch. For this type of problems, our main contribution was a novel reformulation, which, with the help of modern machine learning tools, simplified the original problem.

Finally, for the most complex problems considered, classical methods could not be implemented or at least did not have a realistic chance of being successful. In situations like this, neural networks and their optimization procedures are often the last resort and their capability truly stood out.

In this section, we walk through the thesis chapter by chapter, and describe the impact of using neural networks as an approximation method. In addition, we explain some interesting lines of future research.

Chapters 2-3

In Chapters 2 and 3, part of the task was to find the fair value for high-dimensional Bermudan options. The accuracy obtained by the neural network-based method was shown to be fully comparable with an (almost) analytic PDE solver and significantly better than state of the art Monte-Carlo methods. In high dimensions, the PDE solver is no longer a feasible alternative and even though the Monte-Carlo method is still implementable dimensionality hampers both accuracy and practicability. These problems did not seem to be present when using the neural network-based methods (even without a reference solution it was possible to compare the accuracy since all tested methods were biased low, implying that, the higher the approximate value, the better the approximation). It is well known that neural networks often perform relatively well on problems that are not feasible with classical methods, but in these chapters, we also showed that the accuracy was even sometimes better than conventional methods, even in low to moderate dimensions.

A disadvantage of the least squares Monte–Carlo method is that, in order to chose an appropriate set of basis functions, one often requires some apriori knowledge of the problem solution. In general, for high dimensional function surfaces, this is not the case, and we are left with a cumbersome trial and error procedure.

In Chapter 3, the algorithm was extended to act on the level of a portfolio of options, *i.e.,* the stopping decisions as well as the valuation were performed for multiple high-dimensional options simultaneously. The double high-dimensionality in this problem really put the framework to an extreme test, but the performance was shown to be comparable to the performance for less complex problems.

In Chapters 2-3, the main objective was to compute pathwise option values of Bermudan or American options. Even though this is a challenging task on its own, the final goal is usually to use these values to compute X-valuation adjustments (xVA). In Chapter 3, we computed credit valuation adjustments (CVA), the valuation adjustment that is used to take the credit risk of the counterpart into account. Although this is one of the most common valuation adjustments, a formalized framework to compute other xVAs would be an interesting and straight-forward extension.

Chapter 4

In this Chapter, a stochastic control problem was reformulated into a FBSDE, which was then approximated with the help of a sequence of neural networks. The loss function used a combination of properties from the stochastic control problem and the associated FBSDE. We demonstrated that the proposed method was both accurate and robust when applied to problems that classical methods as well as similar machine learning methods fail to converge. For instance, a direct extension of the deep BSDE method to FBSDEs fails to converge for low dimensional Linear-Quadratic Gaussian control problems. The main conclusion from this chapter was however not about the capability of machine learning. What should instead be

noticed is that the seemingly small adjustments to the problem formulation (compared to the deep BSDE method) changed the outcome drastically. From not converging, to being able to robustly solve high-dimensional problems as well as problems with significantly more complex dynamics. One conclusion drawn from this is that even though neural networks can be thrown at many problems and perform well, it is still important to use as much problem specific knowledge as possible.

The problem considered in this chapter was control of a SDE with a cost functional of standard type for which the DPP holds. The SDE took the control process as input in the drift but not in the diffusion coefficient. However, there are many interesting applications in which also control in the diffusion coefficient is a natural way to model the problem. One important example is portfolio optimization in the presence of market impact. This, slightly more general problem formulation yields an associate FBSDE of second order (2FBSDE), in which the sought solution is a quadrupel, (X, Y, Z, Γ), instead of the triple, (X, Y, Z), we search for when solving first-order FBSDEs. In principle, the same neural network structure and loss function could be used for such a more general problem. It is however likely that one could use the extended problem formulation with known mathematical properties from the Γ-process to construct an improved loss function.

Although the generalization discussed above is the most straight-forward extension to our method, there are several other interesting future lines of research. Arguably, the most important generalization of our deep FBSDE method would be an extension so that it could be applicable for all kinds of FBSDEs, not only the ones stemming from stochastic control. With the current problem formulation, this would not be possible since a crucial term in the loss function is directly connected to stochastic control problems. Therefore, in order to achieve this extension, one would have to find a similar penalty term for general FBSDEs. Exactly how this term would look is not obvious, but it is surely an interesting as well as important line of research.

Chapter 5

In Chapters 2-4, all problem formulations were well-established with nice mathematical properties. In Chapter 5, on the other hand, the problems were formulated from a realistic economic perspective. One consequence of this procedure was that much of the mathematical foundation enjoyed for standard formulations collapsed. For instance, we imposed various trading constraints, transaction costs and we did not assume a complete market. Moreover, we allowed for trading in a bond, stocks and options with a general objective function. All the above combined led to a problem in which the existing mathematical theory was limited and we could generally not guarantee existence or uniqueness of solutions. Although these are important issues for mathematicians, especially if one wants to formulate a dual problem, they are of relatively minor importance to an investor. For instance, the important task for an investor is to follow a trading strategy which yields the best possible risk-reward properties according to some pre-determined objective. Usually, it does not matter whether there are other strategies that perform equally well. With this in mind, it seemed plausible to simply adopt a straight-forward approach to replace the allocation strategy at each time point with a neural network and to use an empirical version of the objective function as loss function. From

the numerical results, it became evident that the performance of the neural network-based algorithm, D-TIPO, was more than acceptable. It offers practitioners the possibility of relaxing many of the restrictive constraints that historically had to be incorporated in a trading strategy.

In the numerical results section of Chapter 5, we only considered synthetic asset data in the shape of a jump-diffusion SDE. Even though, this is an interesting starting point for our methodology, it would be highly interesting to input real world historical data into our algorithm. This is possible due to the fact that the D-TIPO algorithm is model independent and fully data-driven. This means that if we would have enough historical data (and we trust that this data is representative) we could implement our method without any assumptions on an underlying model. Even though this is clear in principle, it comes with some challenges. First of all, with synthetic data we can sample as much as needed which constitutes a significant difference when compared to the single time series of historical data available per asset. Furthermore, a viable question to ask is what the historical distribution of returns says about the future counterparts? If a time series could be considered stationary in time, then one could find clever ways to split up a long time series into many shorter ones. There are many interesting challenges associated with implementing our method with historical data, but given its properties of being completely data-driven, it is clearly a compelling future line of research.

References

[1] Kristoffer Andersson and Cornelis Oosterlee. "A deep learning approach for computations of exposure profiles for high-dimensional Bermudan options". In: *Applied Mathematics and Computation* 408 (2021), p. 126332.

[2] Kristoffer Andersson and Cornelis W Oosterlee. "Deep learning for CVA computations of large portfolios of financial derivatives". In: *Applied Mathematics and Computation* 409 (2021), p. 126399.

[3] Kristoffer Andersson, Adam Andersson, and Cornelis W Oosterlee. "Convergence of a robust deep FBSDE method for stochastic control". In: *SIAM Journal on Scientific Computing* 45.1 (2023), A226–A255.

[4] Jiequn Han, Arnulf Jentzen, and E Weinan. "Solving high-dimensional partial differential equations using deep learning". In: *Proceedings of the National Academy of Sciences* 115.34 (2018), pp. 8505–8510.

[5] Kristoffer Andersson and Cornelis W Oosterlee. "D-TIPO: Deep time-inconsistent portfolio optimization with stocks and options". In: *arXiv preprint arXiv:2308.10556* (2023).

[6] Jiequn Han and E Weinan. "Deep Learning Approximation for Stochastic Control Problems". In: *ArXiv* abs/1611.07422 (2016). URL: `https://api.semanticscholar.org/CorpusID:16386889`.

[7] Xun Yu Zhou and Duan Li. "Continuous-time mean-variance portfolio selection: A stochastic LQ framework". In: *Applied Mathematics and Optimization* 42 (2000), pp. 19–33.

[8] René Carmona and François Delarue. *Probabilistic theory of mean field games with applications I-II*. Springer, 2018.

[9] Sebastian Becker, Patrick Cheridito, and Arnulf Jentzen. "Deep Optimal Stopping". In: *Journal of Machine Learning Research* 20 (2019).

[10] John Gregory. *The xVA Challenge: counterparty credit risk, funding, collateral and capital*. John Wiley & Sons, 2015.

[11] Andrew Green. *XVA: credit, funding and capital valuation adjustments*. John Wiley & Sons, 2015.

[12] Peter Forsyth and Kenneth Vetzal. "Quadratic convergence for valuing American options using a penalty method". In: *SIAM Journal on Scientific Computing* 23.6 (2002), pp. 2095–2122.

[13] Christoph Reisinger and Jan Hendrik Witte. "On the use of policy iteration as an easy way of pricing American options". In: *SIAM Journal on Financial Mathematics* 3.1 (2012), pp. 459–478.

[14] Carlos Vázquez. "An upwind numerical approach for an American and European option pricing model". In: *Applied Mathematics and Computation* 97.2-3 (1998), pp. 273–286.

[15] Tinne Haentjens and Karel in't Hout. "ADI schemes for pricing American options under the Heston model". In: *Applied Mathematical Finance* 22.3 (2015), pp. 207–237.

[16] Karel in't Hout. *Numerical Partial Differential Equations in Finance Explained; An Introduction to Computational Finance*. Palgrave Macmillan UK, 2017.

[17] Fang Fang and Cornelis Oosterlee. "Pricing early-exercise and discrete barrier options by Fourier-cosine series expansions". In: *Numerische Mathematik* 114.1 (2009), p. 27.

[18] Oleksandr Zhylyevskyy. "A fast Fourier transform technique for pricing American options under stochastic volatility". In: *Review of Derivatives Research* 13.1 (2010), pp. 1–24.

[19] Fang Fang and Cornelis Oosterlee. "A Fourier-based valuation method for Bermudan and barrier options under Heston's model". In: *SIAM Journal on Financial Mathematics* 2.1 (2011), pp. 439–463.

[20] Mark Broadie and Jerome Detemple. "American option valuation: new bounds, approximations, and a comparison of existing methods". In: *The Review of Financial Studies* 9.4 (1996), pp. 1211–1250.

[21] Mark Rubinstein. "Edgeworth binomial trees". In: *Journal of Derivatives* 5 (1998), pp. 20–27.

[22] Jens Jackwerth. "Option-implied risk-neutral distributions and implied binomial trees: A literature review". In: *The Journal of Derivatives* 7.2 (1999), pp. 66–82.

[23] Richard Bellman. "Dynamic programming". In: *Science* 153.3731 (1966), pp. 34–37.

[24] Leif Andersen and Mark Broadie. "Primal-dual simulation algorithm for pricing multi-dimensional American options". In: *Management Science* 50.9 (2004), pp. 1222–1234.

[25] Francis Longstaff and Eduardo Schwartz. "Valuing American options by simulation: a simple least-squares approach". In: *The review of financial studies* 14.1 (2001), pp. 113–147.

[26] Mark Broadie and Paul Glasserman. "A stochastic mesh method for pricing high-dimensional American options". In: *Journal of Computational Finance* 7 (2004), pp. 35–72.

[27] Mark Broadie and Menghui Cao. "Improved lower and upper bound algorithms for pricing American options by simulation". In: *Quantitative Finance* 8.8 (2008), pp. 845–861.

[28] Shashi Jain and Cornelis Oosterlee. "The stochastic grid bundling method: Efficient pricing of Bermudan options and their Greeks". In: *Applied Mathematics and Computation* 269 (2015), pp. 412–431.

[29] Michael Kohler, Adam Krzyżak, and Nebojsa Todorovic. "Pricing of High-Dimensional American Options by Neural Networks". In: *Mathematical Finance: An International Journal of Mathematics, Statistics and Financial Economics* 20.3 (2010), pp. 383–410.

[30] Sebastian Becker, Patrick Cheridito, and Arnulf Jentzen. "Pricing and hedging American-style options with deep learning". In: *Journal of Risk and Financial Management* 13.7 (2020), p. 158.

[31] Bernard Lapeyre and Jérôme Lelong. "Neural network regression for Bermudan option pricing". In: *Monte Carlo Methods and Applications* 27.3 (2021), pp. 227–247.

[32] Vikranth Lokeshwar, Vikram Bhardawaj, and Shashi Jain. "Neural network for pricing and universal static hedging of contingent claims". In: *Available at SSRN 3491209* (2019).

[33] Andrew David Green, Chris Kenyon, and Chris Dennis. "KVA: Capital valuation adjustment". In: *Risk, December* (2014).

[34] Shashi Jain, Patrik Karlsson, and Drona Kandhai. "KVA, Mind Your P's and Q's!" In: *Wilmott* 2019.102 (2019), pp. 60–73.

[35] Cornelis De Graaf et al. "Efficient computation of exposure profiles for counterparty credit risk". In: *International Journal of Theoretical and Applied Finance* 17.04 (2014), p. 1450024.

[36] Yanbin Shen, Johannes AM Van Der Weide, and Jasper HM Anderluh. "A benchmark approach of counterparty credit exposure of Bermudan option under Lévy Process: the Monte Carlo-COS Method". In: *Procedia Computer Science* 18 (2013), pp. 1163–1171.

[37] Qian Feng et al. "Efficient computation of exposure profiles on real-world and risk-neutral scenarios for Bermudan swaptions". In: *Available at SSRN 2790874* (2016).

[38] Roberto Baviera, Gaetano La Bua, and Paolo Pellicioli. "CVA with wrong-way risk in the presence of early exercise". In: *Innovations in Derivatives Markets*. Springer, Cham, 2016, pp. 103–116.

[39] Michèle Breton and Oussama Marzouk. *An efficient method to price counterparty risk*. Groupe d'études et de recherche en analyse des décisions, 2014.

[40] Bernt Oksendal. *Stochastic differential equations: an introduction with applications*. Springer Science & Business Media, 2013.

[41] Chigozie Nwankpa et al. "Activation functions: Comparison of trends in practice and research for deep learning". In: *Preprint arXiv:1811.03378 Comment: Published as a conference paper at the 2nd International Conference on Computational Sciences and Technologies, 17-19 Dec 2020* (2018).

[42] Diederik Kingma and Jimmy Ba. "Adam: A method for stochastic optimization". In: *arXiv preprint arXiv:1412.6980. ISO 690. Comment: Published as a conference paper at the 3rd International Conference for Learning Representations, San Diego* (2015).

[43] Mark Broadie and Jérôme Detemple. "The valuation of American options on multiple assets". In: *Mathematical Finance* 7.3 (1997), pp. 241–286.

[44] A Max Reppen, H Mete Soner, and Valentin Tissot-Daguette. "Neural optimal stopping boundary". In: *arXiv preprint arXiv:2205.04595* (2022).

[45] László Györfi et al. *A distribution-free theory of nonparametric regression*. Springer Science & Business Media, 2006.

[46] Halbert White. *Asymptotic theory for econometricians*. Academic press, 2014.

[47] Paul Glasserman. *Monte Carlo methods in financial engineering*. Vol. 53. Springer Science & Business Media, 2013.

[48] Iñigo Arregui et al. "PDE models for American options with counterparty risk and two stochastic factors: Mathematical analysis and numerical solution". In: *Computers & Mathematics with Applications* (2019).

[49] Steven Heston. "A closed-form solution for options with stochastic volatility with applications to bond and currency options". In: *The review of financial studies* 6.2 (1993), pp. 327–343.

[50] Leif Andersen and Vladimir Piterbarg. "Moment explosions in stochastic volatility models". In: *Finance and Stochastics* 11.1 (2007), pp. 29–50.

[51] John Cox, Jonathan Ingersoll, and Stephen Ross. "A theory of the term structure of interest rates". In: *Theory of valuation*. World Scientific, 2005, pp. 129–164.

[52] William Feller. "Two singular diffusion problems". In: *Annals of mathematics* (1951), pp. 173–182.

[53] Leif Andersen. "Efficient simulation of the Heston stochastic volatility model". In: *Available at SSRN 946405* (2007).

[54] Marjon Ruijter and Cornelis Oosterlee. "Two-dimensional Fourier cosine series expansion method for pricing financial options". In: *SIAM Journal on Scientific Computing* 34.5 (2012), B642–B671.

[55] John Hull and Alan White. "The impact of default risk on the prices of options and other derivative securities". In: *Journal of Banking & Finance* 19.2 (1995), pp. 299–322.

[56] Richard Zhou. "Back to CVA: the case of American option". In: *Available at SSRN 3189805* (2019).

[57] "Basel III: A global regulatory framework for more resilient banks and banking systems." In: *Available at: https://www.bis.org* (2010).

[58] Damiano Brigo, Massimo Morini, and Andrea Pallavicini. *Counterparty credit risk, collateral and funding: with pricing cases for all asset classes*. Vol. 478. John Wiley & Sons, 2013.

[59] Qian Feng and Cornelis W Oosterlee. "Wrong Way Risk Modeling and Computation in Credit Valuation Adjustment for European and Bermudan Options". In: *Available at SSRN 2852819* (2016).

[60] Long Teng. "A review of tree-based approaches to solve forward-backward stochastic differential equations". In: *The Journal of Computational Finance* (2021).

[61] Jin Ma et al. "Numerical method for backward stochastic differential equations". In: *The Annals of Applied Probability* 12.1 (2002), pp. 302–316.

[62] Marjon J Ruijter and Cornelis W Oosterlee. "A *F*ourier cosine method for an efficient computation of solutions to BSDEs". In: *SIAM J. Sci. Computing* 37.2 (2015), A859–A889.

[63] TP Huijskens, Maria J Ruijter, and Cornelis W Oosterlee. "Efficient numerical *F*ourier methods for coupled forward–backward *SDE*s". In: *Journal of Computational and Applied Mathematics* 296 (2016), pp. 593–612.

[64] Marjon J Ruijter and Cornelis W Oosterlee. "Numerical *F*ourier method and second-order Taylor scheme for backward *SDE*s in finance". In: *Applied Numerical Mathematics* 103 (2016), pp. 1–26.

[65] Bruno Bouchard and Nizar Touzi. "Discrete-time approximation and *M*onte-*C*arlo simulation of backward stochastic differential equations". In: *Stochastic Processes and their applications* 111.2 (2004), pp. 175–206.

[66] Ki Wai Chau and Cornelis W Oosterlee. "Stochastic grid bundling method for backward stochastic differential equations". In: *International Journal of Computer Mathematics* 96.11 (2019), pp. 2272–2301.

[67] Arash Fahim, Nizar Touzi, and Xavier Warin. "A probabilistic numerical method for fully nonlinear parabolic *PDE*s". In: *The Annals of Applied Probability* 21.4 (2011), pp. 1322–1364.

[68] Christian Bender and Jessica Steiner. "Least-squares *M*onte *C*arlo for backward *SDE*s". In: *Numerical methods in finance*. Springer, 2012, pp. 257–289.

[69] Dan Crisan, Konstantinos Manolarakis, and Nizar Touzi. "On the *M*onte *C*arlo simulation of *BSDE*s: *A*n improvement on the *M*alliavin weights". In: *Stochastic Processes and their Applications* 120.7 (2010), pp. 1133–1158.

[70] Emmanuel Gobet et al. "Stratified regression *M*onte-*C*arlo scheme for semilinear *PDE*s and *BSDE*s with large scale parallelization on *GPU*s". In: *SIAM J. Sci. Computing* 38.6 (2016), pp. C652–C677.

[71] Jin Ma, Philip Protter, and Jiongmin Yong. "Solving forward-backward stochastic differential equations explicitly - *A* four step scheme". In: *Probability theory and related fields* 98.3 (1994), pp. 339–359.

[72] Emmanuel Gobet and Céline Labart. "Error expansion for the discretization of backward stochastic differential equations". In: *Stochastic processes and their applications* 117.7 (2007), pp. 803–829.

[73] Emmanuel Gobet and Céline Labart. "Solving *BSDE* with adaptive control variate". In: *SIAM J. Numerical Analysis* 48.1 (2010), pp. 257–277.

[74] Martin Hutzenthaler, Arnulf Jentzen, and Thomas Kruse. "On multilevel *P*icard numerical approximations for high-dimensional nonlinear parabolic partial differential equations and high-dimensional nonlinear backward stochastic differential equations". In: *Journal of Scientific Computing* 79.3 (2019), pp. 1534–1571.

[75] Christian Bender and Robert Denk. "A forward scheme for backward *SDE*s". In: *Stochastic processes and their applications* 117.12 (2007), pp. 1793–1812.

[76] Jared Chessari et al. “Numerical methods for backward stochastic differential equations: A survey”. In: *Probability Surveys* 20 (2023), pp. 486–567.

[77] Christian Beck et al. “Solving the *K*olmogorov *PDE* by means of deep learning”. In: *Journal of Scientific Computing* 88.3 (2021), pp. 1–28.

[78] Pierre Henry-Labordere. “Deep primal-dual algorithm for *BSDE*s: Applications of machine learning to *CVA* and *IM*”. In: *Available at SSRN 3071506* (2017).

[79] Christian Beck, Arnulf Jentzen, et al. “Machine learning approximation algorithms for high-dimensional fully nonlinear partial differential equations and second-order backward stochastic differential equations”. In: *Journal of Nonlinear Science* 29.4 (2019), pp. 1563–1619.

[80] Christian Beck et al. “Deep splitting method for parabolic *PDE*s”. In: *SIAM J. Sci. Computing* 43.5 (2021), A3135–A3154.

[81] Masaaki Fujii, Akihiko Takahashi, and Masayuki Takahashi. “Asymptotic expansion as prior knowledge in deep learning method for high dimensional *BSDE*s”. In: *Asia-Pacific Financial Markets* 26.3 (2019), pp. 391–408.

[82] Maziar Raissi. “Forward-backward stochastic neural networks: *D*eep learning of high-dimensional partial differential equations”. In: *Preprint arXiv:1804.07010* (2018).

[83] Shaolin Ji et al. “Three algorithms for solving high-dimensional fully coupled *FBSDE*s through deep learning”. In: *IEEE Intelligent Systems* 35.3 (2020), pp. 71–84.

[84] Alessandro Gnoatto, Athena Picarelli, and Christoph Reisinger. “Deep xVA Solver: A Neural Network–Based Counterparty Credit Risk Management Framework”. In: *SIAM Journal on Financial Mathematics* 14.1 (2023), pp. 314–352.

[85] Shaolin Ji et al. “A control method for solving high-dimensional Hamiltonian systems through deep neural networks”. In: *Preprint arXiv:2111.02636* (2021).

[86] Yutian Wang and Yuan-Hua Ni. “Deep *BSDE-ML L*earning and *I*ts *A*pplication to *M*odel-*F*ree *O*ptimal *C*ontrol”. In: *Preprint arXiv:2201.01318* (2022).

[87] Jiequn Han and Jihao Long. “Convergence of the deep *BSDE* method for coupled *FBSDE*s”. In: *Probability, Uncertainty and Quantitative Risk* 5.1 (2020), pp. 1–33.

[88] Martin Hutzenthaler et al. “A proof that rectified deep neural networks overcome the curse of dimensionality in the numerical approximation of semilinear heat equations”. In: *SN partial differential equations and applications* 1.2 (2020), pp. 1–34.

[89] Julius Berner, Philipp Grohs, and Arnulf Jentzen. “Analysis of the generalization error: *E*mpirical risk minimization over deep artificial neural networks overcomes the curse of dimensionality in the numerical approximation of Black–Scholes partial differential equations”. In: *SIAM J. Mathematics of Data Science* 2.3 (2020), pp. 631–657.

[90] Dennis Elbrächter et al. “D*NN* expression rate analysis of high-dimensional *PDE*s: Application to option pricing”. In: *Constructive Approximation* (2021), pp. 1–69.

[91] Philipp Grohs et al. “A proof that artificial neural networks overcome the curse of dimensionality in the numerical approximation of *B*lack–*S*choles partial differential equations”. In: *ArXiv* abs/1809.02362 (2018).

[92] Arnulf Jentzen, Diyora Salimova, and Timo Welti. "A proof that deep artificial neural networks overcome the curse of dimensionality in the numerical approximation of *K*olmogorov partial differential equations with constant diffusion and nonlinear drift coefficients". In: *Preprint arXiv:1809.07321* (2018).

[93] Yifan Jiang and Jinfeng Li. "Convergence of the Deep *BSDE* method for *FBSDE*s with non-*L*ipschitz coefficients". In: *Preprint arXiv:2101.01869* (2021).

[94] Côme Huré, Huyên Pham, and Xavier Warin. "Deep backward schemes for high-dimensional nonlinear *PDE*s". In: *Mathematics of Computation* 89.324 (2020), pp. 1547–1579.

[95] Côme Huré, Huyên Pham, and Xavier Warin. "Some machine learning schemes for high-dimensional nonlinear *PDE*s". In: *Preprint arXiv:1902.01599* 33 (2019).

[96] Quentin Chan-Wai-Nam, Joseph Mikael, and Xavier Warin. "Machine learning for semi linear *PDE*s". In: *Journal of Scientific Computing* 79.3 (2019), pp. 1667–1712.

[97] Fang Fang and Cornelis W Oosterlee. "A novel pricing method for European options based on Fourier-cosine series expansions". In: *SIAM J. Sci. Computing* 31.2 (2009), pp. 826–848.

[98] Balint Negyesi, Kristoffer Andersson, and Cornelis W. Oosterlee. "The *O*ne *S*tep *M*alliavin scheme: new discretization of *BSDE*s implemented with deep learning regressions". In: *Preprint arXiv:2110.05421* (2021).

[99] Christian Beck et al. "An overview on deep learning-based approximation methods for partial differential equations". In: *Preprint arXiv:2012.12348* (2020).

[100] Kristoffer Andersson. "Approximate stochastic control based on deep learning and forward backward stochastic differential equations". MA thesis. 2019.

[101] Marcus Pereira et al. "Learning deep stochastic optimal control policies using forward-backward *SDE*s". In: *Preprint arXiv:1902.03986* (2019).

[102] Yujun Liu et al. "Deep *FBSDE C*ontroller for *A*ttitude *C*ontrol of *H*ypersonic *A*ircraft". In: *2021 6th IEEE Int. Conf. on Advanced Robotics and Mechatronics (ICARM)*. IEEE. 2021, pp. 294–299.

[103] Bolun Dai et al. "Learning *L*ocomotion *C*ontrollers for *W*alking *U*sing *D*eep *FBSDE*". In: *Preprint arXiv:2107.07931* (2021).

[104] Christoph Reisinger, Wolfgang Stockinger, and Yufei Zhang. "A posteriori error estimates for fully coupled McKean-Vlasov forward-backward SDEs". In: *arXiv preprint arXiv:2007.07731* (2020).

[105] Christoph Reisinger and Yufei Zhang. "Rectified deep neural networks overcome the curse of dimensionality for nonsmooth value functions in zero-sum games of nonlinear stiff systems". In: *Analysis and Applications* 18.06 (2020), pp. 951–999.

[106] Christoph Reisinger, Wolfgang Stockinger, and Yufei Zhang. "Linear convergence of a policy gradient method for finite horizon continuous time stochastic control problems". In: *arXiv e-prints* (2022), arXiv–2203.

[107] Michael Giegrich, Christoph Reisinger, and Yufei Zhang. "Convergence of policy gradient methods for finite-horizon stochastic linear-quadratic control problems". In: *arXiv preprint arXiv:2211.00617* (2022).

[108] Christoph Reisinger, Wolfgang Stockinger, and Yufei Zhang. "A fast iterative PDE-based algorithm for feedback controls of nonsmooth mean-field control problems". In: *arXiv preprint arXiv:2108.06740* (2021).

[109] Fabio Antonelli. "Backward forward stochastic differential equations". PhD thesis. Purdue University, 1993.

[110] Yajie Yu, Narayan Ganesan, and Bernhard Hientzsch. "Backward deep BSDE methods and applications to nonlinear problems". In: *Risks* 11.3 (2023), p. 61.

[111] George Cybenko. "Approximations by superpositions of a sigmoidal function". In: *Mathematics of Control, Signals and Systems* 2 (1989), pp. 183–192.

[112] Raphael Kruse. "Characterization of bistability for stochastic multistep methods". In: *BIT Num. Math.* 52.1 (2012), pp. 109–140.

[113] Wolf-Jürgen Beyn and Raphael Kruse. "Two-sided error estimates for the stochastic theta method". In: *Discrete & Continuous Dynamical Systems-B* 14.2 (2010), p. 389.

[114] Remigius Mikulevicius and Eckhard Plate. "Rate of convergence of the Euler approximation for diffusion processes". In: *Mathematische Nachrichten* 151.1 (1991), pp. 233–239.

[115] Arturo Kohatsu-Higa, Antoine Lejay, and Kazuhiro Yasuda. "Weak rate of convergence of the *E*uler–*M*aruyama scheme for stochastic differential equations with non-regular drift". In: *Journal of Computational and Applied Mathematics* 326 (2017), pp. 138–158.

[116] Giuseppe Buttazzo and Gianni Dal Maso. "Γ-convergence and optimal control problems". In: *Journal of optimization theory and applications* 38.3 (1982), pp. 385–407.

[117] R Tyrrell Rockafellar and Roger J-B Wets. *Variational analysis*. Vol. 317. Springer Science & Business, 2009.

[118] Joseph Frédéric Bonnans, Justina Gianatti, and Francisco J Silva. "On the time discretization of stochastic optimal control problems: the dynamic programming approach". In: *ESAIM: Control, Optimis. Calculus of Variations* 25 (2019), p. 63.

[119] Diederik P Kingma and Jimmy Ba. "Adam: A method for stochastic optimization". In: *Preprint arXiv:1412.6980* (2014).

[120] Karl J Åström. *Introduction to stochastic control theory*. Courier Corporation, 2012.

[121] Lorenc Kapllani and Long Teng. "Deep Learning algorithms for solving high dimensional nonlinear Backward Stochastic Differential Equations". In: *arXiv preprint arXiv:2010.01319* (2020).

[122] HM Markowitz. "Portfolio Selection, the journal of finance. 7 (1)". In: *N* 1 (1952), pp. 71–91.

[123] Fei Cong and Cornelis W Oosterlee. "Multi-period mean–variance portfolio optimization based on Monte-Carlo simulation". In: *Journal of Economic Dynamics and Control* 64 (2016), pp. 23–38.

[124] F Cong and CW Oosterlee. "On robust multi-period pre-commitment and time-consistent mean-variance portfolio optimization". In: *International Journal of Theoretical and Applied Finance* 20.07 (2017), p. 1750049.

[125] Peter A Forsyth and Kenneth R Vetzal. "Multi-Period Mean Expected-Shortfall Strategies:'Cut Your Losses and Ride Your Gains'". In: *Applied Mathematical Finance* (2022), pp. 1–37.

[126] Giuseppe Carlo Calafiore. "Multi-period portfolio optimization with linear control policies". In: *Automatica* 44.10 (2008), pp. 2463–2473.

[127] Pieter M van Staden, Duy-Minh Dang, and Peter A Forsyth. "The surprising robustness of dynamic mean-variance portfolio optimization to model misspecification errors". In: *European Journal of Operational Research* 289.2 (2021), pp. 774–792.

[128] Suleyman Basak and Georgy Chabakauri. "Dynamic mean-variance asset allocation". In: *The Review of Financial Studies* 23.8 (2010), pp. 2970–3016.

[129] Min Dai et al. "A dynamic mean-variance analysis for log returns". In: *Management Science* 67.2 (2021), pp. 1093–1108.

[130] R Tyrrell Rockafellar, Stanislav Uryasev, et al. "Optimization of conditional value-at-risk". In: *Journal of risk* 2 (2000), pp. 21–42.

[131] Tomas Bjork and Agatha Murgoci. "A general theory of Markovian time inconsistent stochastic control problems". In: *Available at SSRN 1694759* (2010).

[132] Peter A Forsyth. "Multiperiod mean Conditional Value at Risk asset allocation: Is it advantageous to be time consistent?" In: *SIAM Journal on Financial Mathematics* 11.2 (2020), pp. 358–384.

[133] Elena Vigna. "Tail optimality and preferences consistency for intertemporal optimization problems". In: *SIAM Journal on Financial Mathematics* 13.1 (2022), pp. 295–320.

[134] Elena Vigna. "On time consistency for mean-variance portfolio selection". In: *International Journal of Theoretical and Applied Finance* 23.06 (2020), p. 2050042.

[135] Pieter M van Staden, Duy-Minh Dang, and Peter A Forsyth. "On the distribution of terminal wealth under dynamic mean-variance optimal investment strategies". In: *SIAM Journal on Financial Mathematics* 12.2 (2021), pp. 566–603.

[136] ST Tse et al. "Comparison between the mean-variance optimal and the mean-quadratic-variation optimal trading strategies". In: *Applied Mathematical Finance* 20.5 (2013), pp. 415–449.

[137] Tomas Björk, Mariana Khapko, and Agatha Murgoci. "On time-inconsistent stochastic control in continuous time". In: *Finance and Stochastics* 21 (2017), pp. 331–360.

[138] Yuying Li and Peter A Forsyth. "A data-driven neural network approach to optimal asset allocation for target based defined contribution pension plans". In: *Insurance: Mathematics and Economics* 86 (2019), pp. 189–204.

[139] Pieter M van Staden, Peter A Forsyth, and Yuying Li. "A parsimonious neural network approach to solve portfolio optimization problems without using dynamic programming". In: *arXiv preprint arXiv:2303.08968* (2023).

[140] Sebastian Jaimungal. "Reinforcement learning and stochastic optimisation". In: *Finance and Stochastics* 26.1 (2022), pp. 103–129.

[141] Volodymyr Mnih et al. "Human-level control through deep reinforcement learning". In: *nature* 518.7540 (2015), pp. 529–533.

[142] Yan Duan et al. "Benchmarking deep reinforcement learning for continuous control". In: *International conference on machine learning*. PMLR. 2016, pp. 1329–1338.

[143] Biao Luo and Huai-Ning Wu. "Approximate optimal control design for nonlinear one-dimensional parabolic PDE systems using empirical eigenfunctions and neural network". In: *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)* 42.6 (2012), pp. 1538–1549.

[144] Yann LeCun et al. "Gradient-based learning applied to document recognition". In: *Proceedings of the IEEE* 86.11 (1998), pp. 2278–2324.

[145] Yoshua Bengio et al. "Learning deep architectures for AI". In: *Foundations and trends® in Machine Learning* 2.1 (2009), pp. 1–127.

[146] Warren B Powell. *Approximate Dynamic Programming: Solving the curses of dimensionality*. Vol. 703. John Wiley & Sons, 2007.

[147] Richard S Sutton, Andrew G Barto, et al. "Reinforcement learning". In: *Journal of Cognitive Neuroscience* 11.1 (1999), pp. 126–134.

[148] David Silver et al. "Mastering the game of Go with deep neural networks and tree search". In: *nature* 529.7587 (2016), pp. 484–489.

[149] John Schulman et al. "High-dimensional continuous control using generalized advantage estimation". In: *arXiv preprint arXiv:1506.02438* (2015).

[150] John Schulman et al. "Trust region policy optimization". In: *International conference on machine learning*. PMLR. 2015, pp. 1889–1897.

[151] Vigdis Boasson, Emil Boasson, and Zhao Zhou. "Portfolio optimization in a mean-semivariance framework". In: *Investment management and financial innovations* 8, Iss. 3 (2011), pp. 58–68.

[152] Luís Lobato Macedo, Pedro Godinho, and Maria João Alves. "Mean-semivariance portfolio optimization with multiobjective evolutionary algorithms and technical analysis rules". In: *Expert Systems with Applications* 79 (2017), pp. 33–43.

[153] Stefano Ciliberti, Imre Kondor, and Marc Mézard. "On the feasibility of portfolio optimization under expected shortfall". In: *Quantitative Finance* 7.4 (2007), pp. 389–396.

[154] Yasuhiro Yamai and Toshinao Yoshiba. "Value-at-risk versus expected shortfall: A practical perspective". In: *Journal of Banking & Finance* 29.4 (2005), pp. 997–1015.

[155] Jonathan Raimana Chan et al. "Portfolio optimisation with options". In: *arXiv preprint arXiv:2111.12658* (2021).

[156] Joshua D Coval and Tyler Shumway. "Expected option returns". In: *The journal of Finance* 56.3 (2001), pp. 983–1009.

[157] Joost Driessen, Pascal J Maenhout, and Grigory Vilkov. "The price of correlation risk: Evidence from equity options". In: *The Journal of Finance* 64.3 (2009), pp. 1377–1406.

[158] José Afonso Faias and Pedro Santa-Clara. "Optimal option portfolio strategies: Deepening the puzzle of index option mispricing". In: *Journal of Financial and Quantitative Analysis* 52.1 (2017), pp. 277–303.

[159] Pedro Santa-Clara and Alessio Saretto. "Option strategies: Good deals and margin calls". In: *Journal of Financial Markets* 12.3 (2009), pp. 391–417.

[160] Andrew Ang. *Asset management: A systematic approach to factor investing*. Oxford University Press, 2014.

[161] Cornelis W Oosterlee and Lech A Grzelak. *Mathematical modeling and computation in finance: with exercises and Python and MATLAB computer codes*. World Scientific, 2019.

Acknowledgements

I would like to begin by expressing my heartfelt gratitude to my supervisor, Prof. Cornelis 'Kees' Oosterlee. Kees has been an exceptional guide in the realm of financial mathematics, providing not only invaluable subject-specific advice but also steadfast support in navigating the intricate world of academic formalities. Beyond his academic mentorship, Kees demonstrated remarkable patience and flexibility, especially during the period of the Covid-19 pandemic.

I am also indebted to Dr. Adam Andersson, who served as my master thesis supervisor, colleague, and, subsequently, co-author of the research article forming the basis for Chapter 4 in this thesis. His assistance and support have been instrumental to my academic journey.

Furthermore, I extend my gratitude to the consortium behind the ABC-EU-XVA project, including Prof. Iñigo Arregui, Prof. Pasquale Cirillo, Prof. Griselda Deelstra, Dr. Lech Grzelak, Prof. Andrea Pascucci, and Prof. Carlos Vazquez. Their collective contributions significantly enriched my research. I would also like to thank my fellow PhD candidates within the consortium, namely Graziana Colonna, Kevin Kamm, Roberta Simonella, Luis Souto Arias, Davide Trevisani, and Felix Wolf, for the enriching and enjoyable interactions during the course of this project.

Finally, I am deeply grateful to all my colleagues during my tenures at CWI, TU Delft, and the University of Utrecht. A special word of appreciation goes out to my office room-mates: Ki-Wai Chau, Sangeetika Ruchi, Balint Négyesi, Nikolaj Mücke, and Leonardo Perotti, whose friendships and shared moments have made this academic journey all the more meaningful.

Curriculum Vitae

2019-2022 Marie-Curie Early Stage Researcher
Centrum Wiskunde & Informatica, Amsterdam, the Netherlands

2022-2023 PhD Researcher
University of Utrecht, Utrecht, the Netherlands

2023 PhD. Applied Mathematics
University of Utrecht, Utrecht, the Netherlands
Thesis: *Neural networks for stochastic control and decision making in mathematical finance*
Promoter: Prof. dr. ir. C.W. Oosterlee

List of publications

5. **Kristoffer Andersson**, and Cornelis W. Oosterlee. *D-TIPO: Deep time-inconsistent portfolio optimization with stocks and options.* arXiv preprint arXiv:2308.10556 (2023).

4. **Kristoffer Andersson**, Adam Andersson, and Cornelis W. Oosterlee. *Convergence of a robust deep FBSDE method for stochastic control.* SIAM Journal on Scientific Computing 45.1 (2023): A226-A255.

3. **Balint Negyesi**[1], Kristoffer Andersson, and Cornelis W. Oosterlee. *The One Step Malliavin scheme: new discretization of BSDEs implemented with deep learning regressions.* arXiv preprint arXiv:2110.05421 (2021). To appear in: IMA Journal of Numerical Analysis Oxford University Press

2. **Kristoffer Andersson**, and Cornelis W. Oosterlee. *Deep learning for CVA computations of large portfolios of financial derivatives.* Applied Mathematics and Computation 409 (2021): 126399.

1. **Kristoffer Andersson** and Cornelis W. Oosterlee. *Deep learning for CVA computations of large portfolios of financial derivatives.* Applied Mathematics and Computation 409 (2021): 126399. APA

[1]The main author of this paper is Balint Negyesi and the contents of this paper do not appear in this thesis

List of presentations

Talks

6. **21th Winter school on Mathematical Finance**, Amersfoort, the Netherlands, January 2024.

5. **Dutch Math Finance afternoons**, Delft, the Netherlands, March 2023.

4. **4th International Conference on Computational Finance**[2] (ICCF), Wuppertal, Germany, June 2022.

3. **SAAB Technical Forum - Radar systems**, Gothenburg, Sweden, March 2022.

2. **Workshop on Machine Learning in Quantitative Finance and Risk Management**[3], Amsterdam, the Netherlands, July 2020.

1. **9th International Congresson Industrial and Applied Mathematics** (ICIAM), Valencia, Spain, July 2019.

Posters

1. **Utrecht AI Labs Event**, Utrecht, the Netherlands, May 2023.

[2]Awarded *"Best youngster presentation"*
[3]Organized by Cornelis W. Oosterlee and Kristoffer Andersson