Regression Monte Carlo and Applications in Energy

Alessandro Balata

Submitted in accordance with the requirements for the degree of
Doctor of Philosophy

The University of Leeds
Department of Applied Mathematics

2019

Abstract

This Thesis contains three separate parts. The first part is a monograph, where we present the history and novel developments to the family of Regression Monte Carlo algorithms applied to stochastic control problems. The second part explores different applications and experiments where we show the behaviour, and some interesting characteristics, of the methods presented in part I. The third, and last part of the thesis deals with the current energy systems and features a study of optimal design and management of a microgrid system inspired by the facility installed in Huatacondo, Atacama desert, Chile.

To my parents, without who none of this would have been possible

and to Roberta, who has always been on my side despite long weekends of typing.

Chentu concas, chentu berrittas

Popular Sardinian proverb. “hundred heads, hundred hats”

Acknowledgements

This research features the outcome of different projects that have been carried out by a team which has included C. Alasseur (FIME director EDF, [1]), S.B. Aziza (LAMSIN, [1]), C. Hure (University Paris Diderot, [5]), M. Laurière (NYU Shanghai, [5]), M. Ludkovski (UC Santa Barbara, [6]), A. Maheshwari (UC Santa Barbara, [1, 6]), J. Palczewski (University of Leeds, [3, 4, 6]), H. Pham (University Paris Diderot, [5]), I. Pimentel (Ecole Polytechnique, [5]), P. Tankov (ENSAE ParisTech, [1]) and X. Warin (FIME EDF, [1]). My contribution to these works has been different in each project, however, it has mostly involved the design, implementation, tuning and analysis of convergence of Regression Monte Carlo algorithms employed in aforementioned papers.

I am deeply grateful to the people who accompanied me during this journey. First and foremost my supervisor Jan Palczewski, who inspired me to enter the academic world, and has offered me constant personal and professional support and advice. I am also thankful to Elena Issoglio and Tiziano De Angelis, for the example they set for me, but mostly for their friendship. I feel lucky to have met Dario Domingo and Fabio Peruzzo, along with all the fellow Ph.D. students with whom I shared this journey, and who have been an invaluable company to share ideas and thoughts of all kinds. Mousa Huntul, Shambo Bhattacharjee and Fatima Almulhim have made the days spent at the desk as pleasant as they could have been, and I thank them for this. I am also very grateful to the whole stochastic control group in Leeds, who has offered me, during seminars and tea-time discussions, an invaluable insight on topics that I only marginally explored in my personal research activity. Finally, I am in debt with all the people, mentioned and unmentioned in this paragraph, who have contributed to make this journey one of the best experiences in my life, as well as shape my character during the past four years, producing the best myself I have ever been.

Contents

List of Figures

List of Tables

Preface

I started working on this project with a clear goal in mind. I wanted to contribute with my research to the fight against pollution and climate change. From my position the best way of doing so was to investigate the causes hindering an increasing penetration of renewable energy generation. One of the most challenging obstacles to a conversion to a fully sustainable energy system is represented by the unpredictability and lack of control over the energy produced by renewables. Electricity demand is growing rapidly in the less developed countries, while in the old world the increasing electrification is changing the historical patterns of consumption on which the design of the power system had been based when it was built decades ago. At the same time, the diminishing share of dispatchable power decreases the ability of the grid operator to react to fast changes in the power production/consumption. To better understand the dynamics driving the difficulties just mentioned, and to devise possible remedies, we turned to problems of stochastic control, whose solutions were very difficult to approximate precisely with the available numerical techniques. For this reason, in this project, we developed an efficient algorithm to solve general stochastic control of discrete Markov processes problems and prove the convergence to the true, optimal solution.

This project addresses the problems highlighted in the previous paragraph, focusing on the role of energy storage in the modernization of the electrical grid. Energy storages are studied, in particular, using tools and techniques developed in the

field of stochastic analysis, which is the branch of mathematics that investigates the properties of random movements and evolution through time. The aim of the project is to develop robust and efficient numerical techniques to analyse and evaluate the optimal control of a network comprising storage devices and renewable forms of energy production, along with traditional generation and different nodes of demand. The model of power network which will be studied is called microgrid. Microgrids contain all elements present on a full-scale national grid but on a smaller scale. The microgrid concept is gaining much attention today as more distributed generation is being installed, and this configuration allows to exploit their potential at best, avoiding curtailments of production. In order to manage the operation of the microgrid, a controller device usually exploits the flexibility of the electricity storages to increase the dispatchability of the renewable generation and to support the system during hours of low generation. Microgrids, however, are not just islanded systems; they can rather be pieces of a larger system, which dynamically controls topology and connectivity of its components in order to guarantee reliability and quality of service in a scenario of increasing electricity demand and renewable penetration.

In the thesis that follows, we will delve first into Markov processes and stochastic control problems, and describe the dynamic programming equation which inspires the class of numerical methods that we want to explore. In the second chapter, we will introduce approximations of conditional expectations via empirical projections and present two algorithms based on such approximation to solve general stochastic control problems. In addition, we prove the convergence of the scheme to the analytical solution in three theorems that represent the main mathematical contribution of the thesis. In the two chapters that follow, we will explore three interesting problems in detail: inventory, stopping and switching. We conclude the first part of the thesis with a chapter that investigates practical aspects of the implementation and the differences between the two algorithms introduced. The second part of the thesis features two chapters that present a number of examples of

inventory and full control problems that are used to highlight particular aspects of the algorithms. In the third part of the thesis, we will come back to energy systems and describe some of the steps necessary to transform microgrids into "smart-grids". We begin with an overview of the current energy landscape, challenges and possible solutions, and a description of the most representative technologies. The chapter that follows introduces our mathematical modelling of a microgrid and the description of the numerical experiments. A collection of management and design advice concludes this thesis.

Part I

Regression Monte Carlo

Chapter 1

Roadmap

In this chapter, we present the framework of stochastic control of Markov processes and provide an introduction to the different problems we will deal with in the rest of the Thesis. Recall that, as explained in the preface, our main contribution resides in developing, and providing convergence results for, the Regress Later Monte Carlo algorithm employed to solve stochastic control problems of the kind listed in this chapter.

The structure of the chapter is as follow: we open by explaining what a control problem of discrete stochastic Markov processes involves, and then present some special cases of the general formulation that are worthwhile discussing given the popularity they enjoy in the community. We will continue by discussing the dynamic programming principle that allows the reduction of these very high dimensional problems into a sequence of lower dimensional ones, thereby allowing us to solve them. We will conclude introducing the difficulties and possible approaches to solve these problems numerically and then, finally, we show how continuous time problems can be approximated within our framework too.

1.1 Problem formulation

Let us consider a controlled Markov process $(X_n)_{n=0}^N$ in discrete time, on a domain $\mathcal{D} \subseteq \mathbb{R}^d \times \{0, \ldots, N\}$, to which we sometimes refer to as *state space*, specified as follows:

$$X_{n+1} = \varphi(n, X_n, \xi_n, u_n),\ X_0 = x, \tag{1.1.1}$$

where φ is a Borel-measurable function and $\{\xi_n\}_{n=0}^N$ is a collection of i.i.d. uniformly distributed random variables on $[0, 1]$. Define by $(\mathcal{F}_n),\ n > 0$ the filtration generated by the sequence of r.v. $\xi_0, \ldots, \xi_{n-1}$, i.e. $\mathcal{F}_n = \sigma\{\xi_k : k < n\}$. The process $\{u_s\}_{s=0}^N \in \mathcal{D}_u \subseteq \mathbb{R}^p \times \{0, \ldots, N\}$, the *action space*, is called the control, and we require that at time n it is adapted to $(\mathcal{F}_n)$ or, in other words, no future information should be used to determine its value. We further postulate that the control process can always be given in feedback form as $u_s = u(s, X_s)$, which corresponds to an assumption of Markovianity on the system. We can see from Hernandez-Lerma and Lasserre [40] that (1.1.1) represents a general description of controlled Markov processes.

In this setting, we define a pathwise performance measure

$$J(n, (X_s, u_s)_{s=n}^N) = \sum_{s=n}^{N-1} f(s, X_s, u_s) + g(X_N), \quad n = 0 \ldots N, \tag{1.1.2}$$

where the functions f and g represent the current and the terminal reward. The functional J provides us with a way of assessing the goodness of the control process $\{u_s\}_{s=0}^N$ on a realised path $\{X_s\}_{s=0}^N$. Notice that we used u_N as a dummy variable that does not influence the state of the system but eases notation.

Define the control problem as the problem of finding the control process u that optimises the average performance measure, equivalently, the optimal control problem can be stated in terms of computing the maximal (minimal) average value, called value function, of the performance measure with respect to the choice of the

control process:

$$V(n,x) = \sup_{u \in \mathcal{U}_{\{n:N\}}(x)} \mathbb{E}\Big[J(n,(X_s,u_s)_{s=n}^N)\Big|X_n = x\Big], \tag{1.1.3}$$

where we define the set of admissible control processes from time n to N as

$$\mathcal{U}_{\{n:N\}}(x) = \{(u_s)_{s=n}^N : u_s \in \mathcal{W}_s(X_s) \cap \mathcal{D}_u,\ a.s.\ s = n,\ \ldots,\ N-1\},\ X_n = x$$

and $\mathcal{W}_s$ represents a set of problem specific constraints that might be given either implicitly or explicitly.

Numerical solution of this class of problem is presented in chapter 2.

1.1.1 Optimal control process

We denote by $(u_s^*)_{s=0}^N$ the optimal control that maximises the performance measure (1.1.2) and defines the value function in (1.1.3). The existence and uniqueness of such process have been long established for the class of problems we are interested in; in the following, we provide a brief discussion. For more details we refer to Hernandez-Lerma and Lasserre [40].

Recall that we decided to limit ourself to control processes that can be expressed in a feedback form. It turns out, however, that the optimal control processes, solutions to the problems introduced in this chapter, belong to the family of feedback controls. Feedback controls are processes that can be described in terms of a function of the current value of the state space variables, i.e $(u_s)_{s=0}^N = u(s,x)$. In order to guarantee the existence and uniqueness of the optimal control process in our setting, we need to guarantee the existence and uniqueness of a maximiser to the dynamic programming equation (1.3.12), that we introduce later, independently at each time step. The set of conditions under which the optimum is attained are usually called measurable selection conditions and include assumptions on the compactness of the admissible

control space and the semi-continuity or boundedness of the reward function . For further details refer to section 3.2 in Hernandez-Lerma and Lasserre [40]. Notice that, even though outside the scope of this Thesis, results presented in the following chapters might be extended to ϵ-optimal controls whenever the actual maximum cannot be achieved.

Example 1.1.1. ***Control of a robot*** *In this example, we show how a very tangible real-world problem can be cast in the framework introduced so far. Consider a robot operating on a disrupted surface such that the movement inputs provided by the controller translate in a stochastic movement of the robot. We consider the case of controlling a rescuing robot operating on debris which influences the movement of the robot as it passes over them with its wheels. We can assume the dynamics:*

$$X_{n+1} = \Big(X_n + \frac{u_n}{100} + \frac{1}{10}\xi_n\Big) \vee -2 \wedge 2,\, n = 0,\, \ldots,\, N$$

which corresponds to the case where the robot is moving inside a building (the external walls are -2*,* 2 *and* 0*,* N*). The controller wants to move the robot from one room to another avoiding hitting the walls, which would damage the machine. It is therefore the role of the controller to find the best controlling policy to move the robot from its starting point to the terminal one, passing through the doors that connect the different rooms. Notice that the charge that allows the robot to move is finite and therefore the controller also wants to minimise the use of control:*

$$V(n,x) = \min_{u \in \mathcal{U}_{\{n:N\}(x)}} \mathbb{E}\Big[\sum_{s=n}^{N} a\mathbb{1}_{\{u_s \neq 0\}} + b(u_s)^2 + c\mathbb{1}_{\{s=\tau_s\}}\mathbb{1}_{\{X_s \notin [d^-_{\tau_s}, d^+_{\tau_s}]\}}\Big| X_n = x\Big]. \quad (1.1.4)$$

We will discuss this problem at length in Part 2 of the thesis.

1.2 Special cases

We describe now different classes of stochastic control problems which can be formulated as special cases of problem (1.1.3) but, nonetheless, deserve a separate

	X	I
Stopping	uncontrolled	$\{0, 1\}$
Switching	uncontrolled	$\{1, \ldots, R\}$
Inventory	uncontrolled	$[0, I_{max}]$
Control	controlled	N/A

Table 1.1: The "X" column shows whether the process with stochastic dynamics is controlled, the "I" column shows the domain of the controlled process.

discussion for both historical and technical reasons. All problems that follow can be described by means of an additional process I, where I stands for index or inventory, which can be regarded as additional deterministic dimensions of the process $X = (X, I)$. For convenience, we use the same notation, whenever it does not cause confusion, for the exogenous dimension as well as its first set of dimensions. For reference, we display the main differences between the special cases discussed in this section in table 1.1.

1.2.1 Optimal switching problems

A well known special case of the general problem presented in the previous section is given by those problems where the control only takes values in a discrete set with low cardinality. The implications of this characteristic make the problem considerably easier to solve numerically. Notice that, by construction, the problem can always be rewritten with $\mathcal{D}_u \in \mathbb{R}$ simply by enlarging the set or regimes, to account for all possible combinations of regimes per dimension.

Enlarge the state space with an index process I taking values in a set of R different regimes to each of which a running reward function $f(n, x, u) = f_u(n, x)\,,\ u = 1 \ldots R$ is associated. The switching problem is to decide from which regime to profit at

a given point in space-time characterized by the triple (n, X_n, I_n). In order to choose the regime we want to switch to, we use the control process at time n, $u_n \in \{0, \dots, R\}$, while the index process $I \in \{0, \dots, R\}$ records the current regime by $I_{n+1} = u_n$. In the following, we will focus on a subclass of such problems to which most of the attention from literature has been devolved, namely the case where the regime does not influence the dynamics of X.

In general, some state-dependent switching costs will be associated to the problem: define $c_{i,j}(x)$ the cost of switching from regime i to regime j when in state x. Notice that in our discrete time framework, where only one switching decision per time step is allowed, we do not need to make any assumption (except for boundedness) about the switching costs. It is however common in literature and applications, and of fundamental importance in continuous time or when multiple switching decisions are allowed at one instant of time, to assume that (while dropping the explicit dependence from x in notation): not switching does not bring any cost, $c_{i,i} = 0 \, \forall i = 1 \dots R$; switching directly between two regimes is the cheapest choice, $c_{i,k} + c_{k,j} > c_{i,j} \, \forall i, j, k = 1 \dots R$, $i \neq j$; switching back and forth brings a cost, $c_{i,j} + c_{j,i} > 0 \, \forall i, j = 1 \dots R$. The previous set of assumptions is necessary to preserve existence and uniqueness and avoid infeasible control policies (switch an infinite number of times in a finite time horizon), but it is not strictly required in our setting where the number of possible switches is finite and bounded by N.

The general form for the performance measure in optimal switching problems is given as follow:

$$J(n, (X_s, I_s, u_s)_{s=n}^{N}) = \sum_{s=n}^{N-1} [f(s, X_s, u_s) - c_{I_s, u_s}] + g(X_N, u_N). \tag{1.2.5}$$

The optimal switching problem aims to find the optimal strategy $u^*(n, x, i) \in \{0, \dots, R\}$ to which the value function is associated:

$$V(n, x, i) = \sup_{u \in \mathcal{U}_{\{n:N\}}(x)} \mathbb{E}[J(n, (X_s, I_s, u_s)) | X_n = x, I_n = i]. \tag{1.2.6}$$

Classical formulations define a collection of value functions $\{V^j(n,x)\}_{j=1}^R$ while in our approach we consider the current regime to be part of the state space, allowing us to define only one value function and remain within the framework introduced at the beginning of the chapter.

Example 1.2.1. ***Gas-powered Turbines Control*** *Consider the controller of a Gas-fueled power plant whose job is to decide the operative regime of the plant in order to maximise profits on selling electricity. Consider a set of turbines with different characteristics; they can be operated separately or combined, producing R different regimes. Each regime is characterised by a different conversion coefficient $\{\mathcal{H}_i,\ \mathcal{Q}_i\}_{i=1}^R$ between gas burnt and electricity produced. Define the switching problem faced by the controller of the gas turbines as follow:*

$$J(n,(P_s,F_s,I_s,u_s)_{s=n}^N) = \sum_{s=n}^{N}(\mathcal{Q}_{u_s}P_s - \mathcal{H}_{u_s}F_s - C_{I_s})^+ - k - c_{I_s,u_s},$$

where some switching costs $c_{i,j}$ are paid every time a different set of turbines is shut off and turned on to switch from regime i to regime j. We indicate by C_j a deterministic carbon price paid according to the set of turbines in use, fix costs are represented by k and are not affected by the regime we select; P and F represents the exogenous processes for electricity and fuel price respectively.

The controller wants, therefore, to find the optimal control process $u^*(p,f)$ which defines:

$$V(n,p,f,i) = \sup_{u\in\mathcal{U}_{\{n:N\}}} \mathbb{E}[J(n,(P_s,F_s,I_s,u_s)_{s=n}^N)|P_n=p,F_n=f,I_n=i]$$

Numerical solution of this class of problem is presented in chapter 4.

1.2.2 Inventory problems

Inventory problems originated from the early works of Benes et al. [10] and Jacka [44] in a more restrictive framework of finite fuel problems, i.e. the inventory cannot be

replenished, and only a certain amount of "energy" is available for the control. We are motivated by problems inspired by real-option valuation, where the optionality comes from managing the inventory level over time. For further reference see Balata and Palczewski [4].

Consider an inventory process $I_{n+1} = \varphi_I(n, I_n, u_n)$, along with the process $X_{n+1} = \varphi_X(n, X_n, \xi_n)$, living in a compact domain $[0, I_{\max}]$ which can be depleted or replenished by the control u. The latter, we assume, can take values in a set $\mathcal{D}_u$ which might be continuous or discrete.

Remark 1.2.2. *When the control process is discrete, it is possible to rewrite the Inventory control problem as an Optimal Switching problem, without loss of generality. In the following, however, we will focus on the original inventory control forumlation.*

A typical optimal inventory problem reads:

$$V(n,x,i) = \sup_{u \in \mathcal{U}_{\{n:N\}}(x,i)} \mathbb{E}\Big[\sum_{s=n}^{N} f(s, X_s, I_s, u_s) \Big| X_n = x, I_n = i + g(X_N, I_N) \Big], \quad (1.2.7)$$

with its main peculiarity being the set of state dependent admissible controls

$$\mathcal{U}_{\{n:N\}}(x,i) = \Big\{ (u_s)_{s=n}^{N} : u_s \in \mathcal{D}_u : I_{s+1} = \varphi_I(s, I_s, u_s) \in [0, I_{\max}],\ s \in [n, N-1]] \Big\}, X_s = x,\ I_s = i.$$

Example 1.2.3. ***Battery charge management*** *A classic inventory type problem, which will be thoroughly discussed in the following chapters, is represented by the optimal control of the charging and discharging operations of a battery. Assume we are given a system where the controller has a reward associated with different levels of output/input from/to the battery. Consider, for example, the price arbitrage case discussed in chapter 6, where the controller profits from selling electricity to the grid when prices are high and buying electricity back when prices are low. Regardless of the specific reward function, however, one peculiarity of inventory management problems is that they could be formulated in terms of optimal inventory*

level, rather than optimal charging/discharging policy, redefining the control to be the current inventory level. This characteristic is due to the deterministic nature of the inventory, which guarantees a one-to-one correspondence between the values of (u_n, I_n) *and* I_{n+1}*. Managing the charging operations of a battery can, therefore, be seen as a problem of finding the optimal state of charge (SoC) in order to be prepared for intense charging or discharging operations in the future, at times of favourable electricity prices.*

Numerical solution of this class of problem is presented in chapter 3.

1.2.3 Optimal stopping problems

Optimal stopping problems have been introduced in the literature by the early work of Wald [77] and Snell [71] in a discrete time setting, in connection with sequential analysis. A complete analysis of optimal stopping problems for Markov processes can be found in Shiryaev [70]. The stopping problem is classically formulated through the following performance measure:

$$J(n, (X_s, \tau)_{s=n}^{N}) = \sum_{s=n}^{\tau} f(X_s) + h(X_\tau) \tag{1.2.8}$$

where $\tau \in [n, N]$ is the stopping time at which we decide to stop profiting from $f(X_t)$, to get the instantaneous terminal reward $h(X_\tau)$. The optimal stopping problem requires to find the optimal stopping time τ^* for which the expected value of the reward is maximal; we call such optimal expected reward the value function of the optimal stopping problem. The value function can be written as:

$$V(n, x) = \sup_{\tau \in [n,N]} \mathbb{E}\Big[J(n, (X_s, \tau)_{s=n}^{N}) \Big| X_n = x\Big]. \tag{1.2.9}$$

In order to fit (1.2.9) within our framework in (1.1.3), we identify the class of optimal stopping problems as a subclass of optimal switching problems, where only two

regimes are available and only one switching opportunity is allowed. In order to formalise the possibility of switching only once, we can introduce a switching cost $c_{1,0} = -\infty$ (respectively $+\infty$ for minimisation problems) or alternatively, define $\varphi_I(n, i, u) = i + u$ and the set of admissible controls as $\{u_s \in \{0, 1\} : I_{s+1} \in \{0, 1\} \text{ and } I_N = 1\}$. In addition, we set $f(n, x, 1) = 0 = g(x, \cdot)$, while we can regard the function h to be the switching cost from the regime $I = 0$ to the regime $I = 1$, i.e. $c_{0,1}(\cdot) = h(\cdot)$. In our framework we define therefore $\tau = \min\{s : u_s = 1\}$.

The equation for the value function associated with this framework is given by:

$$V(n, x, 0) = \sup_{u \in \mathcal{U}_{\{n:N\}}} \mathbb{E}\Big[\sum_{s=n}^{N-1} f(s, X_s, I_s) + c_{I_s, I_s+u_s} \Big| X_n = x, I_n = 0 \Big]. \tag{1.2.10}$$

Example 1.2.4. ***Bermudan options*** *A classic example, see McKean [55], of optimal stopping problem is represented by the American option pricing problem. This derivative contract grant the owner with the right, but not the obligation, to enter a long or short position on the underlying asset (usually a stock) at a predetermined fix price, called strike price K, over a time horizon (maturity). Contrary to a future contract, the payoff of these derivatives is usually asymmetric, implying that at the writing time, the contract has a value, or price, which the writer will charge to the buyer. This price is represented by the value function of the optimal stopping problem (which determines the best time to exercise the option). By their very nature, however, American option pricing problems should be formulated in continuous time, as the right to exercise the option, and therefore entering the position on the stock, can happen at any time from the writing to the maturity. Let us consider therefore a variant of American options called Bermudan, which offers exercise rights only at some prescribed dates during the life of the contract. The name of this contract comes from the geographical position of the Bermudan islands, situated "halfway" between Europe and America. Recall that European options are contracts that can only be exercised at maturity. The discrete set of exercise dates for a Bermudan option allows us to formulate our problem in discrete time. Let X*

represent a dynamic model for the price of the underlying, which we might think of as a stock. The payoffs of vanilla options are either of call (long) type or put (short) type and given, at time n, by $\left(X_n - K\right)^+$ *and* $\left(X_n - K\right)^-$ *respectively. In general, option contracts might exhibit more complex payoffs, and we might write a more general expression for the payoff as* $c_{0,1}(x)$*, where we require* $c_{0,1}$ *to be square integrable with respect to the conditional measure* $X_n|X_0 \ \forall\, n$*. Recalling that at maturity the optionality expires, and the payoff obtained is* $c_{0,1}$*, we can compute the price of the option at a given time* $n < N$ *by computing the value of the contract for the buyer at that time:*

$$V(n,x) = \sup_{u \in \mathcal{U}_{\{n:N\}}} \mathbb{E}[\sum_{s=n}^{N} c_{I_s, I_s+u_s}(X_s)|X_n = x].$$

Computing the price of a Bermudan option is, therefore, equivalent to finding the optimal stopping time for collecting the reward $c_{0,1}$*.*

1.3 Dynamic programming

A convenient equivalent representation of the value functions introduced in the previous section can be given by the dynamic programming principle, which states that the current value of the optimisation problem is given by the sum of the current reward function and the expectation of all future profits conditional on the present state. More rigorously, and using the words of Richard Bellman about his principle of optimality : "An optimal policy has the property that whatever the initial state and initial decision are, the remaining decisions must constitute an optimal policy with regard to the state resulting from the first decision" (See Bellman, 1957, Chap. III.3.).

Recall that we limited ourselves to work with feedback controls. In such instance we can formalise Bellman's words by considering the following equation associated

with the dynamic programming optimality principle for problem (1.1.3):

$$\begin{aligned} V(n,x) &= \mathbb{E}\Big[\sum_{s=n}^{N} f(s,X_s,u_s^*) + g(X_N)\Big|X_n = x\Big] \\ &= \mathbb{E}\Big[\sum_{s=n}^{n+t} f(s,X_s,u_s^*) + \sum_{s=n+t+1}^{N} f(s,X_s,u_s^*) + g(X_N)\Big|X_n = x\Big], \\ &= \mathbb{E}\Big[\sum_{s=n}^{n+t} f(s,X_s,u_s^*) + \mathbb{E}\Big[\sum_{s=n+t+1}^{N} f(s,X_s,u_s^*) + g(X_N)\Big|X_{n+t+1}\Big]\Big|X_n = x\Big] \\ &= \mathbb{E}\Big[\sum_{s=n}^{n+t} f(s,X_s,u_s^*) + V(n+t+1,X_{n+\tau+1})\Big|X_n = x\Big],\ t\in[0,N-n-1], \end{aligned} \tag{1.3.11}$$

where we indicated the optimal control $u^*(n,x)$ by u_n^*. Notice that in the expression above, second to third line, we used the tower property of conditional expectations.

Let us now introduce a shorthand notation that will be used throughout the thesis.

Definition 1.3.1. *We define the following notation to represent conditional expectations:*

$$\mathbb{E}_{n,x,u}\left[f(X_{n+1})\right] = \mathbb{E}\left[f(X_{n+1})\Big|X_n = x, u_n = u\right] = \int_{\mathcal{D}} f(y)\, d\nu(y)$$

where ν is the density of $(X_{n+1}|X_n)$. The values $\{n,x,u\}$ might, where relevant, be replaced by $\{n,x,i,u\}$ or other variables we need to condition upon. When the conditional expectation is applied to the value function, we will refer to it as the continuation value. For a formal definition of conditional expectation refer to chapter 4 in Rogers and Williams [67]

In the following subsections we will present the one-step dynamic programming equations (DPE, $t = 0$), for the different problems introduced before, and discuss the consequences that follow from this formulation.

1.3.1 General Markov processes

By setting $t = 0$ in the dynamic programming equation (1.3.11), we obtain the following representation of the value function, which offers a convenient separation between the immediate effect of the control and its influence on the state of the system:

$$V(n, X_n) = \max_{u \in \mathcal{U}_{\{n:n\}}(X_n)} \left\{ f(n, X_n, u) + \mathbb{E}_{n,X_n,u}\Big[V(n+1, X_{n+1})\Big] \right\}, \tag{1.3.12}$$

where the conditional expectation $\mathbb{E}_{n,X_n,u}[V(n+1, X_{n+1})]$ represents the most challenging expression to compute at each step of the Bellman dynamic equation.

Intuitively, the DPE shows that at each time step the controller faces a trade-off between maximising the deterministic instant reward $f(n, X_n, u)$ and driving the controlled process X towards areas of the state space where the reward in the future will be higher. As future performance are intrinsically random, to make decisions, the controller can only use its expectation of future rewards given the control chosen at present time, i.e. $\mathbb{E}_{n,X_n,u}\Big[V(n+1, X_{n+1})\Big]$. Notice the implicit dependence of the measure $X_{n+1}|X_n$ on the control u. The optimal control at time n is therefore given by the expression:

$$\begin{aligned} u_n^* = u^*(n, X_n) &= \operatorname*{argmax}_{u \in \mathcal{U}_{\{n:n\}}(X_n)} \left\{ f(X_n, u) + \mathbb{E}_{n,X_n,u}\Big[V(n+1, X_{n+1})\Big] \right\} \\ &= \operatorname*{argmax}_{u \in \mathcal{U}_{\{n:n\}}(X_n)} \left\{ f(X_n, u) + \mathbb{E}_{\xi_n}\Big[V\Big(n+1, \varphi(n, X_n, \xi_n, u)\Big)\Big] \right\}, \end{aligned} \tag{1.3.13}$$

where $\mathbb{E}_{\xi_n}$ represents the expectation with respect to the law of the random variable ξ_n, introduced in (1.1.1).

In the following subsections, we present the dynamic programming equation associated with the classes of problems introduced before: inventory, switching and stopping. The common characteristic of these problems is that they are easier to

solve than the general problem (1.1.3). In particular, the ease for switching problems comes from the set of admissible controls being discrete, for inventory problems from the control process not affecting the random dynamics of X, while for stopping both characteristics are present: the control takes discrete values (only two, in particular), and it does not affect random dynamics.

Remark 1.3.2. *In the following, for simplicity of notation, we will indicate* $\mathcal{U}_{\{n:n\}}(x)$ *by* $\mathcal{U}_n(x)$.

1.3.2 Inventory

When we consider the problems of optimal control of inventory, the resulting DPE shows why they are considered to be easier to solve than the general case and worth of separate analysis:

$$\begin{aligned} V(n,x,i) &= \max_{u\in\mathcal{U}_n(x,i)}\Big\{f(n,x,i,u)+\mathbb{E}_{n,x,i,u}[V(n+1,X_{n+1},I_{n+1})]\Big\} \\ &= \max_{u\in\mathcal{U}_n(x,i)}\Big\{f(n,x,i,u)+\mathbb{E}_{\xi_n}\Big[V\Big(n+1,\varphi_X(n,x,\xi_n),\varphi_I(n,i,u)\Big)\Big]\Big\}. \end{aligned} \tag{1.3.14}$$

The second line of (1.3.14) shows that the effect of the control u on the conditional expectation can be decoupled from the effect of the noise ξ allowing us to compute an explicit dependence on known values of i and u. We will see in the following how this becomes very helpful for the Regression Monte Carlo family of algorithms.

1.3.3 Optimal stopping and switching

Whenever the problem features discrete controls, we can formulate it in terms of optimal switching problem whose dynamic programming equation reads as follows:

$$V(n,x,i) = \max_{u\in\mathcal{U}_n(x)}\Big\{f(n,x,u)-c_{i,u}+\mathbb{E}_{n,x}[V(n+1,X_{n+1},u)]\Big\}, \tag{1.3.15}$$

where we leave $x \mapsto \mathbb{E}_{n,x}[V(n+1, X_{n+1}, u)]$ described implicitly.

The Bellman equation is telling us that the optimal policy is given by the rule:

$$\text{switch from } i \to j \text{ whenever } \max_{j \neq i}(f(n,x,j) - c_{i,j} + \mathbb{E}_{n,x}[V(n+1, X_{n+1}, j)] \geq f(n,x,i) + \mathbb{E}_{n,x}[V(n+1, X_{n+1}, i)].$$

The switching rule prescribes that at each time step we choose the best regime to stay in, comparing the alternatives of not switching and keep profiting from the regime we are currently in, and to switch to the best possible alternative, paying the switching cost and then continuing into the new regime for the next time step. The most striking simplification compared with the general case and the inventory problems is that only a finite set of conditional expectations, taken as a function of the state only, needs to be estimated, i.e. $\left\{\mathbb{E}_{n,x,i}[V(n+1, X_{n+1}, u)]\right\}_{u=u_1}^{u_R}$.

In the particular case of optimal stopping problems, where only two controls are available, the DPE reduces to an optimization problem in which we are left with the binary choice of stopping or continuing to the next time period:

$$\begin{aligned} V(n,x,i) &= \max_{u \in \{0,1\} : i+u \leq 1} \left\{ f(n, X_n, u) + c_{i,i+u} + \mathbb{E}_{n,x}[V(n+1, X_{n+1}, i+u)] \right\} \\ &= \max\left\{ f(n, X_n, 0) + \mathbb{E}_{n,x}[V(n+1, X_{n+1}, 0)],\ c_{0,1}(n, X_n) \right\}. \end{aligned} \tag{1.3.16}$$

The optimal decision for the optimal stopping problem at each time step is given by the following rule:

$$\text{stop whenever } c_{0,1} \geq f(n, X_n, 0) + \mathbb{E}_{n,x}[V(n+1, X_{n+1}, 0)].$$

that means we are left with a sequence of problems in which we chose the most profitable option between stopping the process and obtaining the terminal reward $c_{0,1}$, or taking the running reward f_0 and continuing optimally from time $n+1$ onward.

1.4 Numerical solution

We will focus our attention now on the numerical techniques that can be used to solve the problems introduced so far. Thanks to the formulation provided by the Bellman equation, we can tackle our problems in a probabilistic way, iteratively backward in time starting from a given terminal condition at time N. The simplicity of the control rules highlighted in the previous sections allows us to quickly determine the optimal policy, provided that we are able to compute the effect of choosing any given control on the system. As we work in a stochastic environment, the effect of the control can only be assessed in expectation; we show in the next chapter how to apply linear regression to samples from Monte Carlo simulations to approximate these expectations. Notice, in fact, that the conditional expectation $\mathbb{E}_{n,x}[V(n+1, X_{n+1})]$ in (1.3.12) is not known analytically (we would need to know the value function) nor numerically as, proceeding backwards on simulated trajectories, we only have samples from $V(n+1, \cdot)$ available. We will invest the next chapters in explaining how to efficiently transform these samples in estimates of the continuation value (conditional expectation of the value function) for the three classes of problems: general control of MPs, inventory and stopping and switching.

1.5 Continuous time problems

In this first chapter, we have described the mathematical framework within which we want to work and we specified that we were interested in looking at the system only at discrete moments in time. This choice is motivated by real-life applications in which it is not possible to observe the system too often, or perhaps the continuous dynamics are unknown. In other situations, on the other hand, we might require to have very frequent observations of the system and a natural description of the

phenomenon of interest is available in continuous time.

Similarly to the discrete case, let X be a Markov process and, for clarity, assume (but we don't have to) that X admits a representation as a solution of an SDE:

$$dX_t = b(t, X_t, u_t)dt + \sigma(t, X_t, u_t)dW_t \tag{1.5.17}$$

where W_t is a Brownian motion with respect to $\mathbb{P}$ and b and σ are bounded and Lipschitz. Previous notation stays in place, $\mathcal{F}_t$ is the sigma algebra generated by the process X up to time t, i.e. by $(W_s)_{s\leq t}$, and the control process u is adapted to such filtration.

A general representation of value function in continuous time can be given as follow

$$V(t,x) = \sup_{u\in\mathcal{U}_{\{t:T\}}(x)} \Big\{\mathbb{E}\Big[\int_t^T f(s, X_s, u_s)\, ds + g(X_T)\Big| X_t = x\Big]\Big\}, \tag{1.5.18}$$

while the DPE suggests that we have

$$V(t,x) = \sup_{u\in\mathcal{U}_{\{t:\theta\}}(x)} \Big\{\mathbb{E}\Big[\int_t^\theta f(s, X_s, u_s) d\, s + V(\theta, X_\theta)\Big|\mathcal{F}_t\Big]\Big\},\ \theta\in(t,T). \tag{1.5.19}$$

At this point, if we want to keep working in continuous time, we might decide to study (1.5.19) as $\theta \to t$. We can expect, intuitively, that this operation will provide us with a differential equation whose solution is the value function. Set first $\theta = t + dt$, $dt \approx 0$

$$\begin{aligned} V(t,x) &= \sup_{u\in\mathcal{U}_t(x)} \Big\{\mathbb{E}\Big[f(t, X_t, u_t)dt + V(t+dt, X_{t+dt})\Big|\mathcal{F}_t\Big]\Big\}. \\ \Rightarrow \sup_{u\in\mathcal{U}_t(x)} &\Big\{\mathbb{E}\Big[f(t, X_t, u_t)dt + V(t+dt, X_{t+dt}) - V(t,x)\Big|\mathcal{F}_t\Big]\Big\} = 0, \end{aligned}$$

and then approximate the integral with a discrete sum

$$\begin{aligned} &\sup_{u\in\mathcal{U}_t(x)} \left\{\mathbb{E}\left[f(t, X_t, u_t) + \frac{V(t+dt, X_{t+dt}) - V(t,x)}{dt}\Big|\mathcal{F}_t\right]\right\},\ \text{as } dt\to 0 \\ &\sup_{u\in\mathcal{U}_t(x)} \left\{\mathbb{E}\left[f(t,x,u) + \frac{\partial V(t,x)}{\partial t} + \mathcal{L}V(t,x,u)\Big|\mathcal{F}_t\right]\right\} = 0, \end{aligned} \tag{1.5.20}$$

where, from second to third line, we used Itô's formula, and $\mathcal{L}$ is the differential operator (generator) associated with the SDE (1.5.17) and given by $\mathcal{L}h(t,x,u) = b(t,x,u)\frac{\partial h(z)}{\partial z} + \frac{1}{2}\sigma(t,x,u)^2\frac{\partial^2 h(z)}{\partial z^2}$. For a more rigorous derivation refer to [72].

The PDE in (1.5.20) can be solved by means of numerical techniques, usually, by discretising both time and space dimension. Finite difference and finite element approaches heavily suffer from the underlying dimension of the state space, preventing the application to problems where the dimension of X is greater than three. Further discussion is however outside the scope of this thesis.

An alternative formulation of the optimality condition for the value function in (1.5.18) can be given in terms of Pontryagin's maximum principle [75], leading to a system of Forward-Backward stochastic differential equations (FBSDE) that can be solved by means of probabilistic as well as deterministic techniques. The literature in this area is vast; note the French school in particular, with Gobet et al. [34] for example, and the review given in Bender and Steiner [9].

Another approach is to apply time discretisation earlier, to equation (1.5.19) rather than (1.5.20). Consider a uniform discretisation of the interval $[0,T]$ into N sub-intervals delimited by the points $\{t_0 = 0,\, t_1,\, \ldots,\, t_N = T\}$, where $t_{i+1} - t_i = T/N =: \Delta t$. The first step to approximate a continuous time problem on such time grids, is to establish whether a closed form expression is available for the transition density of the process X, in which case we can just simulate its value at the points $\{t_0 = 0,\, t_1,\, \ldots,\, t_N = T\}$. When this is not the case we need to rely on approximation schemes that allow us to simulate any value X_n given the predecessor X_{n-1} and the control u_{n-1}. We approximate the effect of the reward function f over $[t,\theta)$, originally expressed as an integral $\int_t^\theta f(s, X_s, u_s)d\,s$, by holding f constant over this interval and computing $f(t, X_t, u_t)(\theta - t)$. Using the previous approximations we

can define a new control problem in discrete time as follow:

$$V(t_n, x) = \sup_{u \in \mathcal{U}_{t_n}(x)} \{f(t_n, X_{t_n}, u)\Delta t + \mathbb{E}[V(t_{n+1}, X_{t_{n+1}})|X_{t_n} = x, u_{t_n} = u]\}, \quad (1.5.21)$$

where we have set $\theta = t_n + \Delta t$. Notice that the optimal control u associated with the discrete time approximation (1.5.21) is not a discretisation of the optimal control process associated with the continuous time problem (1.5.19) but rather the solution of the newly defined discrete time one. Results about the convergence of the approximate value function to the continuous time one can be found in Kushner and Dupuis [48].

We can notice that (1.5.21) has brought us back to the more familiar discrete time setting presented at the beginning of the chapter. Results therein extend here by redefining the different functions in order to account for the multiplicative term Δt.

In this chapter we presented a roadmap of the topics in stochastic control that we intend to discuss in this Thesis: we introduced the general formulation of a controlled Markov process and stochastic control problem; we described the special cases of inventory, switching and stopping problems, all sharing the feature that the control does not affect the distribution of the stochastic component of the controlled process. We presented the dynamic programming principle and its one step formulation, commonly known as the dynamic programming equation, which provides us with a clear intuition on how to develop a numerical scheme to solve such problems. We shortly introduced the algorithms that are the central theme of this thesis and, finally, we touched upon the continuous time case.

In the next chapter, we introduce a numerical approximation of the conditional expectation appearing in (1.3.12), and describe two full algorithms that produce a control policy that approximates the optimal one for the general case of controlled Markov process. Additionally, we propose a framework to describe such algorithms that allow us to prove the convergence of the scheme, thereby providing the first

example in literature of proof of convergence for Regression Monte Carlo applied to this class of problems.

Chapter 2

RMC for control of Markov processes

In this chapter, we introduce the numerical approximation underpinning the whole Regression Monte Carlo family of algorithms: the regression approximation of conditional expectations. We first provide a high-level description of this technique, followed by a summary of the relevant literature on the topic where we underline the need for a framework that can allow us to prove the convergence of such algorithms. We then proceed by formulating the aforementioned framework by presenting some assumptions and useful lemmas. The three sections that follow, our main mathematical contribution to the literature, present a detailed description of three convergence theorems.

Let us recall the setting introduced in chapter 1. We consider a controlled Markov process $(X_n)_{n=0}^N$ on a domain $\mathcal{D} \subseteq \mathbb{R}^d$ specified by the transition equation (1.1.1):

$$X_{n+1} = \varphi(n, X_n, \xi_n, u_n), \tag{2.0.1}$$

where $\{\xi_n\}_{n=0}^{N-1}$ is a collection of i.i.d. uniformly distributed random variables on $[0, 1]$. Recall that in the previous chapter we assumed that the optimal control (u_n^*)

can be represented in a feedback form, i.e. the value of the process u^* at time n and state x can be computed as $u_n^* = u^*(n, x)$, we therefore limit ourself to search among the class of control processes that can be described in such a way.

We want to study the problem of computing the value function defined in (1.1.3) as:

$$V(n,x) = \sup_{u \in \mathcal{U}_{\{n:N\}}(x)} \mathbb{E}\Big[\sum_{s=n}^{N-1} f(s, X_s, u_s) + g(X_N) | X_n = x\Big],$$

which can be written in terms of dynamic programming equation (1.3.12):

$$\begin{cases} V(N,x) = & g(x) \\ V(n,x) = & \max\limits_{u \in \mathcal{U}_n(x)} \Big\{ f(n,x,u) + \mathbb{E}\Big[V(n+1, X_{n+1}) | X_n = x, u_n = u\Big] \Big\} \end{cases}$$

Remark 2.0.1. *For simplicity of notation, in the following, we will drop the dependence of the admissible control set $\mathcal{U}_n(x)$ on its argument x and simply write $\mathcal{U}_n$.*

2.1 Regression Monte Carlo

Notably, Monte Carlo methods are mostly known in their static version, in which a large number of samples is generated directly from a probability law, and expectations are computed with respect to the empirical measure induced by the simulations. The law of large numbers assures that for growing sample size, the Monte Carlo average will converge to the true mean of the distribution from which the data have been simulated. In fact, the empirical distribution itself will converge to the true one.

When a temporal dimension is included in the problem, the Monte Carlo method is usually called dynamic Monte Carlo; its main feature is that the sample average, classically computed by the method, is obtained from a sample of simulated paths

rather than single points (which nonetheless might be seen as higher dimension, correlated, r.v.s). Dynamic Monte Carlo simulations are usually started from an initial value X_0 and, from there, M simulated path are generated from the transition density of the process X. More formally:

Definition 2.1.1. *We denote by "training points", the set of d dimensional points* $\{X_s^m\}_{s=0,m=1}^{N,M}$. *We use double indexing to distinguish between different time steps (bottom index* $s \in [0, N]$*) and simulations (top index* $m \in [1, M]$*). Practically we might generate successors* $\{X_1^m\}_{m=1}^M$ *either directly from the conditional distribution of* $X_n|X_0$ *or from the dynamics of* X *as described by its transition function, starting from a set of initial values* $\{X_0^m\}_{m=1}^M$. *Iterating the process for* N *steps we obtain a collection of* M *paths* $\{X_s^m\}_{s=0,m=1}^{N,M}$.

2.1.1 How it works

To solve (1.1.3) we can intuitively assign to each simulated path (started from a common initial point $X_0^m = x,\ \forall\, m$) the value $J(0, (X_s^m, u_s^*)_{s=0}^N)$ so that a sample average over the paths will provide us with an estimation of the value function at the point $(0, x)$. Notice that, even though the method might not involve random simulations at all (paths might be generated deterministically for example) it is still called Monte Carlo for historical reasons.

For a given controlled path $\{X_s^m\}_{s=1}^N$ the quantity $J(0, (X_s, u_s^*)_{s=0}^N)$ can only be computed provided that we know the function $u^*(\cdot)$, which, in turn, allow us to compute the value of the control for each state visited by the process X. In order to estimate the control $(u_s^*)_{s=0}^N$ we need to refer to (1.3.12) that shows that at every state (n, x) the function $u^*(\cdot)$ at that point is represented as the solution of a maximisation problem involving the running reward f and the continuation value $\mathbb{E}[V(n+1, X_{n+1})|X_n = x]$. There exists a number of algorithms that can be

employed to approximate the solution of this maximisation problem (provided that the continuation value is known) but this lies outside the scope of this thesis. In the following, we will, instead, focus on the hurdle of approximating the continuation value. Such conditional expectation is, in general, not easily computable both analytically and numerically.

Following (1.3.12) we start from the known terminal condition and set $V(N, X_N^m) = g(X_N^m)\ \forall m$, which provides us with a starting value for the the backward iterative procedure where, at each time step, the value function is computed applying the one-step Bellman equation. To proceed to the next step of the iteration we have to maximise a function involving the continuation value. Naively, as Monte Carlo is the tool used to compute expectations, we could decide to run additional one-step simulations to compute each of the expectations needed during the main backward procedure, i.e. given a number of simulations L, $\mathbb{E}[V(n+1, X_{n+1})|X_n = x^m] \approx \frac{1}{L}\sum_{l=1}^{L} V(n+1, Y^l),\ Y^l = \varphi(n, x^m, u, \xi^l)\ \forall\, m$. However, as it can be understood, the nested simulations used to compute sample averages bring a burden of computational complexity that precludes practical implementation. The already high complexity of $O(MNd)$ simulated values (in the d-dimensional case) would increase to $O(MNdL)$, where L is the number of simulations used for each approximation of conditional expectations.

An alternative approach, within the simulation setting introduced so far, makes use of the cross-sectional responses $\{V(n+1, X_{n+1}^m)\}_{m=1}^M$ (where X_{n+1}^m are generated from potentially different starting points X_n^m, hence cross-sectional), available when the backward procedure has reached time n, to approximate the continuation value. Recall that a conditional expectation of a random variable Z given another random variable Y, $\mathbb{E}(Z|Y)$, might be regarded as a projection of Z from $\sigma(Z)$ to $\sigma(Y)$, where $\sigma(Y)$ represent the sigma algebra generated by Y. Under this interpretation we can then approximate the continuation value (projection) via linear regression of the

samples $\{V(n+1, X_{n+1}^m)\}_{m=1}^M$ over some transformations of the features $\{X_n^m\}_{m=1}^M$. This procedure will compute a sample approximation of the projection from $\sigma(X_{n+1})$ to $\sigma(X_n)$. We can see that if the control affects the distribution of X, we need samples of $V(n+1, \cdot)$ at points $X \sim (X_{n+1}|X_n, u_n^*)$ which, *a priori*, are available only after the problem has been solved. In other words, it is unclear how to sample from the distribution of $V(n+1, \cdot)$ without knowing the value of the optimal control.

Contrary to the case of optimal stopping and switching, or uncontrolled dynamics in general, in which the conditional law of $V(n+1, X_{n+1})|\,(X_n = x, u_n = u)$ can be estimated from the cross-sectional information contained in the simulated trajectories, in the case of controlled Markov processes such trajectories depend on the control process (u_n) and cannot, therefore, be simulated beforehand. In particular, given that the forward trajectories can be computed only by fixing the control, the estimated conditional expectation will be relevant only for that particular choice of the control.

In the following, we present an overview of the history of Regression Monte Carlo in literature, from its first appearances at the beginning of the 2000s, until the most recent developments (as of 2018). For the convenience of the reader, we display table 2.1 where we introduce the 4 main different specifications of the Regression Monte Carlo algorithm obtained by combining 2 possibilities for the projected function and 2 possibilities for projection space. More in detail, we will discuss Regress Now (RN - which has served as example so far), where we regress over functions evaluated at sample points $\{X_n^m\}_{m=1}^M$, as well as Regress Later (RL), where we regress over functions evaluated at sample points $\{X_{n+1}^m\}_{m=1}^M$. Similarly, we can project samples of the value function $\{V(n+1, X_{n+1}^m)\}_{m=1}^M$ in the Value Iteration (VI) approach, as well as samples $\{J(n+1, (X_s^m, u_s^m)_{s=n+1}^N)\}_{m=1}^M$ in the Performance Iteration (PI) approach. In the following, we will discuss each separately and prove the convergence of $RLVI$ and $RLPI$.

		projected variable	
		VI	PI
basis functions	RN	RNVI	RNPI
	RL	RLVI	RLPI

Table 2.1: Summary of the different specifications of Regression Monte Carlo algorithms. Different variables might be projected, VI for Value Iteration and PI for Performance Iteration (projecting V and J respectively). Regressors can be of two types too, RN for Regress Now and RL for Regress Later (evaluate basis functions at time n and $n+1$ respectively).

2.1.2 Regression Monte Carlo in literature

Regress Now approximations represent the hallmark of the first publications in the Regression Monte Carlo literature. The paternity of the idea is generally attributed to Carriere [19], who introduced a backward iterative scheme where conditional expectations were represented as non-parametric functions. On the other hand, Regression Monte Carlo as we know it today was introduced, in its main variations of Value and Performance Iteration, by Tsitsiklis and Van Roy [76] and Longstaff and Schwartz [51] respectively.

The initial papers have been focused exclusively on optimal stopping problems, and American option pricing in particular. This was due to the method being initially conceived in its (as it is known nowadays) Regress Now specification, thereby proving challenging to generalise to more general control problems, as explained in the next section. The interest in the method picked up relatively quickly, with a number of papers addressing some complementary issues: Clement et al. [21] showed the convergence rate of the scheme introduced in Longstaff and Schwartz [51], Moreno and Navas [58] compared different choices of parametric basis functions

to approximate conditional expectations, Glasserman and Yu [32] studied the trade-off between increasing the number of basis functions against the computational budget required for a good fitting, among others. A number of additional papers appeared during the years introducing more in-depth analysis or improvements on the standard approach. Note for example Egloff [27], that sharpened the results in Clement et al. [21] in a higher dimensional setting or Stentoft [73] who, in a similar setting, showed the increased efficiency over finite difference and binomial tree numerical schemes. Notice that we exclude from this discussion, and barely touch upon in the thesis, the numerous works involving the application of Regression Monte Carlo to the solution of systems of forward-backward stochastic differential equations (FBSDE), even though many control problems may be cast in such framework.

The Regress Later algorithm, a non-standard variation of Regression Monte Carlo, can be traced back to Broadie and Glasserman [14], Broadie et al. [15], Glasserman and Yu [31] and recently studied in Beutner et al. [11], Jain and Oosterlee [45], Nadarajah et al. [60]. In those papers, the regress-later approach is regarded as a tool to reduce the approximation error when applied to optimal stopping problems (e.g., American option pricing). Beutner et al. [11] demonstrated that it has a potential for significant improvement in the convergence rate of the approximations to the true value function. To the best of our knowledge, its ground-breaking potential for problems with controlled state variables has not been recognised yet, and it is discussed in this thesis. A recent application to the solution of systems of FBSDEs can be found in Briand and Labart [13] and Gnameho et al. [33]. In applications, Nadarajah et al. [60] compare Regress-Now and Regress-Later estimates in the context of energy real options, while Nadarajah and Secomandi [59] explore the links between Regression Monte Carlo and Approximate Dynamic Programming, a branch of Operational Research that focuses on the solution of control problems, for more details see Powell [64].

For the first 7-8 years, the literature on Regression Monte Carlo has been limited mostly to stopping and simple switching problems; noteworthy Carmona and Ludkovski [16]). Only after 2008 the first papers applying this algorithm to more general problems started to appear. The first field of application has been energy markets, where among others Boogert and de Jong [12] and Carmona and Ludkovski [17] introduced two distinct variations of the original to deal with control of noise-less processes (Inventory). See chapter 3 for more details.

The first methodological contribution aimed at generalising the standard Regression Monte Carlo method to problems of control of Markov processes has been the control randomisation approach proposed in Kharroubi et al. [46] and Langrené et al. [49], which will be presented in the case of inventory problems in section 3.2.3. The technique makes up for the limitations of the traditional Regress Now approach by explicitly introducing dependence on the control in the basis functions, in turn obtaining an estimated conditional expectation that depends on the choice of the control. In order for the regression approximation to have the correct statistical properties, an initial set of random trajectories of the control should be simulated and then used in the estimation of the projection coefficients. Notice that proof of convergence for this scheme is currently not available.

The first proof of convergence for a Regression Monte Carlo algorithm in a general setting was our contribution and appeared in Balata and Palczewski [3], to which this chapter is inspired. A comparison between Regress Later and Control Randomisation can be found in Balata et al. [5], however, the general conclusion was that Regress Later is easier to tune than Control Randomisation which is highly dependent on the choice of the initial randomised control. Further discussion can be found in Part II.

The contribution of this chapter is twofold: we give a systematic description of a powerful but simple algorithm to solve problems of stochastic control of Markov

processes, as well as provide theoretical and heuristic results. The Regress Later approach relies on the choice of basis functions and training measure, and we provide guidance on the different choices. We prove convergence of the Regress Later scheme, enriching the literature with both a new, effective numerical scheme with a proof of convergence and a new framework within which convergence of different Regression Monte Carlo schemes could be proved.

2.1.3 Regress Now and its limitations

In this section we describe one of the regression procedures used to approximate conditional expectations. Notice that, for square integrable functions h, the conditional expectation $\mathbb{E}[h(X_{n+1})|X_n = x]$ is a square integrable function of x which therefore admits (for a basis $\{\phi_1, \phi_2, \dots\}$) a series representation $\sum_{k=1}^{\infty} \alpha_k \phi_k(x)$, where the coefficients $\{\alpha_k\}_{k=1}^{\infty}$ are such that the norm with respect to a measure μ, denoted by $\|h(X_{n+1}) - \sum_{k=1}^{\infty} \alpha_k \phi_k(x)\|_\mu$, is zero. Recall, in fact, that the conditional expectation of a random variable, is the minimiser of the square distance from such random variable. This fact inspired the regression approximation which is the central subject of this thesis, for further reference see Emmanuel Gobet [28]. Before proceeding, let us set the notation. We will indicate by $\nu_n \sim (X_{n+1}|X_n)$ the conditional distribution of samples X_{n+1} and by $L^2_{\nu_n}$ the space of square integrable functions $f : \mathcal{D} \mapsto \mathbb{R}$ with respect to ν_n. More explicitly, we define the L^2 norm of a random variable Z on $\mathcal{D}$, denoted by $\|\cdot\|_\mu$, as the integral $\sqrt{\int_{\mathcal{D}}(Z)^2 d\mu}$. Random variables $Z \in \mathcal{D}$ such that $\int_{\mathcal{D}}(Z)^2 d\mu < \infty$ are called square integrable, and define a space denoted by L^2_μ.

Definition 2.1.2. *A family of K linearly independent functions $\{\phi_k(\cdot)\}_{k=1}^K : \mathcal{D} \to \mathbb{R}$ generating a linear subspace of L^2_ν is called a family of* basis functions.

Due to practical reasons that will become clear later on, we will neither assume that

the functions are orthogonal, nor that their norm is 1. In the following, for ease of understanding, the basis functions can be taken to be $\phi_k(z) = z^k$.

The Regress Now approximation of conditional expectation can therefore be written as

$$\mathbb{E}\Big[h(X_{n+1})|X_n = x\Big] \approx \sum_{k=1}^{K} \alpha_k \phi_k(x)$$

where the coefficients $\{\alpha_k\}_{k=1}^K$ minimise the sum of the square distance between samples of $\{h(X_{n+1}^m)\}_{m=1}^M$ and samples of $\{\sum_{k=1}^K \alpha_k \phi_k(X_n^m)\}_{m=1}^M$. Denote by $\boldsymbol{\alpha}$ the vector $(\alpha_1, \ldots, \alpha_K)$ and by $\boldsymbol{\phi}$ the vector of functions $(\phi_1, \ldots, \phi_K)$, then we have:

$$\boldsymbol{\alpha} = \underset{a}{\operatorname{argmin}} \Big\{ \frac{1}{M} \sum_{m=1}^{M} \Big(h(X_{n+1}^m) - \sum_{k=1}^{K} a_k \phi_k(X_n^m) \Big)^2 \Big\}. \tag{2.1.2}$$

Denoting by ℓ the loss function such that $\boldsymbol{\alpha} = \operatorname{argmin}_a \Big\{ \ell(\{X_n^m, X_{n+1}^m\}_{m=1}^M; a) \Big\}$, in (2.1.2), we can compute the regression coefficients by

$$\begin{aligned} \frac{\partial \ell\Big(\{X_n^m, X_{n+1}^m\}_{m=1}^M; \alpha\Big)}{\partial \alpha} &= \frac{2}{M} \sum_{m=1}^{M} \Big(h(X_{n+1}^m) - \boldsymbol{\alpha}\boldsymbol{\phi}(X_n^m) \Big) \boldsymbol{\phi}(X_n^m) = 0 \\ &\Leftrightarrow \\ \boldsymbol{\alpha} &= (\mathcal{A})^{-1} \frac{1}{M} \sum_{m=1}^{M} h(X_{n+1}^m) \boldsymbol{\phi}(X_n^m), \end{aligned} \tag{2.1.3}$$

where $\mathcal{A}$ is the matrix with entries $\mathcal{A}_{i,j} = \phi_i(X_n^m)\phi_j(X_n^m),\ \forall\, i, j$.

Notice that this representation of the conditional expectation using basis function, obtained through samples of $h(X_{n+1})$ and X_n, corresponds to two subsequent projections: the first one with respect to the distribution of X_{n+1} conditional on X_n, i.e. ν_n, to represent the conditional expectation; the second one, with respect to the distribution of X_n, to provide a representation in terms of the weighted sum of basis functions $\{\phi_k\}_{k=1}^K$.

The approach presented in this section is known as Regress Now, because it regresses samples of future values $h(X_{n+1})$ onto current values X_n or, in other

words, projects $h(X_{n+1})$ with respect to the measure ν onto the space generated by $\{\phi_1(X_n), \ldots, \phi_K(X_n)\}$. This nomenclature will feel more natural once we introduce Regress Later.

The main limitations of the Regress Now method lie in the concatenation of two projections mentioned above, a procedure that uses the same set of sample points to fit the basis functions and approximate the measure ν_n. The need to approximate the measure ν_n through samples $\{X_{n+1}^m, X_n^m\}_{m=1}^M$ makes the algorithm viable as long as the trajectories of the process X are not controlled by the process u. Whenever they are, the measure ν depends on the control process itself, preventing to simulate X_{n+1} from X_n without knowing in advance the function $u^*(n, \cdot)$. In practical terms, this would require redoing the regression step for each choice of the feedback function $u(n, X_n)$, $n = 1, \ldots, N-1$ by simulating the appropriate samples.

Given the limitations that characterise Regress Now, in the rest of the chapter, we will focus on the Regress Later technique only, and provide convergence theorems for the Value Iteration and Performance Iteration approaches. Notice, however, that the output of the former requires additional iterative computations before obtaining an unbiased estimation of the value of the control policy; more details will follow in the relevant section.

2.1.4 Regress Later approximations of conditional expectations

Contrary to Regress Now, the regression approximation in Regress Later does not rely on the distribution of the training points to approximate the conditional expectation but rather on analytical formulas, thereby allowing to account for the effect of the control seamlessly.

Denote by μ a probability measure on $\mathcal{D}$, we will sometimes refer to it as the *training*

distribution, and by $L^2_\mu = L^2(\mathcal{D}, \mu)$ the Hilbert space of square integrable functions $f : \mathcal{D} \mapsto \mathbb{R}$ with respect to μ. Notice that the measure μ can be chosen freely and has no direct relation with the randomness ξ driving the process X.

Definition 2.1.3. *Denote by* $\hat{\phi}^n_k$, $k = 1, \ldots, K$, $n = 0, \ldots, N-1$ *the conditional expectation of the basis functions with respect to the density of* $X_{n+1}|X_n$*:*

$$\hat{\phi}^n_k(x, u) = \mathbb{E}\left[\phi_k(X_{n+1}) \,|X_n = x, u_n = u\right].$$

In the Regress Later algorithm, conditional expectations are approximated in two steps: first the function $h(X_{n+1})$, where $X_{n+1} \sim \mu$, is represented as $\sum_{k=1}^K \alpha_k \phi_k(X_{n+1})$, then the conditional expectation is computed exploiting linearity and definition 2.1.3:

$$\mathbb{E}_{n,x,u}[h(X_{n+1})] = \mathbb{E}[\sum_{k=1}^K \alpha_k \phi_k(X_{n+1})|X_n = x, u_n = u] = \sum_{k=1}^K \alpha_k \hat{\phi}_k(x, u). \tag{2.1.4}$$

In formulas, the regression coefficients can be written as

$$\boldsymbol{\alpha} = \underset{a}{\operatorname{argmin}} \left\{ \sum_{m=1}^M \left(h(X^m_{n+1}) - \sum_{k=1}^K a_k \phi_k(X^m_{n+1}) \right)^2 \right\} = \mathcal{A}^{-1} \frac{1}{M} \sum_{m=1}^M h(X^m_{n+1}) \boldsymbol{\phi}(X^m_{n+1}). \tag{2.1.5}$$

We can notice that Regress Later deserves its name from the regression being performed "in the future", when values of $h(X_{n+1})$ are projected with respect to a measure μ onto the space generated by the basis functions $\{\phi_1(X_{n+1}), \ldots, \phi_K(X_{n+1})\}$. The computation of conditional expectations with respect to $(X_{n+1}|X_n, u_n)$ is therefore decoupled from the projection, allowing us to tackle stochastic optimisation problems with full control of the system dynamics.

Remark 2.1.4. *Notice that in RNMC the density of* $(X_{n+1}|X_n, u_n)$*, underpinning the conditional expectation, is recovered in the regression procedure through the use of samples of* X_{n+1} *distributed accordingly. The distribution of the samples* X_n*, is then accounted for when computing the expectations (or sample averages). In RLMC,*

on the other hand, the distribution $(X_{n+1}|X_n, u_n)$ *is used explicitly to compute analytically the result of conditioning* $h(X_{n+1})$ *onto* X_n*.*

Example 2.1.5. *Consider the function* $h(x) = x^4$ *and the random variable* $X \sim \mathcal{N}(z, 1/25)$ *(to be compared with* X_{n+1}*), we consider the two approximations introduced before, Regress Now and Regress Later, and a simpler interpolation of sample averages computed on a grid. Notice that in order to increase the error and improve the visibility (in particular for Regress Later) we consider only polynomial basis functions up to order 2, i.e.* $\{1, x, x^2\}$*. The different approximations obtained, along with the error on the true conditional expectation for* $z \in [0, 1]$ *(to be compared with* X_n*), and the training points employed are shown in Figure 2.1. Notice the high precision displayed by Regress Later compared to Regress Now.*

2.2 A framework for Regress Later projections

In the following, before introducing the projection framework and proving three useful bounds, we introduce some standing assumptions that will be valid throughout this thesis.

2.2.1 Assumptions

Assumption 1. *We assume that the process* X *has a transition density with respect to the measure* μ*, i.e.,*

$$\mathbb{P}\Big(X_{n+1} \in A | X_n = x, u_n = u\Big) = \int_A r(n, x, u; y)\mu(dy), \ \forall A \subseteq \mathcal{D}$$

and, in addition, this density is uniformly bounded

$$r(n, x, u; y) \leq \bar{R}^2 \ \forall\, n,\ x,\ u,\ y.$$

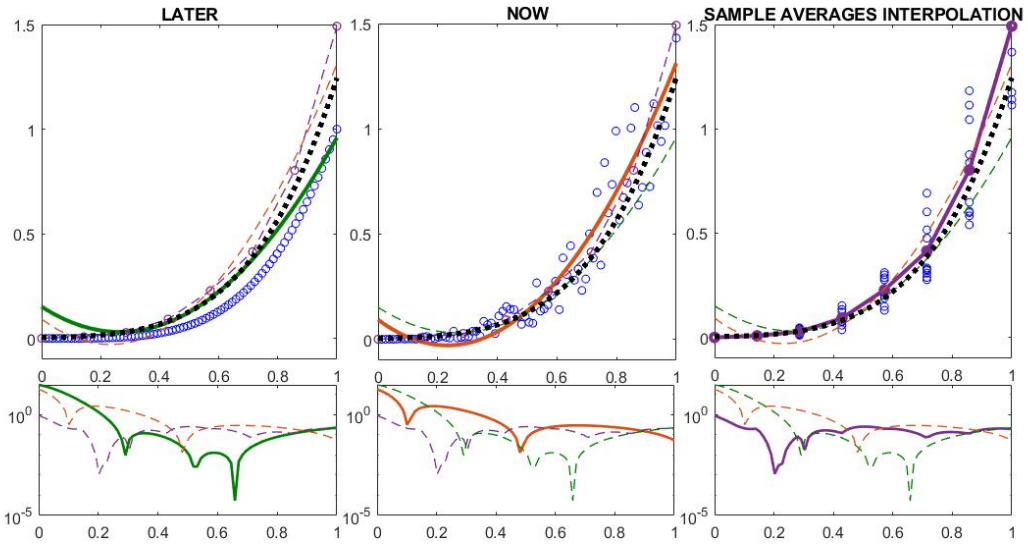


Figure 2.1: The three panels above show, from left to right and with a solid line, the approximations of conditional expectations obtained by Regress Later, Regress Now and the interpolation of sample averages. For comparison, a dashed line corresponding to the two remaining methods in each panel, and the true conditional expectation (blue dotted line), have been added. On the bottom each panel displays the relative error in logarithmic scale. Notice that the same picture, with a different style, is repeated in each panel, with the exception of the training points used to fit the approximations (blue circles), which corresponds to the method highlighted in each panel.

Remark 2.2.1. *Assumption 1, in most cases, is satisfied only when compact domains are considered. Therefore, even though we do not need to explicitly assume compactness of the domain $\mathcal{D}$ in our proofs, in practical applications truncation could be necessary.*

Assumption 2. *The reward function f and the terminal condition g are bounded, i.e. $\|f\|_\infty + \|g\|_\infty < \infty$, and upper-semi continuous on $\{0, 1, \ldots, N-1\} \times \mathcal{D} \times \mathcal{U}_n$ and $\mathcal{D}$ respectively.*

Remark 2.2.2. *The value function $V(n, x)$ is bounded, i.e. $V(n, x) < \Gamma$ for all $n = 0, \ldots, N$ and $x \in \mathcal{D}$. A trivial bound follows from the boundedness of f and g: $\Gamma \leq \bar{\Gamma} := (N-1)\|f\|_\infty + \|g\|_\infty$.*

2.2.2 Random projection operator

Let us introduce now the exact projection operator Π_K on L^2_μ which acts projecting its argument onto the space generated by the basis functions, i.e. $lin(\phi_1, \ldots, \phi_K) \subset L^2_\mu$. For $h \in L^2_\mu$, we have $\Pi_K : L^2_\mu \mapsto L^2_\mu : (\Pi_K h)(x) = \sum_{k=1}^K \alpha_k \phi_k(x)$ with the coefficients $\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_K)^T$ given by

$$\boldsymbol{\alpha} = \mathcal{A}_K^{-1} \Big\langle h, \boldsymbol{\phi} \Big\rangle_{L^2_\mu}, \tag{2.2.6}$$

where $\boldsymbol{\phi} = (\phi_1, \ldots, \phi_K)^T$ and $\mathcal{A}_K = \Big\langle \boldsymbol{\phi}, \boldsymbol{\phi}^T \Big\rangle_{L^2_\mu}$. The scalar product in L^2_μ can be written as an expectation with respect to μ, in the sense that:

$$\Big\langle h, \boldsymbol{\phi} \Big\rangle_{L^2_\mu} = \mathbb{E}_{\tilde{X} \sim \mu} \Big[h(\tilde{X}) \boldsymbol{\phi}(\tilde{X}) \Big], \text{ and } \mathcal{A}_K = \mathbb{E}_{\tilde{X} \sim \mu} \Big[\boldsymbol{\phi}(\tilde{X}) \boldsymbol{\phi}(\tilde{X})^T \Big]. \tag{2.2.7}$$

This guides us to a Monte Carlo estimator of $\boldsymbol{\alpha}$. We draw M i.i.d. copies $\tilde{X}^1, \ldots, \tilde{X}^M$ of $\tilde{X} \sim \mu$ which we call the *training points*. For $h \in L^2_\mu$ we approximate $\boldsymbol{\alpha}$ by

$$\hat{\boldsymbol{\alpha}} = \mathcal{A}_K^{-1} \frac{1}{M} \sum_{m=1}^M \Big[h(\tilde{X}^m) \boldsymbol{\phi}(\tilde{X}^m) \Big] \tag{2.2.8}$$

and define the *random projection operator* $\hat{\Pi}_K : L^2_\mu \mapsto L^2_M$ such that $\hat{\Pi}_K h = \sum_{k=1}^K \hat{\alpha}_k \phi_k$.

Denote by L^2_M the space linked to the training points, $L^2_M = L^2(\mathcal{D}^M, \mu^M)$, where we indicate by $\mathcal{D}^M = \mathcal{D} \times \cdots \times \mathcal{D}$ and $\mu^M = \mu \otimes \cdots \otimes \mu$. We also denote the "extended" space $L^2_e = L^2_M \times L^2_\mu = L^2(\mathcal{D}^{M+1}, \mu^{M+1})$. Notice that we could regard $\hat{\Pi}_K h$ as an element of $L^2_M \times lin(\phi_1, \ldots, \phi_K) \subset L^2_e$ because the random projection coefficients are functions of $(\tilde{X}^1, \ldots, \tilde{X}^M)$, i.e. $\hat{\boldsymbol{\alpha}} = \alpha(\tilde{X}^1, \ldots, \tilde{X}^M) \in L^2_M$.

Remark 2.2.3. *In formula* (2.2.8)*, we assume that $\mathcal{A}_K$ can be evaluated exactly (or precomputed with a very high precision) as it depends only on our choice of basis functions and the measure μ. This compares favourably (in terms of speed and accuracy) to Regress Now Monte Carlo in which $\mathcal{A}_K$ needs to be approximated from the training data.*

2.2.3 Extension of the random projection operator

We extend the projection operator introduced above to functions living in spaces bigger than L^2_μ. This is to introduce the notation which we will need later with no further mathematical complications. Define the space $L^2_{M,n} = L^2(D^{Mn}, \mu^{Mn})$ generated by n collections of M training points, $n = 1, \ldots, N$, denoted by $\{\tilde{X}^m_s\}^{M,N}_{m=1,s=N-n+1}$. The unusual indexing is related to times at which training points are placed while iterating backwards through the dynamic programming equation (1.3.12). We will write the set of n layers of training points $L^2_{e,n} = L^2_{M,n} \times L^2_\mu$. Let $L^2_{e,n} \ni h = h(\{\tilde{X}^m_s\}^{M,N}_{m=1,s=N-n+1}; \tilde{X})$. For the brevity of notation, we will write $\tilde{\mathfrak{X}}_n = \{\tilde{X}^m_s\}^{M,N}_{m=1,s=N-n+1}$, $L^2_{e,1} = L^2_e$, $L^2_{e,0} = L^2_\mu$, and $\tilde{\mathfrak{X}}_0 = \{\emptyset\}$. For a function $h \in L^2_{e,n} = L^2_{M,n} \times L^2_\mu$ we will identify the argument $\tilde{\mathfrak{X}}_n$ with the first set of coordinates corresponding to $L^2_{M,n}$ and $\tilde{X}$ with the remaining coordinate of L^2_μ.

Define an extended projection operator as

$$\Pi_K^{N-n} : L^2_{e,n} \mapsto L^2_{e,n} : \left(\Pi_K^{N-n} h\right)(\tilde{\mathfrak{X}}_n\,;\cdot) = \sum_{k=1}^{K} \alpha_k^{N-n} \phi_k(\cdot),$$

where

$$L^2_{M,n} \ni \boldsymbol{\alpha}^{N-n} = \boldsymbol{\alpha}^{N-n}\left(\tilde{\mathfrak{X}}_n\right) = \mathcal{A}_K^{-1} \mathbb{E}_{\tilde{X}\sim\mu}\Big[h(\tilde{\mathfrak{X}}_n; \tilde{X})\boldsymbol{\phi}(\tilde{X})\Big].$$

Notice that $\Pi_K^{N-n} h \in L^2_{e,n}$ since the coefficients $\boldsymbol{\alpha}^{N-n}$ still depend on the randomness contained in $\tilde{\mathfrak{X}}_n = \{\tilde{X}_s^m\}_{m=1,s=N-n+1}^{M,N}$. The superscript $N-n$ in $\Pi_K^{N-n} h$ indicates the dependence on $\tilde{\mathfrak{X}}_n$. However, from a mathematical perspective, for fixed $\tilde{\mathfrak{X}}_n$ the operator Π_K^{N-n} is identical to Π_K, and, indeed, it can be defined pointwise for each $\tilde{\mathfrak{X}}_n$. In fact, when $\tilde{\mathfrak{X}}_n$ is given and $h \in L^2_e$ is evaluated on $\tilde{\mathfrak{X}}_n$, the remaining function $x \mapsto h(\tilde{\mathfrak{X}}_n, x) \in L^2_\mu$, effectively making the operators Π_K^{N-n} and Π_K identical.

Similarly as above, we define the "random projection operator" acting on $h \in L^2_{e,n}$ by

$$\hat{\Pi}_K^{N-n} : L^2_{e,n} \mapsto L^2_{e,n+1} : \left(\hat{\Pi}_K^{N-n} h\right)(\tilde{\mathfrak{X}}_n\,;\cdot) = \sum_{k=1}^{K} \hat{\alpha}_k^{N-n} \phi_k(\cdot),$$

where

$$\hat{\boldsymbol{\alpha}}^{N-n} = \hat{\alpha}(\tilde{\mathfrak{X}}_n, \{\tilde{X}_{N-n}^m\}_{m=1}^M) = \mathcal{A}_K^{-1} \frac{1}{M} \sum_{m=1}^{M} h(\tilde{\mathfrak{X}}_n; \tilde{X}_{N-n}^m)\boldsymbol{\phi}(\tilde{X}_{N-n}^m),$$

and $\{\tilde{X}_{N-n}^m\}_{m=1}^M$ are i.i.d. random variables with the distribution μ. It follows that $\hat{\Pi}_K^{N-n} h \in L^2_{e,n+1}$. We can notice then that the Monte Carlo projection operator produces projections which live in a bigger space than the space where h lives, in particular, every projection adds one layer of training points so that the original space is enlarged by the addition of L^2_M. For a graphical representation of the spaces introduced in this section, see Figure 2.2 and Table 2.2. In the following section we will also use a corresponding notation for mathematical expectations, where we indicate by $\mathbb{E}_M$ and $\mathbb{E}_e$ expectation with respect to the measure μ^M and $\mu^M \times \mu$ respectively.

Space	Definition
L^2_μ	$L^2(\{D\}, \mu)$
L^2_M	$L^2_\mu, \ldots, L^2_\mu,\ M$ times
L^2_e	$L^2_M \times L^2_\mu$
$L^2_{M,n}$	$L^2_M, \ldots, L^2_M,\ n$ times
$L^2_{e,n}$	$L^2_{M,n} \times L^2_\mu$

Table 2.2: Sumary of the space of square integrable functions introduced so far.

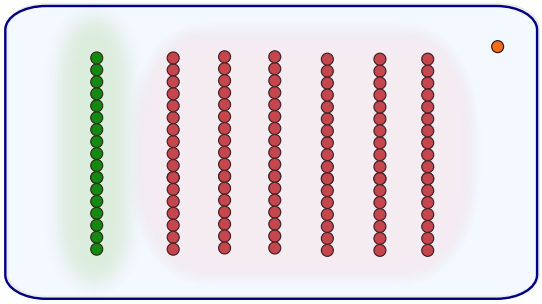


Figure 2.2: In the figure above we give, for the convenience of the reader, a graphical representation of the square integrable spaces introduced so far. Notice that every element is given by the measure μ, a different number of them is considered in the different spaces. L^2_μ in orange, L^2_M in green, $L^2_{M,6}$ in red and $L^2_{e,7}$ all together.

Remark 2.2.4. *Intuitively, we can think of the random projection operator with superscript $N-n$ as acting on a function $h(N-n+1,\cdot)$ that depends on the previous n layers of approximations, each comprising M training points and each distributed as μ. The effect of the random operator is a projection on the space generated by K basis functions and dependent on M i.i.d. training points with distribution μ. The rigorous description offered in this section, and the use of the measure μ to decouple the distribution of the training points from one time step to the next, is what allows us to prove the convergence of two iterative schemes based on the approximation of conditional expectation presented in section 2.1.4.*

2.2.4 Useful bounds

We present now two useful results relating the exact and random projection operators introduced above.

Lemma 2.2.5 (Projection error)**.** *Fix $\tilde{\mathfrak{X}}_n \in L^2_{M,n}$, then for $h \in L^2_{e,n}$, the error of the random projection operator with respect to the projection operator is bounded as follows:*

$$\left\|\hat{\Pi}_K^{N-n} h(\tilde{\mathfrak{X}}_n;\cdot) - \Pi_K^{N-n} h(\tilde{\mathfrak{X}}_n;\cdot)\right\|_{L^2_e} \leq \left\|\mathcal{A}_K^{-1}\right\|_2 \frac{1}{\sqrt{M}} SDev_{\tilde{X}\sim\mu}\Big(h(\tilde{\mathfrak{X}}_n\,;\tilde{X})\boldsymbol{\phi}(\tilde{X})\Big),$$

where

$$SDev_{\tilde{X}\sim\mu}\Big(h(\tilde{\mathfrak{X}}_n\,;\tilde{X})\boldsymbol{\phi}(\tilde{X})\Big) := \left(\sum_{k=1}^{K} Var_{\tilde{X}\sim\mu}\Big(h(\tilde{\mathfrak{X}}_n\,;\tilde{X})\phi_k(\tilde{X})\Big)\right)^{1/2},$$

and $\|\mathcal{A}_K^{-1}\|_2 = \max\{\|\mathcal{A}_K^{-1}x\|_2 :\ x\in\mathbb{R}^K,\ \|x\|_2 = 1\}$ is the matrix operator norm of $\mathcal{A}_K^{-1}$.

Proof. By the definition of projection operators we have

$$\begin{aligned}
&\left\|\hat{\Pi}_K^{N-n}h(\tilde{\mathfrak{X}}_n;\cdot)-\Pi_K^{N-n}h(\tilde{\mathfrak{X}}_n;\cdot)\right\|_{L_e^2}=\Big\|\sum_{k=1}^K\hat{\alpha}_k^n\phi_k(\cdot)-\sum_{k=1}^K\alpha_k^n\phi_k(\cdot)\Big\|_{L_e^2}\\
&=\left\|\Big(\mathcal{A}_K^{-1}(\frac{1}{M}\sum_{m=1}^M h(\tilde{\mathfrak{X}}_n;\tilde{X}^m)\boldsymbol{\phi}(\tilde{X}^m)-\mathbb{E}_{z\sim\mu}\Big[h(\tilde{\mathfrak{X}}_n;z)\boldsymbol{\phi}(z)\Big]\Big)^T\boldsymbol{\phi}(\cdot)\right\|_{L_e^2}\\
&=\left\|\Big(\frac{1}{M}\sum_{m=1}^M h(\tilde{\mathfrak{X}}_n;\tilde{X}^m)\boldsymbol{\phi}(\tilde{X}^m)-\mathbb{E}_{z\sim\mu}\Big[h(\tilde{\mathfrak{X}}_n,z)\boldsymbol{\phi}(z)\Big]\Big)^T\mathcal{A}_K^{-1}\boldsymbol{\phi}(\cdot)\right\|_{L_e^2}\\
&=\left\|\boldsymbol{\beta}_n^T\mathcal{A}_K^{-1}\boldsymbol{\phi}(\cdot)\right\|_{L_e^2},
\end{aligned}$$

where $\boldsymbol{\beta}_n=\frac{1}{M}\sum_{m=1}^M h(\tilde{\mathfrak{X}}_n;\tilde{X}^m)\boldsymbol{\phi}(\tilde{X}^m)-\mathbb{E}_{z\sim\mu}\Big[h(\tilde{\mathfrak{X}}_n;z)\boldsymbol{\phi}(z)\Big]$ and we used that $\mathcal{A}_K$ is symmetric. We have

$$\begin{aligned}
\left\|\boldsymbol{\beta}_n^T\mathcal{A}_K^{-1}\boldsymbol{\phi}(\cdot)\right\|_{L_e^2}^2&=\mathbb{E}_e\Big[\boldsymbol{\beta}_n^T\mathcal{A}_K^{-1}\boldsymbol{\phi}(\cdot)\boldsymbol{\phi}(\cdot)^T\mathcal{A}_K^{-1}\boldsymbol{\beta}_n\Big]=\mathbb{E}_M\Big[\boldsymbol{\beta}_n^T\mathcal{A}_K^{-1}\mathbb{E}_\mu[\boldsymbol{\phi}(\cdot)\boldsymbol{\phi}(\cdot)^T]\mathcal{A}_K^{-1}\boldsymbol{\beta}_n\Big]\\
&=\mathbb{E}_M\Big[\boldsymbol{\beta}_n^T\mathcal{A}_K^{-1}\boldsymbol{\beta}_n\Big]\le\left\|\mathcal{A}_K^{-1}\right\|_2^2\left\|\boldsymbol{\beta}_n^T\boldsymbol{\beta}_n\right\|_{L_M^2}^2\\
&=\left\|\mathcal{A}_K^{-1}\right\|_2^2\frac{1}{M}Var_{\tilde{X}\sim\mu}\Big(h(\tilde{\mathfrak{X}}_n\,;\tilde{X})\boldsymbol{\phi}(\tilde{X})\Big),
\end{aligned}$$

where in the last equality we used that $\{\tilde{X}^m\}_{m=1}^M$ are independent and distributed as μ. □

Remark 2.2.6. *With the dot notation $\cdot$ we often refer to the dependence of a function over a variable $\tilde{X}\sim\mu$.*

Lemma 2.2.7 (Standard Deviation)**.** *For a bounded function $h\in L_\mu^2$ we have:*

$$SDev_{\tilde{X}\sim\mu}\Big(h(\tilde{X})\boldsymbol{\phi}(\tilde{X})\Big)\le\sqrt{K}\|h\|_\infty\max_{k=1,\dots,K}\|\phi_k\|_{L_\mu^2}$$

Proof. Recall from Lemma 2.2.5 that

$$SDev_{\tilde{X}\sim\mu}\Big(h(\tilde{X})\boldsymbol{\phi}(\tilde{X})\Big)=\Big(\sum_{k=1}^K Var_{\tilde{X}\sim\mu}\Big[h(\tilde{X})\phi_k(\tilde{X})\Big]\Big)^{\frac{1}{2}}.$$

We bound now the variance with the second moment, and, using Jensen inequality, we have

$$Var_{\tilde{X}\sim\mu}\Big[h(\tilde{X})\phi_k(\tilde{X})\Big]\le\mathbb{E}_\mu\Big[\Big(h(\tilde{X})\phi_k(\tilde{X})\Big)^2\Big]\le\|h\|_\infty^2\Big\|\phi_k\Big\|_{L_\mu^2}^2.$$

We can now compute a bound for the full expression,

$$\begin{aligned} SDev_\mu\Big(h(\cdot)\boldsymbol{\phi}(\cdot)\Big) &\leq \Big(\sum_{k=1}^{K} \|h\|_\infty^2 \Big\|\phi_k\Big\|_{L_\mu^2}^2\Big)^{\frac{1}{2}} \\ &\leq \|h\|_\infty \Big(K \max_{k=1,\ldots,K} \|\phi_k\|_{L_\mu^2}^2\Big)^{\frac{1}{2}} \\ &= \|h\|_\infty \sqrt{K} \Big(\max_{k=1,\ldots,K} \Big(\|\phi_k\|_{L_\mu^2}\Big)^2\Big)^{\frac{1}{2}}, \end{aligned}$$

which leads to the statement of the Lemma. The last equality follows from the fact that $\|\phi_k\|_{L_\mu^2} \geq 0$. □

Remark 2.2.8 (Norm of the true projection operator)**.** *Let h be an element of $L_{e,n}^2$. The true projection operator Π_K^{N-n} admits the following bound:*

$$\Big\|\Pi_K^{N-n} h(\tilde{\mathfrak{X}}_n; \tilde{X})\Big\|_{L_{e,n}^2} \leq \Big\|h(\tilde{\mathfrak{X}}_n; \tilde{X})\Big\|_{L_{e,n}^2}.$$

Lemma 2.2.9 (Bound on conditional expectation)**.** *Under Assumption 1, i.e. the process X has a transition density with respect to the measure μ, and for any measurable function $\hat{u}(\tilde{\mathfrak{X}}_n, \tilde{X}) : L_{e,n}^2 \mapsto \mathcal{U}_n$ and $h \in L_{e,n}^2$, we have the following bound on the norm of the conditional expectation*

$$\begin{aligned} \Big\|\mathbb{E}\Big[h(\tilde{\mathfrak{X}}_n; X_{n+1})\Big|X_n = \tilde{X}, u_n =& \hat{u}_n(\tilde{\mathfrak{X}}_n, \tilde{X})\Big]\Big\|_{L_{e,n}^2} \\ &\leq \sup_{u \in \mathcal{U}_n} \Big\|\mathbb{E}\Big[h(\tilde{\mathfrak{X}}_n; X_{n+1})\Big|X_n = \tilde{X}, u_n = u\Big]\Big\|_{L_{e,n}^2} \\ &\leq \bar{R}\|h\|_{L_{e,n}^2}, \end{aligned}$$

where

$$\mathbb{E}\Big[h(\tilde{\mathfrak{X}}_n; X_{n+1})\Big|X_n = x, u_n = \hat{u}_n(\tilde{\mathfrak{X}}_n, \tilde{X})\Big] = \int_{\mathcal{D}} h(\tilde{\mathfrak{X}}_n; y)\; r(n, x, \hat{u}_n(\tilde{\mathfrak{X}}_n, x); y)\mu(dy).$$

Proof. Using Jensen inequality and Assumption 1:

$$\begin{aligned}\left\|\mathbb{E}\Big[h(\tilde{\mathfrak{X}}_n; X_{n+1})\Big| X_n = \tilde{X}, u_n = \hat{u}_n(\tilde{\mathfrak{X}}_n, \tilde{X})\Big]\right\|^2_{L^2_{e,n}} & \\ = \mathbb{E}_{M,n}\Big(\int_{\mathcal{D}} r(n, \tilde{X}, \hat{u}(\tilde{\mathfrak{X}}_n, \tilde{X}); y) h(\tilde{\mathfrak{X}}_n; y)\mu(dy)\Big)^2 & \\ \leq \sup_{u\in\mathcal{U}_n} \mathbb{E}_{M,n}\Big(\int_{\mathcal{D}} r^2(n, \tilde{X}, u; y) h^2(\tilde{\mathfrak{X}}_n; y)\mu(dy)\Big) & \\ \leq \bar{R}^2 \left\|h(\tilde{\mathfrak{X}}_n; \tilde{X})\right\|^2_{L^2_{e,n}}, &\end{aligned}$$

where $\mathbb{E}_{M,n}$ is the expectation with respect to the measure underlying the space $L^2_{M,n}$. □

In the following three sections we will present three iterative procedures: Value Iteration, Forward Evaluation and Performance Iteration. Each section will conclude with a theorem establishing the convergence of the quantity estimated to the real one. The two methods, Value Iteration and Performance Iteration, produce approximations of different quantities. The former estimates the value function, the latter the expected performance of the control policy. In order to compare the two methods, we use the Forward Evaluation procedure to compute an unbiased estimator of the average value of the control policy estimated by RLVI through Monte Carlo simulations.

2.3 Value Iteration

In this section we introduce the Value Iteration scheme and exploit our framework to prove the convergence of the algorithm for a general control problem.

Notice that some communities working on Markov decision process and operational research, when talking about Value Iteration, often refer to a different iterative appraoch than the one described in this section. Conscious of the confusion this

might cause, we respect the convention in the Regression Monte Carlo literature and use the term Value Iteration throughout the thesis with the meaning explained in the following paragraphs.

2.3.1 The method

Value Iteration is the most intuitive scheme inspired by the dynamic programming equation (1.3.12), and it has indeed been the model we had in mind when we introduced the iterative procedure that characterises Regression Monte Carlo, in section 2.1.1.

We start the backward procedure from time N, when the terminal condition is known, and we set $\hat{V}(N,x) = V(N,x) = g(x)$, $\forall\, x \in \mathcal{D}$. We move now to time $N-1$. The dynamic programming equation (1.3.12) requires us to compute $\mathbb{E}[V(N,X_N)|X_{N-1} = x, u_{N-1} = u]$; in order to do so we generate M samples $\{\tilde{X}_N^m\}_{m=1}^M$ from the distribution μ. These are used for the estimation of the projection coefficients $\hat{\alpha}^N$ in the random projection operator $\hat{\Pi}_K\hat{V} = \hat{\Pi}_K^N\hat{V} \in L_e^2$, i.e., we compute conditional expectations by first projecting $\hat{V}(N,\cdot) \in L_\mu^2$ over the basis functions $\{\phi_k\}_{k=1}^K$. Then, we compute analytically conditional expectations of the obtained linear combination of basis functions:

$$\begin{aligned}\mathbb{E}_{N-1,x,u}[V(N,X_N)] \approx \mathbb{E}_{N-1,x,u}[\hat{\Pi}_K^N\hat{V}(N,X_N)] &= \sum_{k=1}^K \hat{\alpha}_k^N \mathbb{E}_{N-1,x,u}[\phi_k(X_N)] \\ &= \sum_{k=1}^K \hat{\alpha}_k^N \hat{\phi}_k^{N-1}(x,u)\,.\end{aligned}$$

We then set

$$\hat{V}(N-1,x) = \sup_{u\in\mathcal{U}_n}\Big\{f(N-1,x,u) + \sum_{k=1}^K \hat{\alpha}_k^N\hat{\phi}_k^{N-1}(x,u)\Big\},$$

and we move to the next time step $N-2$ with the function $\hat{V}(N-1,x) \in L_e^2 = L_{e,1}^2$ (due to the randomness in $\hat{\alpha}^N$ introduced by the training points $\{\tilde{X}_N^m\}_{m=1}^M$).

Similarly to the previous time step, we project the value function using the random projection operator, obtaining $\hat{\Pi}_K^{N-1}\hat{V}(N-1,x) \in L^2_{e,2}$, from which we can compute an estimator of the conditional expectation. Iterating this process allows us to compute the approximate value function $\hat{V}(n,\tilde{X}) \in L^2_{e,N-n}$:

$$\begin{cases} \hat{V}(N,x) = g(x), \\ \hat{V}(n,x) = \sup\limits_{u\in\mathcal{D}^U} \Big\{ f(n,x,u) + \mathbb{E}\Big[\hat{\Pi}_K^{n+1}\hat{V}(n+1,\cdot)|X_n = x, u_n = u\Big]\Big\} \\ \qquad\quad = \sup\limits_{u\in\mathcal{D}^U} \Big\{ f(n,x,u) + \sum_{k=1}^K \hat{\alpha}_k^{n+1}\hat{\phi}_k^n(x,u)\Big\}, \end{cases} \tag{2.3.9}$$

to which it corresponds the estimated control map

$$\hat{u}(n,x) = \operatorname*{argsup}_{u\in\mathcal{D}^U} \Big\{ f(n,x,u) + \sum_{k=1}^K \hat{\alpha}_k^{n+1}\hat{\phi}_k^n(x,u)\Big\}.$$

Notice that the random coefficients $\hat{\alpha}^{n+1}$ are independent from $\{\tilde{X}_n^m\}_{m=1}^M$ and also from the law of $(X_{n+1}|X_n = \tilde{X}_n^m, u_n = u)$. Therefore we can compute the conditional expectation in (2.3.9), exploiting linearity, as

$$\mathbb{E}\Big[\hat{\Pi}_K^{n+1}\hat{V}(n+1,\cdot)|X_n = x, u_n = u\Big] = \sum_{k=1}^K \hat{\alpha}_k^{n+1}\hat{\phi}_k^n(x,u).$$

This decomposition enables our approach for optimal control of Markov processes.

Both the value function $\hat{V}(n,x)$ and the regression coefficients $\hat{\alpha}^{n+1}$ depend implicitly on all the training points used at times $n+1,\dots,N$, i.e., on $\tilde{\mathfrak{X}}_{N-n}$. This dependence will be omitted in notation and only indicated in the proof by applying appropriate projection operators Π_K^{N-n} and $\hat{\Pi}_K^{N-n}$.

Remark 2.3.1. *Recall that the estimation of the continuation value in Value Iteration reuses the parametric value function estimated at the future time step, which does not preserve the boundedness property of the true value function. For this reason, and thanks to the bound in Assumption 2, we truncate the Monte Carlo estimation $\hat{V}$ obtaining $\|\hat{V}(n,\tilde{X})\|_{L^2_{e,n}} < \Gamma$. In particular, with abuse of notation, we redefine $\hat{V}(n,x)$ in* (2.3.9) *as $\hat{V}(n,x) \wedge \Gamma \vee -\Gamma \ \forall\, n,x$.*

Details of implementation are collected in Algorithm 1.

Algorithm 1 Regress-later Monte Carlo algorithm (RLMC) - Value iteration

input: M, K, μ, $\{\phi\}_{k=1}^{K}$

1: Pre-compute the inverse of the covariance matrix $\mathcal{A}_K$

2: Generate i.i.d. training points $\{\tilde{X}_N^m\}_{m=1}^M$ accordingly to the distribution μ.

3: Initialise the value function $\hat{V}(N, \tilde{X}_N^m) = g(\tilde{X}_N^m), \quad m = 1, \dots, M.$

4: **for** n=N-1 to 0 **do**

5: $\hat{\alpha}^{n+1} = \mathcal{A}_K^{-1} \frac{1}{M} \sum_{m=1}^{M} \left[\hat{V}(n+1, \tilde{X}_{n+1}^m) \phi(\tilde{X}_{n+1}^m) \right]$

6: Generate a new layer of i.i.d. training points $\{\tilde{X}_n^m\}_{m=1}^M$ accordingly to the distribution μ.

7: **for** m=1 to M **do**

$$\hat{V}(n, \tilde{X}_n^m) = \left(\sup_{u \in \mathcal{U}_n} \left\{ f(n, \tilde{X}_n^m, u) + \sum_{k=1}^{K} \hat{\alpha}_k^{n+1} \hat{\phi}_k^n(\tilde{X}_n^m, u) \right\} \right) \wedge \Gamma \vee (-\Gamma)$$

output: $\{\hat{\alpha}_n^k\}_{n,k=1}^{N,K}$

Remark 2.3.2. *Note that as the matrix $\mathcal{A}_K$ in line 5 of the algorithm is computed with respect to the measure μ, we do not need to estimate it and invert it at every time step as it is required in traditional Regression Monte Carlo methods. Rather, we can pre-compute it with high precision before starting the backward procedure saving computational time and improving the quality of estimations.*

2.3.2 Convergence results

We now present a theorem that estimates the error between the estimated and the true value function. Recall that we indicate by $\bar{R}$ the uniform bound on the transition density of X with respect to the measure μ, and by Γ the upper bound of the value function.

Theorem 2.3.3. *Under Assumptions 1 and 2, for all $n = 0, 1, \dots, N$, we have:*

$$\left\|\hat{V}(n,\tilde{X}) - V(n,\tilde{X})\right\|_{L^2_{e,N-n}} \leq \bar{R}\frac{\bar{R}^{N-n}-1}{\bar{R}-1}\Big(\epsilon_K + \frac{\sqrt{K}}{\sqrt{M}}\Gamma\|\mathcal{A}_K^{-1}\|_2 \max_{n=1,\dots,N}\|\phi_k\|_{L^2_\mu}\Big),$$

where $\epsilon_K = \max_{n=1,\dots,N}\left\|\Pi_K V(n,\tilde{X}) - V(n,\tilde{X})\right\|_{L^2_\mu}$.

In the proofs we will use the following shorthand notation $\mathbb{E}_{u_n(x)}[\,\cdot\,] := \mathbb{E}[\,\cdot\,|X_n = x, u_n = u(n,x)]$. For convenience, denote

$$\hat{\hat{V}}(n,x) = f(n,x,\hat{u}(n,x)) + \mathbb{E}_{\hat{u}(n,x)}\Big[\hat{\Pi}_K\hat{V}(n+1,\cdot)\Big],$$

where $\hat{u}_n(x) = \hat{u}_n(\tilde{\mathfrak{X}}_n, x)$ is the estimated optimal policy which in practice is represented by the maximiser in (2.3.9).

Proof of theorem 2.3.3. Recall that $\hat{V}(n,\tilde{X}) \in L^2_{e,N-n}$ and further notice that

$$\begin{aligned}\left\|\hat{V}(n,\tilde{X}) - V(n,\tilde{X})\right\|_{L^2_{e,N-n}} &= \left\|\hat{\hat{V}}(n,\tilde{X}) \wedge \Gamma \vee (-\Gamma) - V(n,\tilde{X})\right\|_{L^2_{e,N-n}} \\ &\leq \left\|\hat{\hat{V}}(n,\tilde{X}) - V(n,\tilde{X})\right\|_{L^2_{e,N-n}}.\end{aligned} \tag{2.3.10}$$

Now consider that, given the definition of $\hat{\hat{V}}$,

$$\begin{aligned}\hat{\hat{V}}(n,\tilde{X}) - V(n,\tilde{X}) &= f\Big(n,\tilde{X},\hat{u}_n(\tilde{X})\Big) + \mathbb{E}_{\hat{u}_n(\tilde{X})}\Big[\hat{\Pi}_K^{n+1}\hat{V}(n+1,X_{n+1})\Big] \\ &\quad - f\Big(n,\tilde{X},u_n^*(\tilde{X})\Big) - \mathbb{E}_{u_n^*(\tilde{X})}\Big[V(n+1,X_{n+1})\Big] \\ &\geq \mathbb{E}_{u_n^*(\tilde{X})}\Big[\hat{\Pi}_K^{n+1}\hat{V}(n+1,X_{n+1}) - V(n+1,X_{n+1})\Big],\end{aligned}$$

where the inequality is given by the substitution of $\hat{u}(n,x)$, which realises the maximum in $\hat{\hat{V}}$, with the true optimal control $u^*(n,x)$. Similarly replacing $u^*(n,x)$ by $\hat{u}(n,x)$ we obtain an upper bound

$$\hat{\hat{V}}(n,\tilde{X}) - V(n,\tilde{X}) \leq \mathbb{E}_{\hat{u}_n(\tilde{X})}\Big[\hat{\Pi}_K^{n+1}\hat{V}(n+1,X_{n+1}) - V(n+1,X_{n+1})\Big].$$

Therefore, using Assumption 1 and Lemma 2.2.9 we have the following bound:

$$\begin{aligned}\left\|\hat{\hat{V}}(n,\tilde{X})-V(n,\tilde{X})\right\|_{L^2_{e,N-n}} &\\ \leq \left\| \sup_{u\in\mathcal{D}^U} \mathbb{E}_u\left[\hat{\Pi}_K^{n+1}\hat{V}(n+1,X_{n+1})-V(n+1,X_{n+1})\right]\right\|_{L^2_{e,N-n}} &\\ \leq \bar{R}\left\|\hat{\Pi}_K^{n+1}\hat{V}(n+1,\tilde{X})-V(n+1,\tilde{X})\right\|_{L^2_{e,N-n}}.&\end{aligned} \tag{2.3.11}$$

We split now the term $\left\|\hat{\Pi}_K^{n+1}\hat{V}(n+1,\tilde{X})-V(n+1,\tilde{X})\right\|_{L^2_{e,N-n}}$ into three components:

$$\begin{aligned}\left\|\hat{\Pi}_K^{n+1}\hat{V}(n+1,\tilde{X})-V(n+1,\tilde{X})\right\|_{L^2_{e,N-n}} &\\ \leq \left\|\hat{\Pi}_K^{n+1}\hat{V}(n+1,\tilde{X})-\Pi_K^{n+1}\hat{V}(n+1,\tilde{X})\right\|_{L^2_{e,N-n}} &\\ + \left\|\Pi_K^{n+1}\hat{V}(n+1,\tilde{X})-\Pi_K^{n+1}V(n+1,\tilde{X})\right\|_{L^2_{e,N-n}} &\\ + \left\|\Pi_K^{n+1}V(n+1,\tilde{X})-V(n+1,\tilde{X})\right\|_{L^2_{e,N-n}}.&\end{aligned} \tag{2.3.12}$$

For the first term in (2.3.12) we have, using Lemma 2.2.5, Lemma 2.2.7 and the bound Γ for $\hat{V}$:

$$\begin{aligned}&\left\|\hat{\Pi}_K^{n+1}\hat{V}(n+1,\tilde{X})-\Pi_K^{n+1}\hat{V}(n+1,\tilde{X})\right\|_{L^2_{e,N-n}}\\ &=\left\|\left\|\hat{\Pi}_K^{n+1}\hat{V}(n+1,\tilde{X})-\Pi_K^{n+1}\hat{V}(n+1,\tilde{X})\right\|_{L^2_e}\right\|_{L^2_{M,N-n-1}}\\ &\leq \frac{\sqrt{K}}{\sqrt{M}}\Gamma\left\|\mathcal{A}_K^{-1}\right\|_2 \max_{k=1,\dots,K}\left\|\phi_k\right\|_{L^2_\mu}.\end{aligned} \tag{2.3.13}$$

The second term in (2.3.12) represents the backward propagation of the error and, using remark 2.2.8, can be used to set up a recursive relation between errors at different time steps:

$$\begin{aligned}\left\|\Pi_K^{n+1}\hat{V}(n+1,\tilde{X})-\Pi_K V(n+1,\tilde{X})\right\|_{L^2_{e,N-n}} &\\ = \left\|\Pi_K^{n+1}\hat{V}(n+1,\tilde{X})-\Pi_K V(n+1,\tilde{X})\right\|_{L^2_{e,N-n-1}} &\\ \leq \left\|\hat{V}(n+1,\tilde{X})-V(n+1,\tilde{X})\right\|_{L^2_{e,N-n-1}}.&\end{aligned}$$

The last term in equation (2.3.12) is bounded by ϵ_K (defined in the statement of the theorem):

$$\left\|\Pi_K V(n+1,\tilde{X}) - V(n+1,\tilde{X})\right\|_{L^2_{e,N-n}} = \left\|\Pi_K V(n+1,\tilde{X}) - V(n+1,\tilde{X})\right\|_{L^2_\mu} \leq \epsilon_K.$$

Collecting these results together we find that

$$\begin{aligned}\left\|\hat{\Pi}_K^{n+1}\hat{V}(n+1,\tilde{X}) - V(n+1,\tilde{X})\right\|_{L^2_{e,N-n}} \leq \left\|\hat{V}(n+1,\tilde{X}) - V(n+1,\tilde{X})\right\|_{L^2_{e,N-n-1}} \\ + \epsilon_K + \frac{\sqrt{K}}{\sqrt{M}}\Gamma\left\|\mathcal{A}_K^{-1}\right\|_2 \max_{k=1,\dots,K}\|\phi_k\|_{L^2_\mu}.\end{aligned} \tag{2.3.14}$$

Combining this result with (2.3.11) and (2.3.10), and denoting $\beta_n := \left\|\hat{V}(n,\tilde{X}) - V(n,\tilde{X})\right\|_{L^2_{e,N-n}}$, leads to the following recursion:

$$\begin{aligned}\beta_n &\leq \bar{R}\beta_{n+1} + \bar{R}\left(\epsilon_K + \frac{\sqrt{K}}{\sqrt{M}}\Gamma\left\|\mathcal{A}_K^{-1}\right\|_2 \max_{k=1,\dots,K}\|\phi_k\|_{L^2_\mu}\right) \\ &\leq \bar{R}\sum_{s=0}^{N-n-1}\bar{R}^s\left(\epsilon_K + \frac{\sqrt{K}}{\sqrt{M}}\Gamma\left\|\mathcal{A}_K^{-1}\right\|_2 \max_{k=1,\dots,K}\|\phi_k\|_{L^2_\mu}\right) \\ &= \bar{R}\frac{\bar{R}^{N-n}-1}{\bar{R}-1}\left(\epsilon_K + \frac{\sqrt{K}}{\sqrt{M}}\Gamma\left\|\mathcal{A}_K^{-1}\right\|_2 \max_{k=1,\dots,K}\|\phi_k\|_{L^2_\mu}\right),\end{aligned} \tag{2.3.15}$$

where we used that $\hat{V}(N,\cdot) = V(N,\cdot)$, i.e. $\beta_N = 0$. □

Remark 2.3.4. *Notice that even though the number of basis functions K is constant, we have it explicitly in the expression of theorem 2.3.3 because our result can be used to assess the convergence of the scheme as $K, M \to \infty$.*

Corollary 2.3.5. *If, in addition, we assume that $\mathcal{A}_K = Id$, i.e. the family $\{\phi_k\}_{k=1}^K$ is orthonormal, then the following bound holds:*

$$\left\|\hat{V}(n,\tilde{X}) - V(n,\tilde{X})\right\|_{L^2_{e,N-n}} \leq \bar{R}\frac{\bar{R}^{N-n}-1}{\bar{R}-1}\left(\epsilon_K + \frac{\sqrt{K}}{\sqrt{M}}\Gamma\right)$$

Proof. For an ortonormal family in L^2_μ, we have $\mathcal{A}_K = Id$, so $\mathcal{A}_K^{-1} = Id$ and therefore $\|\mathcal{A}_K^{-1}\|_2 = 1$. Since the basis functions are normalised, $\|\phi_k\|_{L^2_\mu} = 1$ for all $k = 1, \ldots, K$. Plugging these estimates in the bound obtained in theorem 2.3.3 leads to the statement of the corollary. □

2.4 Forward Evaluation

Recall that the Value Iteration procedure described above provides not only an approximation of the value function, but also an approximation of the optimal policy; in order to find the control at time n and in state x, it is sufficient to solve the optimization problem in the last line of (2.3.9). In practical applications, it is often the control policy, and not only the value, that is of interest. In this section, we, therefore, assess the value (performance) implied by the estimated policy.

2.4.1 The method

The only output from the backward procedure that we need is the matrix of projection coefficients $\{\hat{\alpha}_k^n\}_{n,k=1}^{N,K}$ which we employ in a *forward scheme* to take decisions. Recall that those projection coefficients are functions of $\tilde{\mathfrak{X}}_n$, but this dependence is suppressed below for the sake of clarity of notation. For a matrix $\{\xi_n^m\}_{m,n=1}^{M',N}$ of i.i.d. $U(0,1)$ variables, and a fixed initial condition x, we perform a Monte Carlo simulation as follows:

$$\begin{cases} X_0^m & = x, \\ X_{n+1}^m & = \varphi(n, X_n^m, \xi_n^m, \nu_n^m), \qquad m = 1, \ldots, M',\ n = 0, \ldots, N-1, \end{cases} \tag{2.4.16}$$

where the estimated optimal control ν_n^m is computed as:

$$\nu_n^m = \underset{u \in \mathcal{D}^U}{\operatorname{argmax}} \Big\{ f(n, X_n^m, u) + \sum_{k=1}^{K} \hat{\alpha}_k^{n+1} \hat{\phi}_k^n(X_n^m, u) \Big\}. \tag{2.4.17}$$

We then set:

$$v(x) = \frac{1}{M'} \sum_{m=1}^{M'} \Big(\sum_{n=1}^{N} f(n, X_n^m, \nu_n^m) + g(X_N^m) \Big).$$

In the following we use the notation "Evaluate the policy" to refer to the routine specified by equations (2.4.16)-(2.4.17).

The above Monte Carlo evaluation of the policy approximates $\tilde{V}(0, x)$, where the valuation function $\tilde{V}$ is defined as follows:

$$\begin{cases} \tilde{V}(N, x) = g(x) \\ \tilde{V}(n, x) = f(n, x, \hat{u}(n, x)) + \mathbb{E}\big[\tilde{V}(n+1, X_{n+1}) \big| X_n = x, u_n = \hat{u}(n, x)\big], \end{cases} \tag{2.4.18}$$

where

$$\hat{u}_n(x) = \underset{u \in \mathcal{D}^U}{\operatorname{argmax}} \Big\{ f(n, x, u) + \sum_{k=1}^{K} \hat{\alpha}_k^{n+1} \hat{\phi}_k^n(x, u) \Big\}.$$

Indeed, it is easy to see that (2.4.18) has a representation

$$\tilde{V}(n, x) = \mathbb{E}\left[\sum_{t=n}^{N} f\big(t, X_t, \hat{u}_t(X_t)\big) + g(X_N) \right].$$

Using (2.4.18) has advantages for proving convergence over the above forward running representation or its Monte Carlo estimate (2.4.16)-(2.4.17).

2.4.2 Convergence results

We now present a theorem that estimates the error between the estimated ($\hat{V}$) and the true (V) average performance of the control policy. Recall that we indicate by $\bar{R}$ the uniform bound on the transition density of X with respect to the measure μ, and by Γ the upper bound of the value function.

Theorem 2.4.1. *Under Assumptions 1 and 2, for all $n = 0, 1, \ldots, N$, we have:*

$$\begin{aligned} \Big\| \tilde{V}(n, \tilde{X}) - V(n, \tilde{X}) \Big\|_{L^2_{e,N-n}} \leq \frac{2\bar{R}}{(\bar{R}-1)^2} \Big((N-n)\bar{R}^{N-n+1} - (N-n+1)\bar{R}^{N-n} + 1 \Big) \\ \times \Big(\epsilon_K + \frac{\sqrt{K}}{\sqrt{M}} \Gamma \Big\| \mathcal{A}_K^{-1} \Big\|_2 \max_{k=1,\ldots,K} \|\phi_k\|_{L^2_\mu} \Big). \end{aligned}$$

where $\epsilon_K = \max_{n=1,\dots,N} \left\| \Pi_K V(n,\tilde{X}) - V(n,\tilde{X}) \right\|_{L^2_\mu}$.

Proof of theorem 2.4.1. The proof will follow by induction. Notice that $V(N,x) = \tilde{V}(N,x)$ for all $x \in D$. For $n < N$ we have

$$\begin{aligned}
\tilde{V}(n,x) - V(n,x) &= f(n,x,\hat{u}_n(\tilde{X})) + \mathbb{E}_{\hat{u}(n,x)}[\tilde{V}(n+1,X_{n+1})] - V(n,x) \\
&\quad + \mathbb{E}_{\hat{u}(n,x)}[\hat{\Pi}_K^{n+1}\hat{V}(n+1,X_{n+1})] - \mathbb{E}_{\hat{u}(n,x)}[\hat{\Pi}_K^{n+1}\hat{V}(n+1,X_{n+1})] \\
&= \hat{\hat{V}}(n,x) - V(n,x) + \mathbb{E}_{\hat{u}(n,x)}[\tilde{V}(n+1,X_{n+1})] - \\
&\quad \mathbb{E}_{\hat{u}(n,x)}[\hat{\Pi}_K^{n+1}\hat{V}(n+1,X_{n+1})],
\end{aligned}$$

where we added and subtracted $\mathbb{E}_{\hat{u}_n(\tilde{X})}\left[\hat{\Pi}_K^{n+1}\hat{V}(n+1,X_{n+1})\right]$ and used that $\hat{\hat{V}}(n,x) = f(x,\hat{u}(n,x)) + \mathbb{E}_{\hat{u}(n,x)}\left[\hat{\Pi}_K^{n+1}\hat{V}(n+1,X_{n+1})\right]$. Using the triangular inequality

$$\begin{aligned}
\left\| \tilde{V}(n,\tilde{X}) - V(n,\tilde{X}) \right\|_{L^2_{e,N-n}} &\leq \left\| \hat{\hat{V}}(n,\tilde{X}) - V(n,\tilde{X}) \right\|_{L^2_{e,N-n}} \\
&\quad + \left\| \mathbb{E}_{\hat{u}(\tilde{X})}\left[\tilde{V}(n+1,X_{n+1}) - \hat{\Pi}_K^{n+1}\hat{V}(n+1,X_{n+1}) \right] \right\|_{L^2_{e,N-n}}.
\end{aligned} \tag{2.4.19}$$

The first term in (2.4.19) has been bounded in theorem 2.3.3, (2.3.11)-(2.3.14), as follows

$$\begin{aligned}
\left\| \hat{\hat{V}}(n,\tilde{X}) - V(n,\tilde{X}) \right\|_{L^2_{e,N-n}} &\leq \bar{R} \left\| \hat{V}(n+1,\tilde{X}) - V(n+1,\tilde{X}) \right\|_{L^2_{e,N-n-1}} + \bar{R}\epsilon_K \\
&\quad + \bar{R}\frac{\sqrt{K}}{\sqrt{M}}\Gamma \left\| \mathcal{A}_K^{-1} \right\|_2 \max_{k=1,\dots,K} \|\phi_k\|_{L^2_\mu}.
\end{aligned} \tag{2.4.20}$$

The second term in (2.4.19) can be bounded making use of Assumption 1 and Lemma

2.2.8:

$$\begin{aligned}
&\left\|\mathbb{E}_{\hat{u}(\tilde{X})}\left[\tilde{V}(n+1,X_{n+1})-\hat{\Pi}_K^{n+1}\hat{V}(n+1,X_{n+1})\right]\right\|_{L^2_{e,N-n}}\\
&\leq \bar{R}\left\|\tilde{V}(n+1,\tilde{X})-\hat{\Pi}_K^{n+1}\hat{V}(n+1,\tilde{X})\right\|_{L^2_{e,N-n}}\\
&\leq \bar{R}\left\|\tilde{V}(n+1,\tilde{X})-V(n+1,\tilde{X})\right\|_{L^2_{e,N-n-1}}\\
&+\bar{R}\left\|V(n+1,\tilde{X})-\hat{\Pi}_K^{n+1}\hat{V}(n+1,\tilde{X})\right\|_{L^2_{e,N-n}},
\end{aligned} \tag{2.4.21}$$

where the second inequality has been obtained using triangular inequality with the aim of highlighting the term representing the propagation of the error. The second term in (2.4.21) has been estimated in (2.3.14):

$$\begin{aligned}
\left\|V(n+1,\tilde{X})-\hat{\Pi}_K\hat{V}(n+1,\tilde{X})\right\|_{L^2_{e,N-n}}&\\
\leq \epsilon_K+\left\|V(n+1,\tilde{X})-\hat{V}(n+1,\tilde{X})\right\|_{L^2_{e,N-n-1}}&\\
+\frac{\sqrt{K}}{\sqrt{M}}\Gamma\left\|\mathcal{A}_K^{-1}\right\|_2\max_{k=1,\dots,K}\|\phi_k\|_{L^2_\mu}.&
\end{aligned} \tag{2.4.22}$$

Combining estimates (2.4.20)-(2.4.22) we obtain

$$\begin{aligned}
\left\|\tilde{V}(n,\tilde{X})-V(n,\tilde{X})\right\|_{L^2_{e,N-n}}&\leq \bar{R}\left\|\tilde{V}(n+1,\tilde{X})-V(n+1,\tilde{X})\right\|_{L^2_{e,N-n-1}}\\
&+2\bar{R}Z\\
&+2\bar{R}\left\|V(n+1,\tilde{X})-\hat{V}(n+1,\tilde{X})\right\|_{L^2_{e,N-n-1}},
\end{aligned} \tag{2.4.23}$$

where

$$Z:=\epsilon_K+\frac{\sqrt{K}}{\sqrt{M}}\Gamma\left\|\mathcal{A}_K^{-1}\right\|_2\max_{k=1,\dots,K}\|\phi_k\|_{L^2_\mu}.$$

From theorem 2.3.3 we have

$$\left\|V(n+1,\tilde{X})-\hat{V}(n+1,\tilde{X})\right\|_{L^2_{e,N-n-1}}\leq \bar{R}\frac{\bar{R}^{N-n-1}-1}{\bar{R}-1}Z.$$

Denote $\gamma_n := \left\| \tilde{V}(n, \tilde{X}) - V(n, \tilde{X}) \right\|_{L^2_{e,N-n}}$. Solving the recursion for γ_n in (2.4.23) yields:

$$\begin{aligned} \gamma_n &\leq \bar{R}\gamma_{n+1} + 2\bar{R}Z\Big(\frac{\bar{R}^{N-n} - \bar{R}}{\bar{R} - 1} + 1\Big) \\ &\leq \frac{2\bar{R}}{(\bar{R}-1)^2} Z\Big((N-n)\bar{R}^{N-n+1} - (N-n+1)\bar{R}^{N-n} - 1\Big), \end{aligned} \tag{2.4.24}$$

where we used that $\gamma_N = 0$ and $\hat{V}(N, x) = V(X, x)$. The statement of the theorem follows. □

2.5 Performance Iteration

In order to discuss the case of Performance Iteration, let us introduce an extension of the random projection operator suitable to work in the more involved setting required by this algorithm. In short, notice that this approach might be seen as a combination of Value Iteration and Forward Iteration, where at each step an assessment of the performance of the control policy on one simulated path is used as dependent variable in the regression approximation. In order to extend our framework we need therefore to introduce a more general projection operator but, before doing so, let us describe the method more in detail.

2.5.1 The method

Policy iteration of Longstaff and Schwartz [51] (here called *Performance Iteration* to distinguish it from classical policy iteration methods in control of Markov chains [48]) improved upon Tsitsiklis and Van Roy [76] by reducing the propagation of the error caused by the estimation of the continuation value in the recursive formula for value function of an optimal stopping problem. In their algorithm, at each step of

the dynamic programming backward procedure the continuation value is computed as a conditional expectation of realised payoffs obtained by following the estimated optimal strategy. This approach relies on the fact that the control (stopping time) manifests itself through the objective functional but does not affect the dynamics of the underlying process.

Our iterative procedure is based on the dynamic programming equation for the performance measure (instead of that for the value function):

$$\begin{cases} J(N, X_N) & = g(X_N) \\ J\Big(n, (X_s, u_s^*)_{s=n+1,\dots,N}\Big) & = f(n, X_n, u_n^*(X_n)) + J\Big(n+1, (X_s, u_s^*)_{s=n+1,\dots,N}\Big), \end{cases} \tag{2.5.25}$$

where the control is given by

$$u_n^*(x) = \underset{u \in \mathcal{D}^U}{\operatorname{argmax}} \Big\{ f(n, x, u) + \mathbb{E}\Big[J(n+1, (X_s, u_s^*)_{s=n+1,\dots,N}) \Big| X_n = x, u_n = u \Big] \Big\} \tag{2.5.26}$$

and, with an abuse of notation (as the control is not defined at N), $(X_s, u_s^*)_{s=n+1,\dots,N}$ denotes the process $(X_s)_{s=n,\dots,N}$ controlled by the process $(u_s^*)_{n,\dots,N-1}$, i.e.,

$$X_{s+1} = \varphi(s, X_s, \xi_s, u_s^*(X_s)), \qquad s = n, \dots, N-1.$$

The value function is recovered by conditioning the performance measure on $X_n = x$,

$$V(n, x) = \mathbb{E}\Big[J\Big(n, (X_s, u_s^*)_{s=n,\dots,N}\Big) | X_n = x \Big]. \tag{2.5.27}$$

The above conditioning can be viewed as a projection which will be particularly useful when assessing $V(n, \tilde{X})$ with $\tilde{X} \sim \mu$ as an element of L^2_μ. Note that even though the conditional expectations in (2.5.26) and (2.3.9) seem different, by the tower property for conditional expectations

$$\begin{aligned} \mathbb{E}[V(n+1, X_{n+1}) | \mathcal{F}_n] &= \mathbb{E}[\mathbb{E}[\sum_{s=n+1}^{N} f(X_s, u_s) + g(X_N) | \mathcal{F}_{n+1}] | \mathcal{F}_n] \\ &= \mathbb{E}[J(n+1, (X_s, u_s)_{s=n}^N) | \mathcal{F}_n], \end{aligned}$$

which implies that even though we iterate over Js rather than Vs, the regression approximation will be targeting the same object $\mathbb{E}_{n,x,u}\left[V(n+1, X_{n+1})\right]$.

Looking at the update rule that characterises the Performance Iteration approach it can be immediately seen that its main advantage compared to Value Iteration is that the error committed in the estimation of the conditional expectation in (2.5.26) is not directly propagated to the following time step (i.e. from $n+1$ to n). We will further discuss this in chapter 5.

A direct implementation of equation (2.5.25) allows to iterate over J's rather than V's but requires the computation of

$$J(n, (X_s, u_s^*)_{s=n,\dots,N}) = \sum_{s=n}^{N-1} f(s, X_s, u_s^*(X_s)) + g(X_N)$$

after u_n^* is established, which numerically means resimulating the path from time n to the terminal time for each training point, incurring a computational cost. For details see Algorithm 2.

Remark 2.5.1. *For all points $(\tilde{X}_n^m)$ at time n, the trajectories $\{X_j^m\}_{j=n+1}^N$ must be recomputed as it is unlikely that the controlled process governed by $u_n(\tilde{X}_n^m)$ matches exactly X_{n+1}^m.*

2.5.2 Projection operator in bigger spaces

In order to assess the convergence of the Performance Iteration algorithm, we need to extend the notation used in the previous sections. As with each training point X_n^m we need to simulate a controlled path up to time N, we consider spaces

$$L^2_{\xi,n} = L^2_\mu \times L^2\Big((0,1)^{N-n}, \lambda^{N-n}\Big), \qquad n < N,$$

where λ is the Lebesgue measure on $(0,1)$. The elements of this space will be denoted $(\tilde{X}, \xi_n^n, \dots, \xi_{N-1}^n)$, where ξ's correspond to uniform random variables driving

Algorithm 2 Regress-later Monte Carlo algorithm (RLMC) - Performance Iteration

input: M, K, μ, $\{\phi\}_{k=1}^{K}$

Pre-compute the inverse of the matrix $\mathcal{A}_K$

Generate i.i.d. training points $\{\tilde{X}_N^m\}_{m=1}^M$ accordingly to the distribution μ.

Initialise the performance measure $\hat{J}(N,m) = g(\tilde{X}_N^m), \quad \forall m$

for n=N-1 to 0 **do**

  $\hat{\alpha}^{n+1} = \mathcal{A}_K^{-1} \frac{1}{M} \sum_{m=1}^{M} \Big[J(n+1, \tilde{X}_{n+1}^m) \phi(\tilde{X}_{n+1}^m) \Big]$

  Generate a new layer of i.i.d. training points $\{\tilde{X}_n^m\}_{m=1}^M$ accordingly to the distribution μ.

  Compute the control mapping $\hat{u}_n(x) = \text{argmax}_{u \in \mathcal{D}^U} \Big\{ f(j,x,u) + \sum_{k=1}^{K} \hat{\alpha}_k^{n+1} \hat{\phi}_k^n(x,u) \Big\}$

  for m=1 to M **do**

    Generate $(\xi_n^m, \dots, \xi_{N-1}^m) \sim U(0,1)$

    $X_n = \tilde{X}_n^m$

    for j=s to N-1 **do**

      $X_{s+1} = \varphi\Big(s, X_s, \xi_s^m, \hat{u}_j(X_s^m)\Big)$

    Set $\hat{J}(n,m) = \sum_{s=n}^{N-1} f\Big(n, X_s, \hat{u}_s(X_s)\Big) + g(X_N)$

output: $\{\hat{\alpha}_k^n\}_{n,k=1}^{N,K}$

the dynamics of the controlled Markov process (2.0.1). To streamline notation, we also set $L^2_{\xi,N} = L^2_\mu$. The space $L^2_{M,n} = (L^2_\mu)^{Mn}$ which collects all the randomness involved in computation of $\hat{\alpha}^n$ in the Value Iteration case, gets a counterpart $L^2_{\chi,n}$ defined by induction as follows:

$$L^2_{\chi,N} = \Big(L^2_\mu\Big)^M$$

as no path is generated at time N, and

$$L^2_{\chi,n} = L^2_{\chi,n+1} \times \Big(L^2_{\xi,n}\Big)^M.$$

In parallel, we define arguments of the functions in the above spaces $L^2_{\chi,n}$: $\tilde{\mathfrak{Y}}_N = (\tilde{X}^1_N, \ldots, \tilde{X}^M_N)$ and, for $n < N-1$

$$\tilde{\mathfrak{Y}}_n = \tilde{\mathfrak{Y}}_{n+1} \times \left(\tilde{X}^m_n, \xi^{n,m}_n, \ldots, \xi^{n,m}_{N-1}\right)_{m=1}^{M}.$$

Finally, we introduce two counterparts of $L^2_{e,n}$. The first one to assess performance of strategies:

$$L^2_{F,n} = L^2_{\chi,n+1} \times L^2_{\xi,n} \tag{2.5.28}$$

as the output of the algorithm is the control strategy which must be assessed by applying it between time n and N with initial value $X_n = \tilde{X} \sim \mu$ and the remaining randomness used to obtain the trajectory until time N. The second counterpart of $L^2_{e,n}$ is a subspace of $L^2_{F,n}$ which is used in assessing an estimated value function and regression coefficients $\hat{\alpha}^n$:

$$L^2_{f,n} = L^2_{\chi,n+1} \times L^2_{\mu}.$$

In the Value Iteration case, the estimated control $\hat{u}_n$ depends on all the training points at future times, i.e., on $\tilde{\mathfrak{X}}_{n+1}$. Here, these controls involve further random variables associated with simulation of the trajectory starting at every training point, which is indicated in $\tilde{\mathfrak{Y}}_{n+1}$. With an abuse of notation, we will write

$$(X_s, \hat{u}_s)_{s=n,\ldots,N} \tag{2.5.29}$$

to mean the sequence of random variables dependent on $\tilde{\mathfrak{Y}}_{n+1}$ in the following way

$$X_{s+1} = \varphi\Big(s, X_s, \xi_s, \hat{u}_s(\tilde{\mathfrak{Y}}_{s+1}; X_s)\Big) \tag{2.5.30}$$

with X_n given and $(\xi_s)_{s=n}^{N-1}$ a sequence of i.i.d $U(0,1)$ random variables independent from $\tilde{\mathfrak{Y}}_{n+1}$. Therefore, $J\Big(n, (X_s, \hat{u}_s)_{s=n,\ldots,N})\Big) \in L^2_{F,n}$, where $(X_n, \xi_n, \ldots, \xi_{N-1})$ are variables corresponding to the space $L^2_{\xi,n}$. Notice that $J\Big(n, (X_s, \hat{u}_s)_{s=n,\ldots,N})\Big)$ is a pathwise evaluation of the control policy, and depends therefore on $\tilde{\mathfrak{Y}}_{n+1}$ only through $\hat{u}$; this is in contrast with the Value Iteration case, where the error propagates in time also directly through the value function approximation $\hat{V}(n+1, \cdot)$.

We extend the projection operator as follows: for $h(\tilde{\mathfrak{Y}}_{n+1}; x, \xi_n, \dots, \xi_{N-1}) \in L^2_{F,n}$ we set

$$\Pi^n_K h(\tilde{\mathfrak{Y}}_{n+1}; x, \xi_n, \dots, \xi_{N-1}) := \sum_{k=1}^{K} \alpha^n_k \phi_k(x),$$

where

$$L^2_{\chi,n+1} \ni \hat{\boldsymbol{\alpha}}^n = \boldsymbol{\alpha}^n(\tilde{\mathfrak{Y}}_{n+1}) = \mathcal{A}_K^{-1} \mathbb{E}_{\xi,n}\Big[h(\tilde{\mathfrak{Y}}_{n+1}; \tilde{X}, \xi_n, \dots, \xi_{N-1})\boldsymbol{\phi}(\tilde{X})\Big|\tilde{\mathfrak{Y}}_{n+1}\Big]$$

and $\mathbb{E}_{\xi,n}$ is the expectation linked to the space $L^2_{\xi,n}$. This is an orthogonal projection in $L^2_{F,n} = L^2_{\chi,n+1} \times L^2_{\xi,n}$ on the space $lin\Big(\phi_1(\tilde{X}), \dots, \phi_K(\tilde{X})\Big)$, where $\tilde{X}$ is the variable corresponding to the L^2_μ part of $L^2_{\xi,n}$.

The Monte Carlo projection operator is defined as follows. For a sequence of i.i.d random variables $(\tilde{X}^1_n, \dots, \tilde{X}^M_n)$ and i.i.d $(\xi^{n,m}_j)_{j=n,\dots,N-1;\ m=1,\dots,M} \sim U(0,1)$, we set

$$\hat{\Pi}^n_K h(\tilde{\mathfrak{Y}}_{n+1}; x, \xi_n, \dots, \xi_{N-1}) := \sum_{k=1}^{K} \hat{\alpha}^n_k \phi_k(x), \tag{2.5.31}$$

where

$$L^2_{\chi,n} \ni \hat{\boldsymbol{\alpha}}^n = \mathcal{A}_K^{-1} \frac{1}{M} \sum_{m=1}^{M} h(\tilde{\mathfrak{Y}}_{n+1}; \tilde{X}^m_n, \xi^{n,m}_n, \dots, \xi^{n,m}_{N-1})\boldsymbol{\phi}(\tilde{X}^m_n). \tag{2.5.32}$$

Notice that $\hat{\Pi}^n_K h(\tilde{\mathfrak{Y}}_{n+1}; \cdot) \in L^2_{f,n-1}$.

We introduce now an extension of lemma 2.2.5 and 2.2.8 for functions living in the relevant spaces for Performance Iteration. Proofs of these lemmas are a straightforward generalisation of those in section 2.2.4 and will not be presented.

Lemma 2.5.2 (Projection error)**.** *For $h \in L^2_{F,N-n}$, the error of the random projection operator is bounded as follows:*

$$\Big\|\hat{\Pi}^{N-n}_K h(\tilde{\mathfrak{X}}_n; \cdot) - \Pi^{N-n}_K h(\tilde{\mathfrak{X}}_n; \cdot)\Big\|_{L^2_{F,N-n}} \leq \Big\|\mathcal{A}_K^{-1}\Big\|_2 \frac{1}{\sqrt{M}} SDev_{\substack{\tilde{X}\sim\mu,\\ \xi_{n+1},\dots,\xi_N\sim\lambda}}\Big(h(\tilde{\mathfrak{X}}_n\,; \tilde{X})\boldsymbol{\phi}(\tilde{X})\Big),$$

where

$$SDev_{\substack{\tilde{X}\sim\mu,\\ \xi_{n+1},\dots,\xi_N\sim\lambda}}\Big(h(\tilde{\mathfrak{X}}_n\,; \tilde{X})\boldsymbol{\phi}(\tilde{X})\Big) := \left(\sum_{k=1}^{K} Var_{\substack{\tilde{X}\sim\mu,\\ \xi_{n+1},\dots,\xi_N\sim\lambda}}\Big(h(\tilde{\mathfrak{X}}_n\,; \tilde{X})\phi_k(\tilde{X})\Big)\right)^{1/2},$$

and $\|\mathcal{A}_K^{-1}\|_2 = \max\{\|\mathcal{A}_K^{-1}x\|_2 : \ x \in \mathbb{R}^K, \ \|x\|_2 = 1\}$ *is the matrix operator norm of* $\mathcal{A}_K^{-1}$.

Lemma 2.5.3 (Bound on conditional expectation)**.** *For any measurable function* $\hat{u}(\tilde{\mathfrak{Y}}_{n+1}, \tilde{X}) : L^2_{f,N-n} \times L^2_\mu \mapsto \mathcal{U}_n$ *and* $h \in L^2_{F,N-n}$*, we have the following bound on the norm of the conditional expectation*

$$\begin{aligned}\Big\|\mathbb{E}\Big[h(\tilde{\mathfrak{X}}_n; X_{n+1})\Big|X_n = \tilde{X}, u_n =& \hat{u}_n(\tilde{\mathfrak{X}}_n, \tilde{X})\Big]\Big\|_{L^2_{F,N-n}} \\ &\leq \sup_{u\in\mathcal{U}_n}\Big\|\mathbb{E}\Big[h(\tilde{\mathfrak{X}}_n; X_{n+1})\Big|X_n = \tilde{X}, u_n = u\Big]\Big\|_{L^2_{F,N-n}} \\ &\leq \bar{R}\|h\|_{L^2_{F,N-n}},\end{aligned}$$

where

$$\mathbb{E}\Big[h(\tilde{\mathfrak{Y}}_{n+1}; X_{n+1})\Big|X_n = x, u_n = \hat{u}(n,x)\Big] = \int_{\mathcal{D}} h(\tilde{\mathfrak{Y}}_{n+1}; y)\ r(n, x, \hat{u}(x); y)\mu(dy).$$

2.5.3 Convergence results

We now present a theorem that estimates the error between the estimated and the true value function. Recall that we indicate by $\bar{R}$ the uniform bound on the transition density of X with respect to the measure μ, and by Γ the upper bound for the value function.

Consider the exact performance of the estimated optimal strategy $\hat{u}_n$ computed in Algorithm 2:

$$\begin{aligned}\tilde{V}(n,x) =& \mathbb{E}\Big[J\Big(n, (X_s, \hat{u}_s)_{s=n}^N\Big)\Big|X_n = x\Big] \\ =& \mathbb{E}\Big[\sum_{s=n}^N f(s, X_s, \hat{u}_n(X_s)) + g(X_N)\Big|X_n = x\Big] \\ =& f(n, x, \hat{u}(n,x)) + \mathbb{E}\Big[\tilde{V}(n+1, X_{n+1})\Big|X_n = x, u_n = \hat{u}_n(x)\Big].\end{aligned}$$

This is an analogous quantity as studied in the previous section concerned with the forward evaluation of a strategy extracted in the Value Iteration scheme. Note that $\tilde{V}(n,\cdot) \in L^2_{f,n}$ due to the random points used in computing the strategy $\hat{u}_n$.

Theorem 2.5.4. *Under Assumptions 1 and 2, for all $n = 0, 1, \dots, N$, we have:*

$$\left\|\tilde{V}(n,\tilde{X}) - V(n,\tilde{X})\right\|_{L^2_{f,n}} \leq 2\bar{R}\frac{(3\bar{R})^{N-n}-1}{3\bar{R}-1}\Big(\epsilon_K + \frac{\sqrt{K}}{\sqrt{M}}\Gamma\left\|\mathcal{A}_K^{-1}\right\|_2 \max_{k=1,\dots,K}\|\phi_k\|_{L^2_\mu}\Big)$$

where $\epsilon_K = \max_{n=1,\dots,N}\left\|\Pi_K V(n,\tilde{X}) - V(n,\tilde{X})\right\|_{L^2_\mu}$.

Proof of theorem 2.5.4. We have

$$\begin{aligned}
&\tilde{V}(n,\tilde{X}) - V(n,\tilde{X}) \\
&= f(n, X_n, \hat{u}_n(\tilde{X})) + \mathbb{E}_{\hat{u}_n(\tilde{X})}\Big[\tilde{V}(n+1,\cdot)\Big] - f(n,\tilde{X},u_n^*(\tilde{X})) - \mathbb{E}_{u_n^*(\tilde{X})}\Big[V(n+1,\cdot)\Big] \\
&\quad + \mathbb{E}_{\hat{u}_n(\tilde{X})}\Big[\hat{\Pi}_K^{n+1}J\Big(n+1,(X_s,\hat{u}_s)_{s=n+1}^N\Big)\Big)\Big] \\
&\quad - \mathbb{E}_{\hat{u}_n(\tilde{X})}\Big[\hat{\Pi}_K^{n+1}J\Big(n+1,(X_s,\hat{u}_s)_{s=n+1}^N\Big)\Big)\Big],
\end{aligned}$$

where we have added and subtracted $\mathbb{E}_{\hat{u}_n(\tilde{X})}\Big[\hat{\Pi}_K^{n+1}J\Big(n+1,(X_s,\hat{u}_s)_{s=n+1}^N\Big)\Big]$. Since

$$\hat{u}(n,x) = \underset{u\in\mathcal{U}_n}{\operatorname{argmax}}\Big\{f(n,x,u) + \mathbb{E}_{n,x,u}\Big[\hat{\Pi}_K^{n+1}J\Big(n+1,(X_s,\hat{u}_s)_{s=n+1}^N\Big)\Big]\Big\},$$

then:

$$\begin{aligned}
\tilde{V}(n,\tilde{X}) - V(n,\tilde{X}) \geq\ & \mathbb{E}_{\hat{u}_n(\tilde{X})}\Big[\tilde{V}(n+1,X_{n+1}) - \hat{\Pi}_K^{n+1}J\Big(n+1,(X_s,\hat{u}_s)_{s=n+1}^N\Big)\Big] \\
&+ \mathbb{E}_{u_n^*(\tilde{X})}\Big[\hat{\Pi}_K^{n+1}J\Big(n+1,(X_s,\hat{u}_s)_{s=n+1}^N\Big) - V(n+1,X_{n+1})\Big].
\end{aligned}$$

Similarly, since $u^*(n,x) = \operatorname{argmax}_{u\in\mathcal{U}_n}\Big\{f(n,x,u_n) + \mathbb{E}_{n,x,u}\Big[V(n+1,X_{n+1})\Big]\Big\}$, we have:

$$\begin{aligned}
\tilde{V}(n,\tilde{X}) - V(n,\tilde{X}) \leq\ & \mathbb{E}_{\hat{u}_n(\tilde{X})}\Big[\tilde{V}(n+1,X_{n+1}) - \hat{\Pi}_K^{n+1}J\Big(n+1,(X_s,\hat{u}_s)_{s=n+1}^N\Big)\Big] \\
&+ \mathbb{E}_{\hat{u}_n(\tilde{X})}\Big[\hat{\Pi}_K^{n+1}J\Big(n+1,(X_s,\hat{u}_s)_{s=n+1}^N\Big) - V(n+1,X_{n+1})\Big].
\end{aligned}$$

Collecting the previous two inequalities we can conclude, using the triangular

inequality, that:

$$\begin{aligned}\Big\|\tilde{V}(n,\tilde{X})-V(n,\tilde{X})\Big\|_{L^2_{f,n}}&\\ \leq&\Big\|\mathbb{E}_{\hat{u}_n(\tilde{X})}\Big[\tilde{V}(n+1,X_{n+1})-\hat{\Pi}_K^{n+1}J\Big(n+1,(X_s,\hat{u}_s)_{s=n+1}^N\Big)\Big]\Big\|_{L^2_{f,n}}\\ &+\Big\|\sup_{u\in\mathcal{D}^U}\Big\{\mathbb{E}_{n,\tilde{X},u}\Big[\hat{\Pi}_K^{n+1}J\Big(n+1,(X_s,\hat{u}_s)_{s=n+1}^N\Big)-V(n+1,X_{n+1})\Big]\Big\}\Big\|_{L^2_{f,n}}.\end{aligned} \tag{2.5.33}$$

Using Assumption 1 and Lemma 2.5.3, (2.5.33) reads

$$\begin{aligned}\Big\|\tilde{V}(n,\tilde{X})-V(n,\tilde{X})\Big\|_{L^2_{f,n}}&\\ \leq&\bar{R}\Big\|\tilde{V}(n+1,\tilde{X})-\hat{\Pi}_K^{n+1}J\Big(n+1,(X_s,\hat{u}_s)_{s=n+1}^N\Big)\Big|_{X_{n+1}=\tilde{X}}\Big\|_{L^2_{f,n+1}}\\ &+\bar{R}\Big\|\hat{\Pi}_K^{n+1}J\Big(n+1,(X_s,\hat{u}_s)_{s=n+1}^N\Big)\Big|_{X_{n+1}=\tilde{X}}-V(n+1,\tilde{X})\Big\|_{L^2_{f,n}}\\ \leq&\bar{R}\Big\|\tilde{V}(n+1,\tilde{X})-V(n+1,\tilde{X})\Big\|_{L^2_{f,n}}\\ &+2\bar{R}\Big\|V(n+1,\tilde{X})-\hat{\Pi}_K\bar{J}\Big(n+1,(X_s,\hat{u}_s)_{s=n+1}^N\Big)\Big\|_{L^2_{f,n}},\end{aligned} \tag{2.5.34}$$

where the last line exploit the triangular inequality. The first term in (2.5.34) represents the propagation of error from future time steps. For the second we use the triangular inequality in order to derive a series of error terms:

$$\begin{aligned}\Big\|\hat{\Pi}_K^{n+1}\bar{J}\Big(n+1,&(X_s,\hat{u}_s)_{s=n+1}^N\Big)-V(n+1,X_{n+1})\Big\|_{L^2_{f,n}}\\ &\leq\Big\|\hat{\Pi}_K^{n+1}\bar{J}\Big(n+1,(X_s,\hat{u}_s)_{s=n+1}^N-\Pi_K^{n+1}J\Big(n+1,(X_s,\hat{u}_s)_{s=n+1}^N\Big)\Big)\Big\|_{L^2_{f,n}}\\ &\quad+\Big\|\Pi_K^{n+1}J\Big(n+1,(X_s,\hat{u}_s)_{s=n+1}^N\Big)-\Pi_K^{n+1}\tilde{V}(n+1,\tilde{X})\Big\|_{L^2_{f,n}}\\ &\quad+\Big\|\Pi_K^{n+1}\tilde{V}(n+1,\tilde{X})-\Pi_K^{n+1}V(n+1,\tilde{X})\Big\|_{L^2_{f,n}}\\ &\quad+\Big\|\Pi_K^{n+1}V(n+1,\tilde{X})-V(n+1,\tilde{X})\Big\|_{L^2_{f,n}}.\end{aligned} \tag{2.5.35}$$

The first term in (2.5.35) can be bounded using Lemma 2.5.3, 2.5.2 and Assumption 2:

$$\Big\|\Pi_K^{n+1} J\Big(n+1,(X_s,\hat{u}_s)_{s=n+1}^N\Big) - \hat{\Pi}_K^{n+1} J\Big(n+1,(X_s,\hat{u}_s)_{s=n+1}^N\Big)\Big\|_{L^2_{f,n}} \leq \frac{\sqrt{K}}{\sqrt{M}}\bar{\Gamma}\Big\|\mathcal{A}_K^{-1}\Big\|_2 \max_{k=1,\dots,K}\|\phi_k\|_{L^2_\mu}. \tag{2.5.36}$$

The second term in (2.5.35) can be computed as follow:

$$\begin{aligned}
&\Big\|\Pi_K^{n+1}\tilde{V}(n+1,\tilde{X}) - \Pi_K^{n+1} J\Big(n+1,(X_s,\hat{u}_s)_{s=n+1}^N\Big)\Big\|_{L^2_{f,n}} \\
&\quad= \Big\|\mathcal{A}_K^{-1}\mathbb{E}_{z\sim\mu}\Big[\tilde{V}(n+1,z)\boldsymbol{\phi}(z)\Big]\boldsymbol{\phi}(\tilde{X}) - \mathcal{A}_K^{-1}\mathbb{E}_{\substack{z\sim\mu,\\ \xi_{n+1},\dots,\xi_N\sim\lambda}}\Big[\\
&\qquad J\Big(n+1,(\varphi(s,X_s,\hat{u}_s,\xi_s),\hat{u}_{s+1})_{s=n}^N|_{X_{n+1}=z}\Big)\boldsymbol{\phi}(z)\Big]\boldsymbol{\phi}(\tilde{X})\Big\|_{L^2_{f,n}} \\
&\quad= \Big\|\mathcal{A}_K^{-1}\mathbb{E}_{z\sim\mu}\Big[\tilde{V}(n+1,z) - \mathbb{E}_{\xi_{n+1},\dots,\xi_N\sim\lambda}\Big[\\
&\qquad J\Big(n+1,(\varphi(s-1,X_s,\hat{u}_s,\xi_s),\hat{u}_{s+1})_{s=n}^N\Big)\Big|_{X_{n+1}=z}\Big]\times\boldsymbol{\phi}(z)\Big]\boldsymbol{\phi}(\tilde{X})\Big\|_{L^2_{f,n}} \\
&\quad= \Big\|\mathcal{A}_K^{-1}\mathbb{E}_{z\sim\mu}\Big[\Big(\tilde{V}(n+1,z) - \tilde{V}(n+1,z)\Big)\boldsymbol{\phi}(z)\Big]\boldsymbol{\phi}(\tilde{X})\Big\|_{L^2_{f,n}} = 0,
\end{aligned} \tag{2.5.37}$$

where we used the tower property of conditional expectations and the notation introduced in (2.5.29)-(2.5.30). The third term in (2.5.35) can be bounded making use of arguments similar to remark 2.2.8

$$\Big\|\Pi_K^{n+1}V(n+1,\tilde{X}) - \Pi_K^{n+1}\tilde{V}(n+1,\tilde{X})\Big\|_{L^2_{f,n}} \leq \Big\|V(n+1,\tilde{X}) - \tilde{V}(n+1,\tilde{X})\Big\|_{L^2_{f,n}}, \tag{2.5.38}$$

note that we obtain an additional term representing the error propagated from future time steps. The last term in (2.5.35) can be bounded can be bounded by ϵ_K

$$\Big\|V(n+1,\tilde{X}) - \Pi_K^{n+1}V(n+1,\tilde{X})\Big\|_{L^2_{f,n}} \leq \Big\|V(n+1,\tilde{X}) - \Pi_K^{n+1}V(n+1,\tilde{X})\Big\|_{L^2_\mu} \leq \epsilon_K; \tag{2.5.39}$$

Collecting the bounds we obtained for (2.5.34) and (2.5.35) we obtain

$$\begin{aligned}\left\|\tilde{V}(n,\tilde{X})-V(n,\tilde{X})\right\|_{L^2_{f,n}} &\leq 3\bar{R}\left\|\tilde{V}(n+1,\tilde{X})-V(n+1,\tilde{X})\right\|_{L^2_{f,n}} \\ &\quad + 2\bar{R}\epsilon_K + 2\bar{R}\frac{\sqrt{K}}{\sqrt{M}}\bar{\Gamma}\left\|\mathcal{A}_K^{-1}\right\|_2 \max_{k=1,\dots,K}\left\|\phi_k\right\|_{L^2_\mu}\end{aligned} \tag{2.5.40}$$

Denoting by $\beta_n := \left\|\tilde{V}(n,\tilde{X})-V(n,\tilde{X})\right\|_{L^2_{f,n}}$ and $Z := \epsilon_K + \frac{\sqrt{K}}{\sqrt{M}}\bar{\Gamma}\left\|\mathcal{A}_K^{-1}\right\|_2 \max_{k=1,\dots,K}\left\|\phi_k\right\|_{L^2_\mu}$ we conclude computing

$$\beta_n \leq 3\bar{R}\beta_{n+1} + 2\bar{R}Z \leq 2\bar{R}\sum_{s=0}^{N-n-1}(3\bar{R})^s Z = 2\bar{R}\frac{(3\bar{R})^{N-n}-1}{3\bar{R}-1}Z.$$

□

Corollary 2.5.5. *Under assumption of ortonormality of the basis functions $\phi_1,\dots,\phi_K$ in L^2_μ, we have:*

$$\left\|\tilde{V}(n,\tilde{X})-V(n,\tilde{X})\right\|_{L^2_{f,n}} \leq 2\bar{R}\frac{(3\bar{R})^{N-n}-1}{3\bar{R}-1}\left(\epsilon_K+\frac{\sqrt{K}}{\sqrt{M}}\Gamma\right)$$

Proof. Analogous to the proof of Corollary 2.3.5. □

Remark 2.5.6 (Policy vs. Performance Iteration)**.** *We decided to use the name Performance Iteration as opposed to policy iteration, in order to avoid misunderstandings. Often in Regression Monte Carlo literature the algorithm just presented is called policy iteration even though such name is used in the more general approximate dynamic programming literature to describe algorithms which iterate over controls rather than over the performance measure.*

Remark 2.5.7 (Truncation)**.** *Notice that the Performance Iteration does not require any truncation of the estimations, contrary to Value Iteration for which we need to introduce truncation in order to obtain convergence.*

We now explore further the error bound for Performance Iteration by defining $\tilde{\epsilon}_K = \max_{n=1,\dots,N} \left\| \Pi_K \tilde{V}(n,\tilde{X}) - \tilde{V}(n,\tilde{X}) \right\|_{L^2_\mu}$ to be the projection error on the estimated value function. In the following proposition we show how to sharpen the error bound in theorem 2.5.4 by means of an assumption on the projection error $\tilde{\epsilon}_K$.

Proposition 1. *Under Assumptions 1 and 2, and further assuming $\tilde{\epsilon}_K \leq \epsilon_K$, i.e. the projection error on the approximate value function is similar to the projection error on the true value function, we have, for all $n = 0, 1, \dots, N$, the following error bound for the value function estimated by the Performance Iteration procedure:*

$$\left\| \tilde{V}(n,\tilde{X}) - V(n,\tilde{X}) \right\|_{L^2_{f,n}} \leq 2\bar{R} \frac{(\bar{R})^{N-n} - 1}{\bar{R} - 1} \left(\epsilon_K + \frac{\sqrt{K}}{\sqrt{M}} \Gamma \left\| \mathcal{A}_K^{-1} \right\|_2 \max_{k=1,\dots,K} \|\phi_k\|_{L^2_\mu} \right)$$

Proof. The proof follows from theorem 2.5.4 which we use as reference.

With the same arguments used before we establish the inequality in (2.5.33):

$$\begin{aligned}
&\left\| \tilde{V}(n,\tilde{X}) - V(n,\tilde{X}) \right\|_{L^2_{f,n}} \\
&\qquad \leq \left\| \mathbb{E}_{\hat{u}_n(\tilde{X})} \left[\tilde{V}(n+1, X_{n+1}) - \hat{\Pi}_K^{n+1} J\left(n+1, (X_s, \hat{u}_s)_{s=n+1}^N\right) \right] \right\|_{L^2_{f,n}} \\
&\qquad\quad + \left\| \sup_{u \in \mathcal{D}^U} \left\{ \mathbb{E}_{n,\tilde{X},u} \left[\hat{\Pi}_K^{n+1} J\left(n+1, (X_s, \hat{u}_s)_{s=n+1}^N\right) - V(n+1, X_{n+1}) \right] \right\} \right\|_{L^2_{f,n}}.
\end{aligned}$$

Using Assumption 1 and Lemma 2.5.3, but applying the triangular inequality to the norm involving $\tilde{V}(n,\tilde{X})$ rather than the one involving $V(n,\tilde{X})$, we obtain an

alternative to (2.5.34):

$$\begin{aligned}\left\|\tilde{V}(n,\tilde{X})-V(n,\tilde{X})\right\|_{L^2_{f,n}}&\\ \leq&\bar{R}\left\|\tilde{V}(n+1,\tilde{X})-\hat{\Pi}_K^{n+1}J\Big(n+1,(X_s,\hat{u}_s)_{s=n+1}^N\Big)\Big|_{X_{n+1}=\tilde{X}}\right\|_{L^2_{f,n+1}}\\ &+\bar{R}\left\|\hat{\Pi}_K^{n+1}J\Big(n+1,(X_s,\hat{u}_s)_{s=n+1}^N\Big)\Big|_{X_{n+1}=\tilde{X}}-V(n+1,\tilde{X})\right\|_{L^2_{f,n}}\\ \leq&\ \bar{R}\left\|\tilde{V}(n+1,\tilde{X})-V(n+1,\tilde{X})\right\|_{L^2_{f,n}}\\ &+2\bar{R}\left\|\tilde{V}(n+1,\tilde{X})-\hat{\Pi}_K\bar{J}\Big(n+1,(X_s,\hat{u}_s)_{s=n+1}^N\Big)\right\|_{L^2_{f,n}}.\end{aligned} \tag{2.5.41}$$

The first term in (2.5.41) represents the propagation of the error from future time steps. The second can be bounded, as shown in (2.5.36), (2.5.39) by

$$\left\|\tilde{V}(n+1,\tilde{X})-\hat{\Pi}_K\bar{J}\Big(n+1,(X_s,\hat{u}_s)_{s=n+1}^N\Big)\right\|_{L^2_{f,n}}\leq\tilde{\epsilon}_K+\frac{\sqrt{K}}{\sqrt{M}}\Gamma\left\|\mathcal{A}_K^{-1}\right\|_2\max_{k=1,\dots,K}\|\phi_k\|_{L^2_\mu}. \tag{2.5.42}$$

Using the bound in (2.5.42) and similarly to the proof of theorem 2.5.4 we obtain the recursion:

$$\begin{aligned}\left\|\tilde{V}(n,\tilde{X})-V(n,\tilde{X})\right\|_{L^2_{f,n}}\leq&\ \bar{R}\left\|\tilde{V}(n+1,\tilde{X})-V(n+1,\tilde{X})\right\|_{L^2_{f,n}}\\ &+3\bar{R}\tilde{\epsilon}_K+2\bar{R}\frac{\sqrt{K}}{\sqrt{M}}\bar{\Gamma}\left\|\mathcal{A}_K^{-1}\right\|_2\max_{k=1,\dots,K}\left\|\phi_k\right\|_{L^2_\mu},\end{aligned} \tag{2.5.43}$$

which, along with the assumption that $\tilde{\epsilon}_K\geq\epsilon_K$, leads to the result presented in proposition 1 □

Chapter 3

RMC for inventory problems

3.1 Introduction

In this chapter, we study a class of problems that fall in between the class of optimal stopping-switching and the full stochastic control of Markov processes introduced in chapter 1. Inventory control problems are similar to switching problems with respect to the controlled deterministic index (or inventory indeed) dimension, while they present similarities with the full stochastic control problems with respect to the continuous support of both dimension, which prevents the use of standard Regression Monte Carlo techniques, and, for an efficient solution approach, it requires the use of Regress Later algorithm described in the previous chapter.

Recall that we consider a multidimensional process (X, I), with an exogenous (uncontrolled) p-dimensional random component (X_n), $n = 0, \ldots, N$, and an endogenous q-dimensional controlled component (I_n) with values in $[0, I^1_{\max}] \times \cdots \times [0, I^q_{\max}]$. We assume that (X_n) is a Markov process with transition density described in (1.1.1) and the discrete-time dynamics of I_n are given by

$$I_{n+1} = \varphi_I(n, X_n, I_n, u_n), \tag{3.1.1}$$

where φ_I is a Borel-measurable function and u_n is a q-dimensional control. The set of state dependent admissible controls $\mathcal{U}_{\{n:N\}}$ is given by feedback functions $(u_s(x,i) = u_s)_{s=n}^N$ such that certain problem specific constraints $\mathcal{W}_s$ are satisfied and $I_{s+1} = \varphi_I(s, X_n, I_s, u_s) \in [0, I^1_{\max}] \times \cdots \times [0, I^q_{\max}]$, $s \in [n, N-1]$. Without loss of generality, and for simplicity of presentation, we assume that the inventory and the controls are one-dimensional, i.e., $I_n \in [0, I_{\max}]$ and $u_n \in [\underline{u}, \overline{u}]$. The value function of the control problem is given by $V(n,x,i) = \sup_{(u_s)\in\mathcal{U}_{\{n:N\}}} \mathbb{E}[J(n,(X_s,I_s,u_s)_{s=n}^N)|X_n = x, I_n = i]$, where $J(n,(X_s,I_s,u_s)_{s=n}^N) = \sum_{s=n}^{N-1} f(s,X_s,I_s,u_s) + g(X_N,I_N)$. Because I_{n+1} in (3.1.1) is a deterministic function of (X_n, I_n, u_n), the Bellman equation for V reads:

$$\begin{cases} V(n,x,i) = \sup_{u\in\mathcal{U}_{\{n:N\}}} \Big\{ f(n,x,i,u) + \mathbb{E}\Big[V\Big(n+1, X_{n+1}, \varphi_I(n,x,i,u)\Big)|X_n = x\Big]\Big\}, \\ V(N,x,i) = g(x,i), \end{cases} \tag{3.1.2}$$

This property will be useful for numerical methods presented in section 3.2.

Our study of inventory control problems is motivated by the management of a limited amount of resources (which can be replenished or not), in order to maximise a measure of performance based on possibly both the inventory level itself and an exogenous process, which, for example could be the price or the demand for that particular resource. Recall that one of the main characteristics of this class of problems is that the inventory level influences the admissible controls since, when the inventory is full, it is not possible to accumulate resources further and, analogously, when it is empty, it is not possible to withdraw them. Examples include management of energy storages such as batteries [35], flywheels or pump-hydro stations for electricity production [81], underground caves for gas storage [12]; or management of warehouse/shop stocks subject to fluctuations in price and demand [69].

Recall that Regression Monte Carlo borrows from standard Monte Carlo methods its independence on the dynamics, as long as these can be simulated. This

feature prompted interest in applying Regression Monte Carlo to inventory control problems where the exogenous process is usually multi-dimensional and evolving on a non-compact state space preventing successful application of Markov chain approximation methods [48]. Although similar to the pricing of American/Bermudan options, the control in inventory problems does not affect the random dynamics, the action of the control changes the future inventory level making the problem significantly more difficult than just choosing among a set of possible future regimes to switch to. The simplest and most commonly used approach borrows from Markov chain approximations by discretising the inventory space and applying Regression Monte Carlo independently at each point of the discrete grid, see, e.g., Boogert and de Jong [12], Denault et al. [23], Kitapbayev et al. [47], Nadarajah et al. [60], effectively recovering the optimal switching framework which we discuss in chapter 4. However, due to separate regressions needed at each discretisation level of the inventory, the method is inefficient and the computational complexity becomes prohibitive when the dimension of the inventory space grows. We discuss this method in section 3.2.1.

A natural improvement to the inventory space discretisation is to include the inventory variable in the regression in order to be able to read from the resulting regression surface all information "contained in each discretisation level" and, without interpolating, those in-between. Algorithms in Carmona and Ludkovski [17], Denault et al. [23] require that the set of available controls is finite and small (in their examples, they allow for up to three values of control: increase, decrease, hold). In particular, the approach by Denault et al. [23] is sensitive to the size of the control set because for each possible control value it runs a separate regression at each time step in the backward dynamic programming procedure. We discuss this method in section 3.2.2.

Kharroubi et al. [46] treat the control as a state variable and approximate the

continuation value in dynamic programming procedure as a function of three variables: the state of the exogenous process, the inventory and the control. This allows them to optimise over an infinite (or even continuous) set of controls at the cost of adding additional dimensions to the basis functions used in the regression. Langrené et al. [49] shows an application of this methodology to natural resource extraction. In fact, Kharroubi et al. [46] solve a more general problem of controlling diffusions, so that the control affects the random dynamics itself. We discuss this method in section 3.2.3.

The contribution of this chapter is twofold: firstly, we introduce alternative Regression Monte Carlo algorithms presented in literature and describe their characteristics compared to the RLMC algorithm. Secondly, we compare the implementation of the RLMC algorithm in the inventory case, with the general case discussed in chapter 2.

3.2 Regression Monte Carlo numerical algorithms

In this section, we present different algorithms that have been proposed in the literature to solve inventory problems using Regression Monte Carlo based-algorithms. In particular, we will discuss Grid Discretisation, Quasi-simulation of the inventory and Control Randomisation.

In the case of inventory problems, compared to the most general case, it might be easier to visualise why the plain Regress Now approach cannot be applied. Notice that the calculation of an optimal control at time n is, in fact, a fixed point problem: one has to find a control u_n^* such that after doing a regression approximation of the conditional expectation with I_{n+1} determined by this control, the maximiser in

(2.3.9) equals to u_n^*.

The absence of stochasticity in the dynamics of the inventory, however, allow us to circumvent the difficulties presented in section 2.1.3 and solve the control problem by modifying the original algorithm for Regress Now projections.

3.2.1 Grid discretisation of inventory levels

This approach effectively consists of discretising the inventory domain and computing separate regression at each level, taking the value of the inventory as given and thereby circumventing the difficulties presented before.

Consider a discretisation $\{\lambda_1 = 0, \ldots, \lambda_L = I_{\max}\}$ of the domain of the controlled inventory and assume that we use an uniform grid where $\lambda_i = \{(i-1)\frac{I_{\max}}{L-1} : \ i = 1, \ldots, L\}$. To generate training points for the exogenous process $(X_s)_{s=1}^N$, on the other hand, simulate M trajectories $(X_s^m)_{m,s=1}^{M,N}$ using the transition density and a given initial point. In order to compute the continuation value in (3.1.2), regress the set of responses $\{V(n+1, X_{n+1}^m, \lambda_l)\}_{m=1}^M$ over $\{\phi_k(X_n^m)\}_{m,k=1}^{M,K}$ for each $l = 1, \ldots, L$ obtaining a collection of coefficients $\{\alpha_n^{k,l}\}_{k,l=1}^{K,L}$ and approximations $\sum_{k=1}^K \alpha_n^{k,l}\phi_k(x)$ of $\mathbb{E}[V(n+1, X_{n+1}, \lambda_l)|X_n = x]$.

Since these approximations are for a fixed set of inventory levels, we need to extend the approximation of the continuation value over the full inventory domain. This is usually done in literature [12] by interpolating values $\{\sum_{k=1}^K \alpha_n^{k,l}\phi_k(x)\}_{l=1}^L$ between grid points $\{\lambda_l\}_{l=1}^L$. The interpolation procedure requires a gridded, uniform design and, therefore, the computational complexity of this task increases exponentially with the number of grid points and dimensions of the inventory. We define a linear interpolation function by

$$\mathcal{I}(x, i, u; \{\alpha_n^{k,l}\}_{k,l=1}^{K,L}, \{\lambda_l\}_{l=1}^L) = c\sum_{k=1}^K \alpha_n^{k,\ell}\phi_k(x) + (1-c)\sum_{k=1}^K \alpha_n^{k,\ell+1}\phi_k(x),$$

where $\ell \in \{1, \ldots, L\} : \varphi_I(n, x, i, u) \in (\lambda_\ell, \lambda_{\ell+1}]$ and $c = \frac{\varphi_I(n,x,i,u)-\lambda_\ell}{\lambda_{\ell+1}-\lambda_\ell}$.

To obtain samples of the value function to use in the regression procedure at the next time step, we solve an approximate backward dynamic programming equation at the states (x, λ_l), $\forall\, x,\ l = 1, \ldots, L$:

$$V(n, X_n^m, \lambda_l) = \sup_{u \in \mathcal{U}_n} \Big\{ f(n, X_n^m, \lambda_l, u) + \mathcal{I}\Big(X_n^m, \lambda_l, u; \{\alpha_n^{k,l}\}_{k,l=1}^{K,L}, \{\lambda_l\}_{l=1}^{L}\Big) \Big\}$$

for $l = 1, \ldots, L$.

Algorithm 3 provides details of the implementation. Extension to performance iteration results in a minor amendment as can be seen in the algorithm.

The inventory discretisation method is wide-spread because of the relative simplicity of extending a traditional Regression Monte Carlo algorithm for optimal stopping to controlled dynamics, see, e.g., Boogert and de Jong [12], Denault et al. [23], Kitapbayev et al. [47], Nadarajah et al. [60]. However, due to separate regressions needed at each discretisation point of the endogenous (controlled) component, it could be prohibitive to implement this algorithm as the dimensionality of the controlled process grows. See the second part of this Thesis, section 6.3 in particular, to appreciate the degradation of performance already observed in a 2-dimensional problem.

Remark 3.2.1. *An alternative to the use of interpolation, to extend the approximation of the continuation value over the full domain of the inventory, is regression. Despite the lack of examples in literature, regression offers a way to smooth out the noise in the approximations at different inventory levels and can be calculated much more efficiently than interpolation when the dimension of the inventory grows, because it is not constrained to uniformly gridded designs. This approach practically means generating, through multiple regressions for fixed inventory levels, training points for the continuation value which are then fit by regression including both exogenous and inventory dimension in the basis function.*

Algorithm 3 Regression Monte Carlo algorithm with Grid Discretisation (GD)

1: Generate training points $\{X_n^m\}_{n=1,m=1}^{N,M}$

2: Initialise the value function $V(N, X_N^m, \lambda_l) = g(X_N^m, \lambda_l), \quad j = 1, \dots, L, \ m = 1, \dots, M$

3: **for** $n = N-1$ to 1 **do**

4: **for** $l = 1$ to L **do**

5: $\alpha_n^l = \underset{\alpha \in \mathbb{R}^K}{\operatorname{argmin}} \{\sum_{m=1}^M [V(n+1, X_{n+1}^m, \lambda_l) - \sum_{k=1}^K \alpha_k \phi_k(X_n^m)]^2\}$

6: **if** Value Iteration **then**

$$V(n, X_n^m, \lambda_l) = \max_{u \in \mathcal{U}_n} \{f(n, X_n^m, \lambda_l, u) + \mathcal{I}(X_n^m, \lambda_l, u; \{\alpha_n^{k,l}\}_{k,l=1}^{K,L}, \{\lambda_l\}_{l=1}^L)\}, \quad \forall m, l$$

7: **if** performance iteration **then**

8: For each m and l, compute inventory trajectories $\{I_s^{m,l}\}_{j=n}^N$ along $(X_s^m)_{s=n}^N$ starting from $I_n^{m,l} = \lambda_l$ using control

9: $u_n^*(x, i) = \underset{u \in \mathcal{U}_n}{\operatorname{argmax}} \{f(n, z, y, u) + \mathcal{I}(X_n^m, i, u; \{\alpha_n^{k,l}\}_{k,l=1}^{K,L}, \{\lambda_l\}_{l=1}^L\}.$

10: Set

11: $V(n, X_n^m, \lambda_l) = \sum_{s=n}^{N-1} f\left(j, X_j^m, I_j^{m,l}, u_n^*\left(X_j^m, I_j^{m,l}\right)\right) + g(X_N^m, I_N^{m,l})$

12: **Evaluate the policy**

Further analysis is necessary to comment on the overall efficiency of this approach over interpolation, however, this lies outside the scope of the thesis.

Remark 3.2.2. *There are cases when the control can only take values in a discrete set, that the inventory process is confined to a low-cardinality set of known fix values dependent on the initial condition I_0. In such cases, separate regressions can be fitted at each level without requiring any form of extensions over the inventory domain.*

3.2.2 Quasi-simulation of the inventory

In order to include the inventory in the regression, its trajectories (I_n) have to be simulated before the control is computed. However, as the control directly guides the evolution of (I_n), the inventory should rather be stated at time N and then its paths constructed backward in time as the dynamic programming procedure progresses. The question therefore becomes, does it exist a predecessor i for the inventory level $I_{n+1} = j$ such that if I am in state (n, x, i) following my control policy, I would land onto $(n+1, X_{n+1}, j)$? Carmona and Ludkovski [17] and Denault et al. [23] resolved this issue in similar ways in the framework of managing an energy storage device for which only 3 controls are available: charge, discharge and store; i.e. $u \in \{-1, 0, 1\}$.

In Carmona and Ludkovski [17] the authors deal for the first time with the task of regressing $\{V(n+1, X^m_{n+1}, I^m_{n+1})\}_{m=1}^M$ over basis functions dependent on both exogenous and inventory processes. Clearly, only three predecessors I^m_n are possible and the authors suggest to test separately whether I^m_{n+1} is an optimal successor of one of these three. This is achieved by regressing the value function at time $n+1$ over observations $(X^m_n, I^m_{n+1})_{m=1}^M$ and later testing whether an optimal control at time n, for any of the three predecessors of I^m_{n+1}, drives the inventory process to I^m_{n+1}. If there is such a predecessor in the interval $[0, I_{\max}]$, the trajectory $(I^m_s)_{s=n+1}^N$ is extended to time n, otherwise, I^m_n is randomly placed in $[0, I_{\max}]$.

The approach in Denault et al. [23] uses basis functions dependent on X_n and I_n, and requires the computation of 3 separate regressions for each choice of control at step n. This procedure does not guarantee that the trajectories constructed backward for paths of the inventory process $(I_n^m)_{n,m}^{N,M}$ will remain inside $[0, I_{\max}]$ and they decided, therefore, to introduce an artificial penalty term and a random replacement of the point I_n^m when it is too far from the interval $[0, I_{\max}]$. The small and finite size of the set of possible controls is paramount for this method, notice how this approach closely reflect what is done in Grid Discretisation even though, in this case, multiple regressions are used to estimate the effect of the controls at all inventory levels, rather than the value of different inventory levels for all controls. Reliable and thorough tests are not available, however, our experience and the literature seem to suggest that the grid discretisation method is preferred over quasi-simulation whenever the dimensionality of the problem is small enough to allow its use.

3.2.3 Control Randomisation

Control Randomisation was introduced in Kharroubi et al. [46]; it differs from the previous methods in that the control becomes a state variable and it is simulated (according to some arbitrary model that ensures admissibility) along trajectories of $(X_s, I_s)_{s=1}^N$. Data underlying the regression procedure consists of three sets of trajectories $\{X_n^m, I_n^m, \tilde{u}_n^m\}_{n,m=1}^{N,M}$. In the case of Value Iteration (2.3.9), $\{V(n+1, X_{n+1}^m, I_{n+1}^m)\}_{m=1}^M$ is regressed against basis functions evaluated at the points $\{X_n^m, I_n^m, \tilde{u}_n^m\}_{m=1}^M$. These regression basis functions are dependent now on the random control $\tilde{u}_n$, in addition to X_n and I_n, so that the estimated continuation value will depend on the choice of the control (which is different on each sample trajectory). An optimal control at time n given $X_n = X_n^m$ and $I_n = I_n^m$ is

approximated by the expression (if the maximum is attained)

$$u_n^*(x,i) = \underset{u\in\mathcal{U}_n}{\operatorname{argmax}}\Big\{ f(n,x,i,u) + \sum_{k=1}^{K} \alpha_n^k \phi_k(x,i,u)\Big\}. \tag{3.2.3}$$

In general, multiple runs of the method could be needed to obtain precise estimates because the initial choice of the dummy control could drive the training points far from where the optimal control would have directed them. For details about the implementation and extension to the performance iteration case refer to Algorithm 4.

Remark 3.2.3. *Notice that Control Randomisation has originally been developed to solve general control of Markov process problems; we decided, however, to discuss this method only in this section to maintain the focus of chapter 2 on the convergence theorems for Regress Later. For further reference see Langrené et al. [49] where the authors apply the numerical method in the context of natural resource extraction.*

3.3 Regress Later Monte Carlo: a decoupling approach

Recall from section 2.1.4 that Regress Later approximates conditional expectations with respect to (X_n, I_n) in two stages. First, a conditional expectation with respect to (X_{n+1}, I_{n+1}) is approximated in a regression step by a linear combination of basis functions of (X_{n+1}, I_{n+1}). Then, analytical formulas (or precomputed numerical approximations) are applied to condition this linear combination of functions of future values on present values (X_n, I_n). Using the notation introduced in chapter 2 we have

$$\mathbb{E}_{n,x,i,u}[V(n+1, X_{n+1}, I_{n+1})] \approx \mathbb{E}_{n,x}[\hat{\Pi}_K \hat{V}\Big(n+1, X_{n+1}, \varphi(n,x,i,u)\Big)],$$

Algorithm 4 Regression Monte Carlo algorithm: Control Randomisation (CR)

Generate initial points $\{X_0^m, I_0^m\}_{m=1}^M$ from an initial distribution of interest.

Simulate the dynamics of the process (X, I) to produce the training points $\{X_n^m, I_n^m\}_{n=1,m=1}^{N,M}$ while generating a random control $\{\tilde{u}_n^m \in \mathcal{U}_{\{n:n\}}\}_{n=1,m=1}^{N,M}$.

Initialise the value function $V(N, X_N^m, I_N^m) = g(X_N^m, I_N^m), \quad \forall m$

for n=N-1 to 1 **do**

$\alpha_{n+1} = \underset{\alpha \in \mathbb{R}^K}{\text{argmin}}\{\sum_{m=1}^M [V(n+1, X_{n+1}^m, I_{n+1}^m) - \sum_{k=1}^K \alpha_{n+1}^k \phi_k(X_n^m, I_n^m, \tilde{u}_{n,m})]^2\}$

if Value Iteration **then**

For all m do

$$V(n, X_n^m, I_n^m) = \sup_{u \in \mathcal{U}_n} \left\{ f(n, X_n^m, I_n^m, u) + \sum_{k=1}^K \alpha_{n+1}^k \phi_k(X_n^m, I_n^m, u) \right\}$$

if Performance Iteration **then**

For each m, update trajectories $\{I_j^m\}_{j=n+1}^N$ using control given by equation (3.2.3)

$V(n, Z_n^m, Y_n^m) = \sum_{j=n}^{N-1} f\left(j, X_j^m, I_j^m, u_n^*\left(X_j^m, I_j^m\right)\right) + g(X_N^m, I_N^m)$

Evaluate the policy

where $\hat{V}(n+1,\cdot)$ is computed recursively backward in time from the terminal condition $\hat{V}(N,x,i) = g(x,i)$.

We provide now a comparison of the Regress Later algorithm with both the methods presented above, Grid Discretisation and Control Randomisation in particular, and the same Regress Later algorithm as implemented in the general case.

3.3.1 Comparison with the control of stochastic dynamics

The implementation of Regress Later for inventory problems features very little difference compared to the general case. Contrary to the case of control of stochastic processes, however, when a degenerate process I (deterministic transition function) is part of the state space, the multidimensional process (X, I) does not satisfy Assumption 1 in chapter 2 thereby making the theorems presented therein not applicable to this class of problems. In particular, notice that the constant $\bar{R}$, which provides a bound on the transition density function, would be infinity. Workarounds seem to be possible, but so far convergence of Regression Monte Carlo algorithms for this class of problems has not been proved.

It is worth mentioning an interesting modification of the Regress Later algorithm that combines the simplicity of Regress Now and the decoupling property of Regress Later by using the former for the exogenous dimension and the latter for the endogenous one. In particular we project over basis function $\{\phi_k(X_n, I_{n+1})\}_{k=1}^K$ and compute conditional expectations, as explained in section 2.1.4, by defining $\hat{\phi}_k(n,x,i) = \phi(x, \varphi_I(n,x,i,u))$. The advantage of this approach is that we do not require to compute the conditional expectation of basis functions, at the cost of having to use training points obtained by sampling simulated trajectories of X. This solution strategy has recently been explored in Ludkovski and Maheshwari [52] and inspired by Carmona and Ludkovski [17].

Remark 3.3.1. *Notice that, regardless of the lack of theoretical results for this class of problems, practical implementations benefit from the structure of the problem which offers clearer guidance on how to chose the training measures μ, compared to the general case. Notice in fact that, as the process X is uncontrolled, we know in advance which part of the state space is more relevant and therefore cluster more point in that area. On the other hand, we do not know where the optimally controlled inventory process I will be driven, conditional on a trajectory of X, and we need to extensively explore the state space $[0, I_{\max}]$ in order to produce a sensible policy that is aware of the hard constraint represented by a state of full or empty inventory. A more detailed discussion can be found in section 5.2.*

3.3.2 Comparison with other methods

Similarly to the other methods presented in this chapter, and as explained in the previous subsection, Regress Later applied to inventory problems lacks a proof of convergence. In this section, however, we explain why we consider it still preferable to Grid Discretisation and Control Randomisation, the two most established techniques in the literature.

As presented in section 3.2.1 the main drawback of Grid Discretisation is that the computational complexity scales badly with the number of dimensions of the controlled process I. Regress Later, on the other hand, only needs one regression to approximate the continuation value over the whole domain. An additional benefit of capturing the full structure of the continuation value in one regression is that the noise in the point-wise estimations produced by Grid Discretisation is automatically smoothed by the regression, improving the consistency of neighbouring estimates.

To a certain extent, the same argument applies also to the comparison with Control Randomisation 3.2.3, where the increasing complexity arises from the need to include

explicitly (as opposed to implicitly, as it is done in Regress Later) the dependence on the control process in the basis functions. For reference, consider a one-dimensional exogenous process X, a d dimensional inventory process I and a q dimensional control process u along with basis functions defined as linear interactions among all subsets of possible combinations of each dimension. In the case of Control Randomisation this accounts for approximately $\mathcal{O}(2^{d+q})$ basis functions as opposed to $\mathcal{O}(2^d)$ for Regress Later; or, in other words, Control Randomisation needs to estimate 2^q times the number of coefficients Regress Later does. In addition, even though the task of choosing a training measure μ in Regress Later is similar to that of choosing trajectories for the randomised control in Control Randomisation, the second problem appears to be more difficult and only partial guideline is offered in the literature.

Chapter 4

RMC for optimal stopping and switching

In this chapter we discuss how to apply the regression approximation introduced in chapter 2 to the Bellman equations (1.3.15)-(1.3.16) in order to solve optimal stopping and switching problems. The application of regression techniques to estimate conditional expectation in more general problems has been discussed in chapter 2 and 3.

Notice that in this chapter we will cover exclusively known results and provide a summary of the early literature on Regression Monte Carlo algorithms. In particular, in section 4.1, we discuss the reasons why the stopping and switching framework is considerably easier than the general case, while in section 4.2 we present a discussion about the results on convergence behaviour available in literature and how they compare to the results in our theorems in chapter 2 .

4.1 A simpler framework

Traditional Regress Now approximations, as introduced by Tsitsiklis and Van Roy [76] and Longstaff and Schwartz [51], can be applied to problems of stopping and switching because of the convenient structure of the controlled second dimension I which takes values in a set with low cardinality. This structure is exploited by performing a number of regression approximations for each fixed value of I, so that the effect of the control on the conditional expectations can be modelled directly by different sets of parameters. Consider for example the case of optimal stopping: I takes values in a set with cardinality 2, i.e. $\{0, 1\}$, and we know that $V(n, x, 1) = c_{0,1}(x)$ for all time steps and states $(x, 1)$, thereby requiring one regression only, for the case $I = 0$. In the optimal switching framework, on the other hand, we consider a small number of regimes R (the cardinality of I), therefore requiring R different regression approximations.

4.1.1 Value Iteration

As usual, we initiate the backward estimation procedure for Value Iteration by assigning the known terminal condition to the value function $V(N, \cdot, i)$, $i = 1, \ldots, R$. Assume that we simulated a collection of paths from the dynamics of X, thanks to which we know that all of our samples follow the appropriate distribution, i.e. $X^m_{n+1} \sim (X_{n+1}|X_n = X^m_n)\ \forall n$, allowing us to compute $\mathbb{E}[V(N, X_{n+1}, i)|X_{N-1} = x] \approx \sum_{k=1}^{K} \alpha^n_{j,k}\phi_k(x)$, $i = 1, \ldots, R$. The backward procedure continues by iterating the optimisation and regression step on the simulated paths until the initial time is reached and the full collection of regression coefficients $\{\alpha^s_{j,k}\}^{R,K,N}_{j,k,s=1}$ is available:

$$V(n, x, i) = \max_{u \in \mathcal{U}_{\{n:n\}}} \left\{ f(n, x, u) + \sum_{k=1}^{K} \alpha^n_{u,k}\phi_k(x) - c_{i,u} \right\},$$

where $\boldsymbol{\alpha}^n_j = \mathcal{A}^{-1} \frac{1}{M} \sum_{m=1}^{M} V(n + 1, X^m_{n+1}, j)\boldsymbol{\phi}(X^m_n)$.

In a "Later" framework, even though we could include the index dimension in the basis functions, it is common to follow the same approach used for Regress Now and repeat the regression procedure for all regimes. This choice is sometimes necessary as the regimes might be categorical rather than numerical and their ordering subject to the modeller choice. Beutner et al. [11], among others, shows that Regress Later approximations are still preferable to Regress Now ones for their lower variance.

4.1.2 Performance Iteration

Moving into the Performance Iteration framework, one striking difference between the problems discussed so far and optimal stopping and switching case arises. Recall that the implementation of Performance Iteration discussed so far required the computation of

$$J(n, (X_s, I_s, u_s)_{s=n,\dots,N}) = \sum_{s=n}^{N-1} f(s, X_s, u_s(X_s)) + g(X_N),$$

after u_n has been decided, which numerically means resimulating the path from time n to the terminal time, for each training point, incurring a considerable computational cost. In case of optimal stopping and switching, on the other hand, the update rule for J can be written in a much more easily computable form; it requires, however, a pathwise interpretation of the simulations, which would force Regress Later to adopt a time-dependent training measure $\mu \sim (X_{n+1}|X_n)$. The reason why this formulation is available, is that the dynamics of the first dimension X, which contains all the randomness in the system, is not affected by the choice of the control and therefore, it is guaranteed that the running cost will be equal to the sum of the current contribution and the cost accumulated up to the previous backward step. Or, in other words, at time $n+1$ we will land on a point for which the performance along a path have already been computed. The following update rules avoid the path resimulation making the Performance Iteration update as fast

as the Value iteration one.

$$\text{stopping: } J(n,(X_s,I_s,u_s^*)_{s=n,\dots,N}) = \begin{cases} f(n,X_n^m,1)+c_{0,1} & \text{if } c_{0,1} > \sum_{k=1}^K \alpha_k^n \hat{\phi}_k(n,X_n^m) \\ J(n+1,(X_s,I_s,u_s^*)_{s=n+1,\dots,N}) + h(X_n) & \text{otherwise} \end{cases} \tag{4.1.1}$$

$$\text{switching: } J(n,(X_s,I_s,u_s^*)_{s=n,\dots,N}) = \begin{cases} J(n+1,(X_s,I_s,u_s^*)_{s=n+1,\dots,N}) + f(n,X_n^m,u^*) - c_{i,u^*} & \text{if } u^* = j \\ J(n+1,(X_s,I_s,u_s^*)_{s=n+1,\dots,N}) + f(n,X_n^m,u^*) & \text{otherwise} \end{cases}$$
$$\text{where } u^* = \underset{j\neq i}{\operatorname{argmax}}\{f(n,X_h^m,j) - c_{i,j} + \sum_{k=1}^K \alpha_k^n \hat{\phi}_k(n,X_n^m,j)\} \tag{4.1.2}$$

Remark 4.1.1. *Notice that, even though one might be tempted to apply a similar approach when using Grid Discretisation to solve Inventory problems (section 3.2.1), this is not possible. In the inventory case, often the control leads the inventory at a level at which an approximation, and therefore a path, is not available. The use of interpolation in this case is ineffective since the performance J depends on one entire simulated trajectory. As discussed in remark 3.2.2, however, there might be cases when the control is discrete and the direct update rule for Performance Iteration can be effectively applied due to the finite number of states the inventory process can be in.*

4.2 Convergence results

In this section, we present a literature review of the convergence behaviour for optimal stopping from the origins of Regression Monte Carlo until today.

4.2.1 Early results

The first study of convergence behaviour after Tsitsiklis and Van Roy [76] and Longstaff and Schwartz [51], was Clement et al. [21], who estimated the convergence rate of the algorithm introduced in the latter. It is also worth noticing the error bound estimated by Glasserman and Yu [32] shortly after, who states in Theorem 3, that the leading term behaves like $\frac{1}{\sqrt{M}}$. This bound has been obtained with strong assumptions on the fourth-order moments of the basis and reward function. Glasserman and Yu [32] also shows that the number of basis functions K should grow as $\ln M$ to guarantee convergence.

Under weaker conditions than Glasserman and Yu [32], and assuming convexity of the basis function approximation, Egloff [27] obtains a bound on the convergence error of the order of $\sqrt{\frac{\ln M}{M}}$. The convergence analysis in Egloff [27], that makes large use of the concept of Vapnik-Chervonekis (VC) dimensions, is wider in scope than most discussed here, as he investigates a dynamic look-ahead algorithm, which includes Tsitsiklis and Van Roy [76] and Longstaff and Schwartz [51] as special cases.

Zanger [79] further improves previous results by proving a bound similar to Egloff [27], replacing the assumption on the convexity of basis functions with uniform boundedness and finite VC dimension, for arbitrary sets of functions. Zanger [80] then shows that the same rate of convergence holds, even when the uniform boundedness assumption is dropped.

These results should be compared with similar estimates for Regress Later types

of iterations, however such results are not available in general. As pointed out by Beutner et al. [11], however, we should find Regress Later estimates to exhibit a faster rate of convergence than Regress Now.

4.2.2 Pseudo regression

We turn now our attention to more recent results featuring a structure much closer to our theorems in chapter 2. In Bayer et al. [8], the authors present a Regress Now algorithm called *pseudo-regression* that, similarly to our approach, pre-computes the matrix $\mathcal{A}$ and places training points at time n, when estimating $\mathbb{E}\Big[V(n+1, X_{n+1})|X_n = x\Big]$, according to a training distribution μ. For the sake of consistency, we will now present the theorems 4.1, 4.2 and 4.5 therein, maintaining the notation used in this thesis. Notice that, similarly to our theorems, and in contrast with the results presented in section 4.2.1, the approximation error here is left unspecified, and indicated by ϵ_K.

In theorem 4.1, Bayer et al. [8] studies the convergence of the estimation of the value function when the matrix $\mathcal{A}$ has been precomputed and an analytical bound for its norm is available.

Theorem 4.2.1 (Bayer et al. [8] 4.1: Accuracy when $\mathcal{A}$ is precomputed, V.I.)**.** *Suppose that assumption 2.3.1 is satisfied and, in addition, assume that* $\mathbb{V}\Big(\hat{V}(n+1, X_{n+1})\Big|X_n = x\Big) < Var$ *for some constant* $Var > 0$ *and* $\forall\, x \in \mathcal{D}$*, where we indicate by* $\mathbb{V}$ *the variance of the random variable* $\hat{V}$*. Further assume that the smallest and largest eigenvalues of the matrix* $\mathcal{A}$ *are bounded by* θ_K *and* $\theta_K^{\max}$*. Then it holds:*

$$\left\|\hat{V}(n, \tilde{X}) - V(n, \tilde{X})\right\|_{L^2_{e,N-n}} \leq \sqrt{\frac{\theta_K^{\max}}{\theta_K}(Var + \Gamma^2)\frac{K}{M} + \epsilon_K}$$

A direct comparison with our theorem 2.3.3 is impossible, given the restricted scope of Theorem 4.2.1 and the Regress Now approach they use. It is interesting to note,

however, the identical behaviour in $\sqrt{\frac{K}{M}}$, with a multiplicative constant dependent, in both cases, on Γ and $\|\mathcal{A}_K^{-1}\|_2$.

The second theorem in Bayer et al. [8] analyses the same convergence, when, on the other hand, the regression coefficients in the estimated conditional expectations are approximated by solving the full linear regression system, without assuming the knowledge of $\mathcal{A}$.

Theorem 4.2.2 (Bayer et al. [8] 4.2: Accuracy when $\mathcal{A}$ is not precomputed, V.I.)**.** *Suppose that assumption 2.3.1 is satisfied and, in addition, assume that* $\mathbb{V}\Big(\hat{V}(n+1, X_{n+1})\Big|X_n = x\Big) < Var$ *for some constant* $Var > 0$ *and* $\forall\, x \in \mathcal{D}$*. Then it holds:*

$$\left\|\hat{V}(n,\tilde{X}) - V(n,\tilde{X})\right\|_{L^2_{e,N-n}} \leq \sqrt{c\max\{Var, \Gamma^2\}\frac{(1+\ln M)K}{M} + 8\epsilon_K}$$

The main difference with 4.2.1, which provides an evidence that placing i.i.d. training points according to a known measure μ has desirables properties, resides in the slower convergence $\frac{(1+\ln M)K}{M}$ observed in Theorem 4.2.2.

To conclude, we present the theorem stating the convergence of the performance iteration procedure.

Theorem 4.2.3 (Bayer et al. [8] 4.5: Accuracy when $\mathcal{A}$ is precomputed, P.I.)**.** *Suppose that assumption 2.3.1 is satisfied and, in addition assume that* $\mathbb{V}\Big(\hat{V}(n+1, X_{n+1})\Big|X_n = x\Big) < Var$ *for some constant* $Var > 0$ *and* $\forall\, x \in \mathcal{D}$*. Then it holds:*

$$\left\|\hat{V}(n,\tilde{X}) - V(n,\tilde{X})\right\|_{L^2_{e,N-n}} \leq c\Big(1+\sqrt{\mathcal{R}}(c+1)\Big)^{N-n-1}\Big(\sqrt{\frac{K}{M}} + \epsilon_K\Big),$$

where $\mathcal{R} = \max_{0\leq n\leq i\leq N}\sup_{x\in\mathcal{D}}\frac{\mu_{n,i}(x)}{\mu_n(x)} < \infty$ *and* $\mu_{n,i}$ *is the distribution of* X_i *simulated from* X_n*.*

Chapter 5

Tuning and considerations

In previous chapters, we discussed extensively how to apply the Regression Monte Carlo algorithm to different problems of optimal stochastic control, but we have given little insight into practical issues that one might encounter when dealing with the implementation.

In this chapter we will first discuss the practical difference between Value and Performance Iteration, i.e. when it is more convenient to use one rather than the other, then move onto the role of the training measure μ and of the basis functions, explaining how to choose both of them in order to improve the quality of the estimated policy. We conclude the chapter with two sections that discuss gradient descent optimisation and backward construction of paths for inventory problems.

This chapter is, *per-se*, one of the contributions of this thesis; it represents three years of numerical experiments and fine tuning of the Regression Monte Carlo algorithms for solving specific problems. In the second part of the thesis, we will put into context some of these techniques within some full-scale optimisation problem.

5.1 Value vs. Performance Iteration

In this section, we present a critical analysis of the two different iterative approaches presented in section 2.3 and 2.5; first, we comment on the theoretical results and then on our experience from practical implementation.

There have not been numerous studies in the literature about the characteristics of the Performance Iteration approach outside the framework of option pricing. In the context of controlled Markov processes, the difference between the two approaches is paramount, as it leads to different computational complexity and accuracy for the two algorithms. The difference in performance is caused by the evaluation step employed by Performance Iteration.

The main reason to decide to use Performance Iteration over Value Iteration is that the latter induces propagation of the projection error, while the former uses an update which does not depend directly on the functional form of the estimator of the conditional expectation and, therefore, does not propagate the error. At a first glance, however, observing the error bound for Value and Performance Iteration algorithms (provided in Theorems 2.3.3 and 2.5.4) one might be tempted to claim that the former has, at least in general situations, a tighter error bound than the latter. Recall, however, that the quantities estimated by the two algorithms are somewhat different. Value iteration provides an estimation of the value function (error assessed in Theorem 2.3.3) and of the control policy (whose value is assessed by the Forward Evaluation procedure). Performance Iteration, on the other hand, provides an estimation of the control policy and an estimation of the value of the latter. When comparing the quality of the estimated policies (which are of interest in most practical applications), we have to turn our attention to Theorem 2.4.1, which indeed provides us with error bounds for the control policy estimated by the Value Iteration algorithm. Straightforward computations show that the error bound for

the under-performance of the policy computed by Value Iteration, compared with the optimal control, might be up to N times higher than the bound on the error made when estimating the value function directly. Further, denoting by η_V and η_P the error bounds for the performance of the control policy estimated by Value and Performance Iteration, as presented in Theorems 2.4.1 and 2.5.4, we have:

$$\frac{\eta_V}{\eta_P} = \frac{3\bar{R}-1}{(\bar{R}-1)^2}\frac{1}{(3\bar{R})^N-1}\Big((N-n)\bar{R}^{N+1}-(N+1)\bar{R}^N+1\Big) \approx \mathcal{O}(N3^{-N}). \quad (5.1.1)$$

Equation (5.1.1) shows that, in general, the policy estimated by the Value Iteration procedure is better than the policy estimated by Performance Iteration, for $N, \bar{R} \gg 1$.

It should be noted, however, that in section 2.5.3, proposition 1, we introduced a sharper bound for Performance Iteration, under a certain assumption on the projection error for the estimated value function, $\tilde{\epsilon}_K = \max_{n=1,\dots,N} \left\|\Pi_K \tilde{V}(n,\tilde{X}) - \tilde{V}(n,\tilde{X})\right\|_{L^2_\mu}$. Under such assumption, namely that $\tilde{\epsilon}_K \leq \epsilon_K$, we can update the comparison in (5.1.1) with

$$\frac{\eta_V}{\eta_P} = \frac{(N)\bar{R}^{N+1}-(N+1)\bar{R}^N+1}{3(\bar{R}^N-1)\bar{R}}(\bar{R}-1) \approx N, \quad (5.1.2)$$

which shows that, in certain situations, the error bound for Performance Iteration is tighter than the one for Value Iteration estimates. The difference in performance between (5.1.1) and (5.1.2) is reflected by the empirical observation that when Performance Iteration produces low-quality estimates, these are usually much worse than those computed by Value Iteration. In other cases, however, Performance Iteration can produce estimations that are more precise than those produced by Value Iteration.

Our theoretical framework, therefore, suggests using Performance Iteration over Value Iteration when $\tilde{\epsilon}_K \leq \epsilon_K$. Our practical experience suggests that this assumption corresponds with a heuristic condition like the one we propose here:

The basis functions can induce a sensible policy. For example, when the conditional expectation plus the running reward is a convex function of the control, and we use second order polynomial basis functions, the vertex of the parabola can give direction, while the amplitude gives the urgency of moving in such direction.

It should be noted however that, in practice, the choice between Value and Performance Iteration, depends on the interplay between the control problem we want to solve and the error propagation and stability properties of the two methods. Often, when a good set of basis functions and training measure cannot be found for Value Iteration, Performance Iteration provides an alternative which, at the cost of higher computational complexity, can provide satisfactory results.

5.1.1 Error propagation

In practice, when the following two conditions are satisfied, the effect of error propagation is considerable:

- the value function cannot be easily represented using the basis functions,
- the value function changes shape substantially from one time step to the next.

When the effect of error propagation is assessed to be considerable, and under the assumption that $\tilde{\epsilon}_K \leq \epsilon_K$, we can observe a substantial improvement in the quality of the estimated control policy when using Performance over Value Iteration. The greater precision is due to the ability of Performance Iteration to stop the error propagation by using separate training points for learning and testing, thereby improving the quality of the estimated regression coefficients. Value Iteration, on the other hand, reuses the estimated conditional expectation to compute the current value, propagating the approximation error at each time step. These small contributions can build up to a considerable error.

5.1.2 Stability

Other than the error propagation it is important to assess the stability of the estimations, both over time and over different runs of the algorithm. Contrary to Value Iteration, where we project a function of a random variable ξ_n, in Performance Iteration we project a function of $\xi_n, \ldots, \xi_{N-1}$. The latter requires more training points for stable parameters estimation, therefore causing high inter-temporal variance. The regression parameters estimated by Value Iteration, on the other hand, have a smaller variance from one time-step to the next, due to the nature of the method.

Consider also the situation where the running profit $f \approx o(g)$. The regression parameters for Value Iteration are hugely influenced by the first regression approximation at time N, and the greater sensitivity to the approximation of the terminal condition is paid in terms of greater inter-experimental variance (variance observed between different runs of the algorithm). Performance Iteration, on the other hand, estimates parameters at each time step with "fresh data", guaranteeing consistent estimates over different experiments.

5.2 Training measure and choice of basis functions

In this section, we analyse different choices of training measures and basis functions, indicating the pros and cons of different alternatives, and guide the reader to balance in the most efficient way the interplay between the two.

5.2.1 Choice of training measure μ

As shown by Theorems 2.3.3, 2.4.1 and 2.5.4 the choice of the training measure μ is paramount for a quick convergence of the estimations. The Theorems show that the distribution of the training points influences the quality of the estimations mainly through the bound on the transition density $\bar{R}$ and the representation error ϵ_K.

Remark 5.2.1. *In the following, we will present the consequences of choosing a particular measure μ in two common situations. Notice, however, that $\bar{R}$ and ϵ_K are defined in terms of the training measure μ or, in other words, the training measure is also used to assess the error, making it impossible to compare error bounds obtained under different measures. For the sake of using the language developed so far, in this section we will intend by $\bar{R}$ and ϵ_K the constants computed under a different, fixed, error measure, which places more relevance to areas of the state space that we deem more interesting.*

Uninformed choice

When no information about the problem is exploited, we use a uniform distribution on $\mathcal{D}$. This choice allows us to deal with any problem, regardless of our prior understanding, at the cost of a more difficult fitting of the basis functions (see figure 5.1). Choosing the uniform measure often results in a small value of $\bar{R}$, but produces a potentially big representation error ϵ_K, due to the lack of focus on the area of the state space which matters most. Notice that this choice of μ always satisfies Assumption 1.

Knowledge based

When previous knowledge about the problem is available, the training measure μ can be chosen in order to maximise the quality of the control policy. Even though

providing theoretical results in this direction is outside the scope of this thesis, notice that concentrated measures μ can produce very low values of ϵ_K (see figure 5.1) at the cost of high values of $\bar{R}$.

We present now a practical example for which we can show the trade off between exploration and accuracy that arises when choosing the training measure μ. Notice that in the following we will use the actual estimation error on this particular problem (available along with the optimal solution) as a proxy for the general error bounds discussed in chapter 2.

Example 5.2.2. *Consider the process* $X_1 = -5 \vee Z_1 \wedge 5$*, where* $Z_1 = Z_0 + u + \xi$ *,*$\xi \sim \mathcal{N}(0,1)$ *and the stochastic control problem*

$$V(0,z) = \inf_{u \in \mathcal{U}} \Big\{ \mathbb{E}\Big[\frac{u^2}{15} + g(X_1) | X_0 = z \Big] \Big\},$$

where $g(x) = \min(1, x^2)$*. We choose to test basis functions* $\{1,\, x,\, x^2\}$ *and a family of measures* $\mu_\sigma = \mathcal{N}(0, \sigma^2)$*. In this framework the trade off between* ϵ_k *and* $\bar{R}$ *is driven by the parameter* σ*, which determines the width of the training distribution. The effect of* μ *on the quality of the estimated conditional expectations can be assessed from Figure 5.1 which displays, on the right, the actual values of* ϵ_k *and* $\bar{R}$ *for some choices of* σ*. The effect on the control policy, however, is more subtle. Figure 5.1 displays on the left an example of the effect of* σ *on the estimated control.*

A better assessment of the effect of the measure μ*, through the parameter* σ*, is displayed in Figure 5.2. The graph shows the average (for different uniformly distributed initial points* $x \sim \mathcal{U}[-5,5]$*) performance of the estimated control policy as a function of the parameter training measure. It can be noticed that in this problem, it exists a point where the trade-off between exploration and exploitation is optimised.*

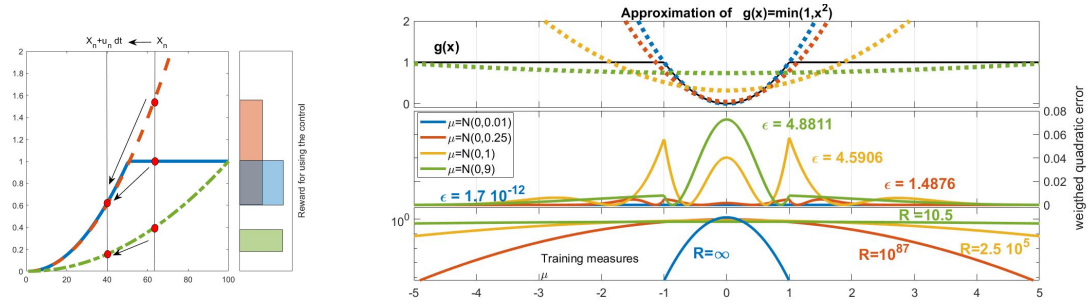


Figure 5.1: In the figure above we show an example of the impact of the measure μ on $\bar{R}$ and ϵ_K. On the right, in the top panel we show four different projections of $g(x) = \min(1, x^2)$ over $\{1, x, x^2\}$ produced by different measures μ. In the central panel we show the pointwise quadratic deviation of the projection from $g(x)$, in the lower panel the logplot of the density function of μ. Notice that to more concentrated measures correspond lower ϵ_K but higher $\bar{R}$, vice-versa higher ϵ_K and lower $\bar{R}$ are associated with wider measures. We display on the left the same trade-off acting on the estimation of the control policy. The graph displays the true (blue) and estimated (green, orange) expected reward from choosing a control u_n. More concentrated measures overestimate the effect of the control and tend to be myopic, achieving higher accuracy but paying a high cost for control. Measures that allow for exploration are more conservative, tend to underestimate the value of a given action, and therefore the control is used only when it is well worth it.

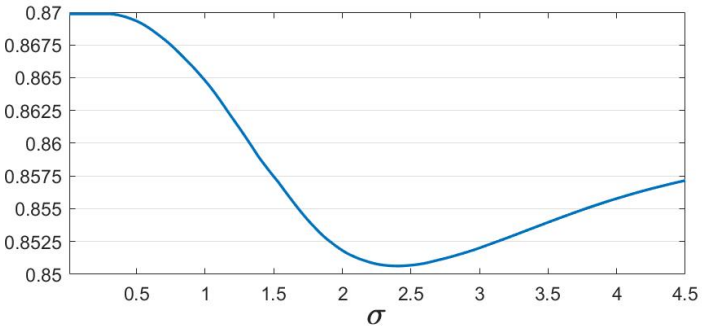


Figure 5.2: The graph shows the average performance of the estimated control policy as a function of the parameter σ, which defines the training measure $\mu \sim \mathcal{N}(0, \sigma^2)$

5.2.2 Choice of basis functions

In this section, we discuss the choice of basis function, which directly affects the precision of the estimates through ϵ_K.

Many authors have tested the performance of Regression Monte Carlo with respect to the main source of inefficiency in the method, namely, the choice of the basis functions. The general rule for choosing the latter is that they should somehow resemble the shape of the value function at each time period. A good choice of basis functions predominately depends on the intuition of the modeller, however, results presented, among others, in Moreno and Navas [58] suggest that the least squares Monte Carlo for a simple optimal stopping problem is robust to choice of basis function while, when the value function has complicated shapes, certain particular functions should be present in the set of basis functions, in order for the estimation to be consistent.

In the following paragraphs, we explore the two possible categories of basis functions: global and local. As the name suggests the former kind uses information from the whole state space to produce a general functional approximator "good enough" over the whole state space. The latter only focuses on local information to produce a collection of functional approximators for different areas of the state space. The most common families of basis functions are:

- Monomials
- Orthogonal polynomials
- Local affine functions
- Radial basis functions

One of the most popular choices of global basis functions when tackling a general stochastic control problem using Regression Monte Carlo algorithms is certainly the family of monomials up to order κ. Let us show in the following the consequences of such choice and examine few more popular alternatives, both global and local.

Monomials

Notice that an arbitrary number of monomials, i.e. $\phi_{k(j_1,\ldots,j_d)}(x) = \prod_{s=1}^{d} x_s^{j_s}$, $j_s \geq 0$ and $x = (x_1, \ldots, x_d)$, $k : (\mathbb{N}^+)^d \mapsto \{1, \ldots, K\}$, are never orthogonal on any domain and under any admissible training measure. In general, we can not provide more specific error bounds than those in Theorems 2.3.3 and 2.5.4. In order to shed more light on the consequences of the choice of these basis functions let us consider a typical situation encountered in practice: we assume a domain $\mathcal{D} = [0, 1]$ and as training measure $\mu = \lambda([0, 1])$ the Lebesgue measure on $\mathcal{D}$. In the described scenario the matrix $\mathcal{A}_K$ is just the Hilbert matrix $H_{i,j} = \frac{1}{i+j-2}$, $i, j = 1, \ldots, K$ while the norm of the basis functions is bounded by 1, i.e. $\max_{j=1,\ldots,K}\{\|\phi_j\|_{L^2_\mu}\} = \max_{j=1,\ldots,K}\{x^j\} = 1,\ x \in \mathcal{D}$. In order to obtain an explicit error bound we have to assess the norm of the inverse of the Hilbert matrix $\mathcal{A}_K$; we have $\|\mathcal{A}_K^{-1}\|_2 \leq 1/\theta_K$, where θ_K is a lower bound for the smallest eigenvalue of $\mathcal{A}_K$. Following Wolkowicz and Styan [78] and some calculations, we obtain the asymptotic behaviour of $\|\mathcal{A}_K^{-1}\|_2$ being $\mathcal{O}\Big(2^{4K^2-3K+1.5}(\pi K)^{4K^2-K}\Big) = \mathcal{O}\Big(K^{K^2}\Big)$. It is well known that the Hilbert matrix is very ill-conditioned and difficult to precisely invert numerically, to mitigate this problem it is advisable in practice to standardise the set of basis functions so that they have similar mean and variance. The final consideration about monomials is that in practical situations when the value function is truly a polynomial of small degree, we might have $\epsilon_K \approx 0$ without having to take K too big, which would otherwise cause $\theta_K \approx 0$.

Orthonormal basis of polynomials

A straightforward generalisation of the family of monomials described in the paragraphs above is given by the orthogonal polynomial basis of which many examples exist on both compact and unbounded domains. In the literature, orthogonal polynomial basis have been extensively studied in the option pricing framework and in relation to optimal stopping Regression Monte Carlo algorithms, see Moreno and Navas [58] among others. Through our Theorems 2.3.3 and 2.5.4 and Corollaries 2.3.5 and 2.5.5 we can assess to which extent it is preferable to chose orthonormal polynomial functions over monomials. Notice that, in general, for low order polynomials it is possible to compute the optimal control, in closed formula, in terms of the projection coefficients.

Remark 5.2.3. *The possibility of computing the optimal control analytically, as a function of the regression coefficients, often influences the choice of the basis functions. Avoiding the numerical search for the optimiser at each time step reduces the computational complexity considerably, and allows to run more precise simulations in the same amount of time. Notice that there are cases where, under a reasonable time constraint, Performance Iteration gives better results when running with quadratic basis functions and numerous training points than when more complex basis functions are added but less training points can be used due to the lengthy search for the optimal control.*

Local affine basis

It is sometimes useful, when little is known about the structure of the value function, to exploit the flexibility of local approximations. A very popular choice is to take affine functions (linear in one dimension) with disjoint supports, i.e. $\phi_k(x) = x^{k \bmod 2} \mathbb{1}_{\{x \in \mathcal{H}_{\lfloor k/2 \rfloor}\}}$, where $\mathcal{H}_{\lfloor k/2 \rfloor}$ represents an hypercube from the partition

of $\mathcal{D}$. By construction this family of basis functions is orthogonal and, under the Lebesgue training measure, normalisation can be obtained with a straightforward rescaling. Notice that for different values of K all the functions change, rather than new ones getting added, as the same domain $\mathcal{D}$ is divided in a different number of hypercubes. Corollaries 2.3.5 and 2.5.5 provide explicit error bounds for all orthonormal functions. Notice that in practical implementations this choice can result in higher computational time due to the evaluation of $\mathbb{P}(x \in \mathcal{H}_{\lfloor k/2 \rfloor}) = \int_{\mathcal{H}_{\lfloor k/2 \rfloor}} r(n, x, y; u)\, dx$, necessary to compute the conditional expectation of a general affine function $(ax + b)\mathbb{1}_{\{x \in \mathcal{H}_{\lfloor k/2 \rfloor}\}}$, which might not be available in closed formula.

Radial basis functions

A less popular, but certainly interesting choice of basis functions when the domain $\mathcal{D}$ is high dimensional, is given by radial basis functions. This class of functions enjoys the property that $\phi_k(x) = \phi_k(\|x\|)$, making them well suited for multidimensional settings. A drawback of this approach is that, after having identified the most suitable class of radial functions, we still have to define two vector parameters for each function. The parameters are two matrices, one describing the position of the centroids while the other the bandwidths or rates of decay of the kernels. A common choice is the Gaussian family, represented by the kernel function $\phi_k(x) = \frac{1}{\sqrt{2\pi}\sigma_k} \exp \sum_{\delta=1}^{d} \frac{(x_\delta - \lambda_{\delta,k})^2}{2\sigma_{\delta,k}^2}$, with $\lambda \in (\mathcal{D})^K$ and $\sigma \in (\mathbb{R}^{d,K})^+$. Such system of functions is clearly normal, but not orthogonal. Recalling that the Gaussian density function is almost zero in the tails, however, it is possible to choose the parameters λ and σ such that, numerically, $\mathcal{A} \approx Id$.

In the following two sections we discuss two techniques that improve the computational complexity of Regression Monte Carlo. We focus in particular on speeding up the Performance Iteration procedure, that suffers from the multiple resimulations used to generate the samples that are projected during the regression

step. In the following two subsections we discuss the maximisation through gradient descent, which exploits the information available on the optimisation problem extracted from the knowledge that it is not a stand-alone problem, but falls within a dynamic estimation where similar problems are solved in a spatial and temporal neighbourhood. Secondly, we describe and generalise the backward construction of paths introduced by Carmona and Ludkovski [17] for inventory type of problems in the Performance Iteration framework.

5.3 Fast maximisation through gradient descent

We can look at the Regression Monte Carlo algorithm as the iteration of two fundamental operations: regression and optimisation. In the previous chapters, we have devoted most space to the discussion of the regression step, leaving the optimisation procedure to the reader to solve through standard techniques. In this section, we propose an approach that exploits the existing knowledge about the optimisation problem at one particular space-time location, gathered from problems already solved in a temporal neighbourhood. The approach works well with both Value and Performance Iteration, but it is in the latter that the greatest improvements are observed.

One of the reasons why Performance Iteration is considerably slower than Value Iteration is the bottleneck induced by the repetition of the deterministic optimisation problem necessary to determine the control to apply at each time step. At the same time, it can be noticed that time-step after time-step the pathwise control changes just slightly, proportionally to the variance of the controlled process itself. This suggests that we might be able to exploit this characteristic to speed up the computation of the optimal control, by exploiting the knowledge of the value of the control computed at the previous time steps (while moving backward). We turn our

attention to the gradient descent technique which is well known to converge very fast if the initial guess for the solution is accurate. By noticing that at terminal time the value function is known and equal to the terminal condition, and by computing the optimal control by extensive search at time $N-1$, we can initialise the gradient descent procedure with a good initial condition thereby guaranteeing that few steps of gradient descent iteration can account for the change in the optimal control from time $n+1$ to time n, caused by a change in the value of (X_n, I_n) and the time dependency of the control map.

Remark 5.3.1. *Notice that in Value Iteration, contrary to Performance Iteration, the training points* $\{X^m\}_{m=1}^M$ *can be kept fix at all time steps providing an easier framework to apply the gradient descent optimisation.*

5.3.1 Value Iteration

For simplicity, consider the Value Iteration case first. In particular, assume that at time $n+1$ the control map is known at a number of locations $\{u_{n+1}(X_{n+1}^m)\}_{m=1}^M$ and, at time n, we want to compute the control $u_n(x)$ at one of the points $X_n^m = x$. Therefore, set $u^0 = u_{n+1}(x)$ and iterate the following equation:

$$u^{\tau+1} = u^\tau + \lambda \frac{\mathcal{E}'(x,i,u)}{\mathcal{E}''(x,i,u)}\,,\ \tau = 0,\, 1,\, 2,\, \ldots$$

where we denote the first and second derivative of the continuation value and reward function by $\mathcal{E}'(x,u) = \frac{\partial f(n,x,u)}{\partial u} + \sum_{k=1}^K \alpha_k \frac{\partial \hat{\phi}_k(x,u)}{\partial u}$ and $\mathcal{E}''(x,u) = \frac{\partial^2 f(n,x,u)}{\partial u^2} + \sum_{k=1}^K \alpha_k \frac{\partial^2 \hat{\phi}_k(x,u)}{\partial u^2}$. We implicitly assume that derivatives of the conditional expectation of the basis functions can be quickly computed analytically. By λ we denote the learning parameter which should be tuned according to the nature of the problem, for reference see Press et al. [65]. Intuitively, consider that, for a quadratic function, $\lambda = 1$ allows to reach the global maximum (minimum) in one step. The

iterative procedure should be stopped once the relative error tolerance γ is reached, i.e. stop the gradient descent optimisation after $\mathcal{N}$ steps, where:

$$\mathcal{N} = \underset{\tau=1,2,\dots}{\operatorname{argmin}} \left\{ \tau : \left| 1 - \frac{f(n,x,u_n) + \sum_{k=1}^{K} \alpha_{n+1}^k \hat{\phi}_k(x,u_n)}{f(n,x,u_{n-1}) + \sum_{k=1}^{K} \alpha_{n+1}^k \hat{\phi}_k(x,u_{n-1})} \right| < \gamma \right\}$$

The implementation of this technique guarantees good solutions provided that the initial guess of the control is close to the true, optimal one.

5.3.2 Performance Iteration

One might notice now that, when approaching the Performance Iteration algorithm, the above procedure cannot be easily applied if the forward simulated paths are resimulated every time that we make a step backward. To overcome the difficulty of finding a suitable initial condition when simulating a new set of paths forward in time, we approximate the control map by regressing the samples $\{u_{t+1}(X_{t+1}^m)\}_{m=1}^M$ over a new set of basis functions. When the procedure has reached time n, we have already computed a regression approximation of the control map at time $t+1$, to be used at time t to set the initial guess of the gradient descent algorithm for all $t > n$ during the forward resimulation:

$$u_0^x = \sum_{k=1}^{K} \beta_k \phi_k(x)\,, \quad \text{where } \beta = arg\min_b \left\{ \left\| u_n^*(\cdot) - \sum_{k=1}^{K} b_k \phi_k(\cdot) \right\|_2 \right\}.$$

An example of the implementation of the gradient descent technique can be found in Algorithm 6.

Remark 5.3.2. *Notice that in order to increase the performance of the method, and to account for the fast-changing control due to the effect of the terminal condition, we compute the control via extensive search during the first backward steps and then switch to gradient descent when we believe that the behaviour of the control is more stable. The trade-off between computational time and precision will be further discussed in the numerical examples presented in the second part of this thesis.*

Algorithm 5 Maximisation via Gradient Descent algorithm

input: M, K, μ, $\{\phi\}_{k=1}^{K}$

[...]

$\hat{\theta}^n = \mathcal{A}_K^{-1} \frac{1}{M} \sum_{m=1}^{M} \hat{u}_n^*(\tilde{X}_n^m)\phi(\tilde{X}_n^m)$

for m=1 to M **do**

    $u_0 = \sum_{k=1}^{K} \hat{\theta}_k^n \phi_k(\tilde{X}_n^m)$, $h = 1$

    while $\frac{u_h - u_{h-1}}{u_{h-1}} > \gamma$ **do**

        $u_{h+1} = u_h + \lambda \frac{\mathcal{E}'(\tilde{X}_n^m, u_h)}{\mathcal{E}''(\tilde{X}_n^m, u_h)}$

        $h = h + 1$

    $u_n^* = u_h$

$$\hat{V}(n, \tilde{X}_n^m) = f(n, \tilde{X}_n^m, u_n^*) + \sum_{k=1}^{K} \hat{\alpha}_k^{n+1} \hat{\phi}_k^n(\tilde{X}_n^m, u_n^*)$$

output: $\{\hat{\alpha}_n^k\}_{k=1}^{K}$

5.4 Backward construction of inventory levels for Performance Iteration

The primary cause of computational complexity when using Performance Iteration for inventory problems is the need to resimulate the inventory level until maturity after each backward step. Even though this problem is alleviated by the use of gradient-like techniques for optimisation, as explained in 5.3, a considerable difference is still observable between Value and Performance Iteration. When the algorithm moves closer to the initial time, the length of trajectories that need to be resimulated grows. Eventually, the computational effort of resimulation is proportional to N^2 compared to a linear cost of simulating the process (X_n).

Notice that in order to avoid resimulation to compute $J(n, (X_j^m, I_j^m, u_j^*)_{j=n,\dots,N})$, as

explained in section 4.1, one needs to be able to reuse $J(n{+}1, (X_j^m, I_j^m, u_j^*)_{j=n+1,\dots,N})$. Interpolation, as it was used in Grid Discretisation to connect samples of the value function, it is not a viable alternative because $J(n, (X_j^m, I_j^m, u_j^*)_{j=n,\dots,N})$ cannot be represented in terms of (X_n, I_n) only, as it depends on the whole future trajectory $(X_j^m, I_j^m)_{j=n,\dots,N}$.

5.4.1 A fix point problem

Here we propose a solution in which we aim to place points I_n^m in such a way that the optimally controlled process reaches the state $(X_{n+1}^m, I_{n+1} = I_{n+1}^m)$ at time $n+1$ and therefore the existing path $(X_j^m, I_j^m)_{j=n+1,\dots,N}$ is a valid successor of (X_n^m, I_n^m) enabling reuse of $J(n+1, (X_j^m, I_j^m, u_j^*)_{j=n+1,\dots,N})$. From the optimality conditions in section 1.2.2, we see that the pathwise optimal antecedent of a point I_{n+1}^m is a solution $y \in [0, I_{\max}]$, if exists, of the equation

$$I_{n+1}^m = \varphi\Big(n, X_n^m, y, u_n^*(X_n^m, y)\Big). \tag{5.4.3}$$

In the following lemma, we explore the existence of such y. Define two quantities $\underline{\varphi_n} = \max\limits_{u\in[\underline{u},\overline{u}]}\{\varphi(n,u,0)\}$ and $\overline{\varphi_n} = \min\limits_{u\in[\underline{u},\overline{u}]}\{\varphi(n,u,I_{\max})\}$ and assume $0 \le \underline{\varphi_n} < \overline{\varphi_n} \le I_{\max}$, $\forall\, n$.

Lemma 5.4.1. *A solution $y \in [0, I_{\max}]$ to (5.4.3) exists if the following conditions are satisfied:*

1) $I_{n+1}^m \in [\underline{\varphi_n}, \overline{\varphi_n}]$,

2) the control function $u_n^*(X_n^m, y)$ is continuous with respect to y.

Moreover, if the map $y \mapsto \varphi(n, u_n^*(X_n^m, y), y)$ is strictly monotone then the solution is unique.

Proof. Introduce the function $\psi(y) = I_{n+1}^m - \varphi(n, u_n^*(X_n^m, y), y)$ and notice that if condition 2) is satisfied the continuity of ψ is enforced by φ. Evaluate ψ it at the boundaries of $[0, I_{\max}]$:

$$\begin{aligned}\psi(0) =& I_{n+1}^m - \varphi(n, X_n^m, 0, u_n^*(X_n^m, 0)) \\ \geq& \underline{\varphi_n} - \underline{\varphi_n} = 0; \\ \psi(I_{\max}) =& I_{n+1}^m - \varphi(n, X_n^m, I_{\max}, u_n^*(X_n^m, I_{\max})) \\ \leq& \overline{\varphi_n} - \overline{\varphi_n} = 0.\end{aligned}$$

Therefore, if condition 1) is satisfied it must exists a point $y \in [0, I_{\max}]$ such that $\psi(y) = 0$, i.e., solve equation (5.4.3). In addition, if $y \mapsto \varphi(n, x, y, u_n^*(x, y))$ is strictly monotone then, so is ψ, and hence there exists only one y which solves equation (5.4.3). □

5.4.2 Algorithm

Algorithm 6 Backward construction of paths algorithm

1: Let y be a solution to $I_{n+1}^m = \varphi\Big(n, u_n(X_n^m, y), y\Big)$.
2: **if** solution exists and $y \in [0, I_{\max}]$ **then**
3: Set $I_n^m = y$
4: $V(n, X_n^m, I_n^m) = f\Big(X_n^m, I_n^m, u_n^*(X_n^m, I_n^m)\Big) + V\Big(n + 1, X_{n+1}^m, \varphi(n, u_n^*(X_n^m, I_n^m), I_n^m)\Big)$
5: **else**
6: Generate I_n^m randomly in $[0, I_{\max}]$
7: Resimulate to compute $V(n, X_n^m, I_n^m)$

Lemma 5.4.1 ensures the existence of solution to the equation (5.4.3) even though such solution could be found also for weaker assumptions, possibly outside the interval $[0, I_{\max}]$. Whenever this solution does not lie in $[0, I_{\max}]$, i.e. $I_{n+1}^m \notin [\underline{\varphi_n}, \overline{\varphi_n}]$

, it is not possible to determine a step backward I_n^m for the inventory variable. In such case, we propose to break the path and position the point I_n^m at random in $[\underline{\varphi_n}, \overline{\varphi_n}]$. Following this step, we resimulate the path $\{I_s^m\}_{s=n+1,\dots N}$ as in the traditional Performance Iteration algorithm. The random positioning of training points I_n^m when the solution of the constrained fixed point problem (5.4.3) cannot be found avoids the clustering of training points in certain areas of the state space.

Remark 5.4.2. *The phenomenon described above that there is no solution to* (5.4.3) *that satisfies the bounds on the inventory level is not unusual in many practical examples and, in particular, does not result from approximation errors or any other features of the numerical scheme. In section 6.2 we consider trading a commodity, with price* (X_t)*, subject to constraints on the size of storage. An optimal behaviour, when the price is high, is to sell. However, if this happens, optimal trading would never lead to a full inventory at the next time period,* $I_{n+1}^m = I_{\max}$*. In fact, with high commodity prices over a period of time and a selling strategy, the backward constructed trajectory* $(I_s^m)_{s=n}^N$ *would climb up (when time goes backwards) finally hitting the level* $I_{\max}$*.*

Example. *It is commonly seen in the inventory problems that the control represents additive adjustments to the inventory level, i.e.* $\varphi(n, u, i) = i + u$*. equation* (5.4.3) *simplifies to*

$$I_{n+1}^m = y + u_n^*(X_n^m, y).$$

Denoting $\Delta y = I_{n+1}^m - y$ *we obtain a classical fixed point problem*

$$\Delta y = u_n^*(X_n^m, I_{n+1}^m - \Delta y).$$

Lemma 5.4.1 implies that there exists an unique solution $\Delta y \in [\underline{u}, \overline{u}]$ *for every* $I_{n+1}^m \in [\overline{u}, I_{\max} + \underline{u}]$*.*

Part II

Applications and Numerical Experiments

Introduction

In this second part of the Thesis, we will discuss more practical aspects linked with the implementation of Regression Monte Carlo algorithms. In particular, for the problems introduced in Part I, we will discuss a number of examples and toy problems to benchmark the performance of Regress Later against competing algorithms, as well as showing some characteristics of the Regression Monte Carlo method itself, including the differences between Value and Performance Iteration and the effect of the improvements presented in chapter 5.

We provide now an overview of the problems we deal with in this Part II of the thesis: We open with inventory problems, first a simple price arbitrage problem where the controller has to increase or decrease the holdings of a certain commodity which he buys/sells according to a stochastic price. We use this problem to benchmark Regress Later against Control Randomisation and Grid Discretisation, as well as display the use of backward construction of paths and gradient descent (see section 5.3-5.4). We then introduce a similar problem inspired by the control of hydro reservoirs which features a two-dimensional inventory and serves as a demonstration of the weaknesses of the Grid Discretisation approach. We conclude this chapter with an example from finance about the liquidation of a big portfolio of shares on the market with price impact and partial information on the drift of the underlying stock price. This example, introduced in Balata et al. [5], provides a benchmark for Regress Later against Control Randomisation. We conclude with four problems of increasing difficulty, where the control affects the dynamics of the Markov Process X. The first one, is a univariate linear quadratic problem for which we show convergence to the continuous time analytical solution as $N \to \infty$. The following problem, inspired by engineering applications, deals with the control of a robot in a disrupted environment. The idea is that the controller has to steer the robot in order for it to avoid obstacles. We exploit this problem to compare the Value and Performance

Iteration approaches and provide some numerical evidence of the claims made in section 5.1. The final problem in this Part of the Thesis is about systemic risk and the optimal control of monetary reserves by a central bank. Such problem, similarly to the portfolio liquidation example, is used to benchmark Regress Later against Control Randomisation.

All algorithms, unless otherwise stated, are implemented in MATLAB and are optimised in a similar fashion in order to reveal the relative differences in their computational complexity.

Chapter 6

Inventory management

6.1 Introduction

This chapter collects numerical results to benchmark the performance of the algorithms introduced in Chapter 3. The first example is a "price arbitrage" problem in the context of energy trading for which we compare the accuracy and running time of Grid Discretisation (GD), Control Randomisation (CR) and the Regress-Later Monte Carlo (RLMC) approach in both Value and Performance Iteration.

The second control problem features a two-dimensional inventory in the context of managing a system of hydro reservoirs. The aim is to maximise the profit from selling the electricity produced by a set of turbines installed in the pipes connecting the reservoirs. The mathematical formulation is similar to the previous problem, however, the higher dimension of the inventory highlights the impact of dimensionality on the computational complexity of GD.

The third problem we consider is optimal portfolio liquidation under partial information. In particular, we consider the task of selling a large number of shares, thereby turning the price against ourselves, without knowing the drift that drives

the dynamics of the price process. In this case, we compare RL and CR algorithms.

6.2 Price arbitrage problem

6.2.1 The model

Consider the following specification of the dynamics of a commodity price as a mean reverting AR(1) process, along with the dynamics of the inventory levels for such commodity:

$$\begin{cases} X_{n+1} = X_n + 2(5 - X_n)\,\Delta + 5\,\xi_n, \\ I_{n+1} = I_n + u_n\,\Delta\,, \;\; u_n \in \mathcal{U}_n, \;\; \Delta = \frac{1}{200} \end{cases} \tag{6.2.1}$$

where $\mathcal{U}_n = \{u_n \in \{-11.5, 0, 11.5\} : \; 0 \leq \varphi_I(n, I_n, u) \leq 1 \; \forall t\}$. The trader goal is to maximise the profits from buying the commodity at times when its price is low and selling it when it is high. We assume that the trader has enough capital to carry over her trades and only accounts for her constraints on the inventory space available. The functional describing her expected profits can be written as:

$$\mathbb{E}\Big[\sum_{s=0}^{N} -(u_s + \rho_{u_s})\,X_s\,\Delta + \frac{1}{2} X_N\,I_N | X_0 = x, I_0 = i\Big], \tag{6.2.2}$$

where ρ_{u_s} introduces a transaction fee associated with each control: $\rho_{u_s} = \mathbb{1}_{\{u_s \neq 0\}} 2$.

The value function associated with this problem can be written as:

$$V(t, x, i) = \max_{u \in \mathcal{U}_n} \Big\{ -(u + \rho_u)\,X_n \Delta + \mathbb{E}\Big[V(n+1, X_{n+1}, i+u) | X_n = x\Big] \Big\}.$$

6.2.2 Numerical implementation

We compute an estimation of the value function and the optimal policy associated with it, implementing some of the algorithms introduced in chapter 3: GD, CR

and RLMC. As discussed in chapter 5, before implementing these methods, basis functions and training measures should be defined. The basis in RL algorithms are composed by polynomials of up to order two in X and I, i.e. $\{1, x, i, xi, x^2, i^2, x^2i, xi^2, x^2i^2\}$. Basis functions for CR have to also include the control variable, so we employed a basis comprising polynomials up to order two in X, I and u, i.e. $\{1, x, i, u, xi, x^2, i^2, u^2, xu, iu\}$. For GD we discretised the inventory in L levels and then used polynomial basis functions up to order two for X, i.e. $\{1, x, x^2\}$. We used uniform training measures on the relevant supports: for RL uniform price and inventory (with price bounded in $[0, 10]$) , for CR the random control values are chosen with equal probabilities of $1/3$, while for GD the discretization points of the inventory have been chosen equispaced.

6.2.3 Comparison

We run our experiments selecting various values for the number of simulated paths and grid points (for Grid Discretisation) or simulated paths (for Control Randomisation) or training points (for Regress Later) and evaluate the resulting policies over a fixed set of simulations shared between all methods and runs in order to minimise the effect of the Monte Carlo error on relative results. We record running time and value of estimated policies and display them in Figure 6.1. The results obtained through Value Iteration are shown on the left panel. Notice that RL is more efficient than all other methods; it can achieve similar levels of accuracy to the other algorithms in less time. GD, on the other hand, is the slowest method, but it can nonetheless achieve high levels of accuracy. CR performs slower to RL as it suffers from the need of fitting a higher number of basis functions.

In the Performance Iteration specification, see the right panel of Figure 6.1, we only display two versions of the RL algorithm: with and without backward construction of paths. CR and GD are less efficient than in the Value Iteration case with the latter

requiring disproportionally long computational time due to multiple resimulations for each discretisation point.

RL algorithm without resimulation of paths is slower, but more accurate, than the one with backward construction of paths. This improved accuracy may be explained by the resimulation present in the former while the latter uses the same training points for both estimations of the optimal policy and its evaluation, introducing bias. This phenomenon has already been observed in literature in other contexts. The use of backward construction of the paths reduces, for this particular problem, the number of resimulated paths to about 25% of the original value as approximately one-quarter of paths are broken at each time step. The lower precision of the RL with backward construction of paths combined with its lower computational complexity makes it useful only when the time constraint is particularly tight. We observe that for problems with continuous space of controls the improvement offered by this backward simulation method is considerable, as the optimization problem to be solved at each time step of the resimulation procedure is more time consuming than in the present problem where controls can only take one out of three values.

The difference in performance of estimated policies between Value and Performance Iteration observed in this example might be explained as follows. Firstly, the Performance Iteration problem has been slightly modified considering shorter timesteps $\Delta = 1/1000$ rather than $\Delta = 1/200$ in order to make the resimulation step more burdensome and highlight the differences between the algorithms with and without backward construction of paths. Secondly, the functional form of the terminal condition and running reward make this optimisation problem linear quadratic over most of the domain (non-linearities are encountered at the boundaries of the inventory) indicating that basis function can replicate well the true value function, limiting the propagation of the error. In these cases the use of Performance Iteration is rarely advised as it implies higher variance of the points used to fit the

regression parameters, reducing the estimation precision.

We try to improve our results by applying the gradient descent technique introduced in section 5.3. We display our results in the top panel of Figure 6.2. We repeat our estimation by increasingly using gradient descent estimation of the optimal control from the initial time to the terminal one. It can be noticed that Value Iteration behaves as expected, sooner we start using gradient descent in the backward procedure, lower the computation time and the precision of the method. Performance Iteration, on the other hand, behaves differently: by increasing the use of gradient descent we gain precision in expectation while the results degenerate in terms of variance. A better representation of this phenomena might be observed in the bottom panel of Figure 6.2 where the box-plots show the performance of the policy estimated by using gradient descent 5 steps after the terminal condition, i.e from $N-5$ to initial time. Our interpretation of this result is that the representation of the control map using regression, in order to provide the starting point to the gradient descent routine, allows the method to gain in stability and thereby to achieve better results in expectations. At times, however, such representation might not provide an initial guess that is good enough and causes, therefore, the few, but severe, observed deviations from the mean value.

6.3 System of pumped hydro-reservoirs

In this section, we extend the previous problem to a case in which the inventory is two dimensional.

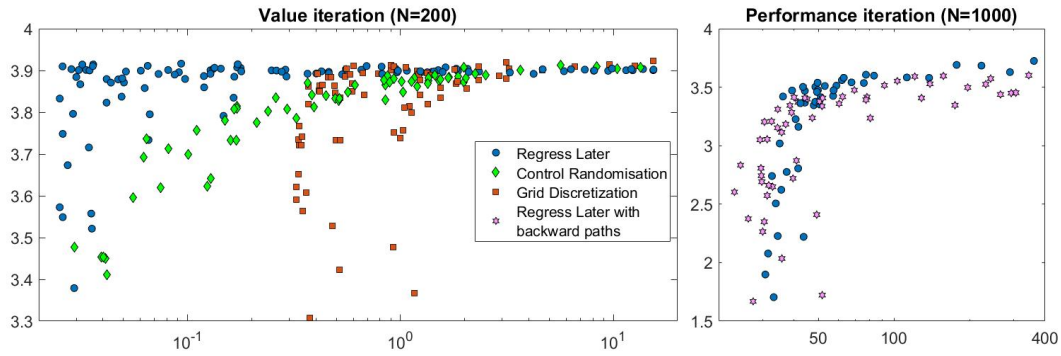


Figure 6.1: Trade-off between estimation time and performance of policies obtained from RL (with and without backward construction of paths), CR and GD. The horizontal axis shows time. The vertical axis displays the performance of estimated policies computed using a fixed set of 1000 simulations. Notice that Value Iteration uses $N = 200$ time steps, while Performance Iteration uses $N = 1000$, to amplify the effect of resimulation.

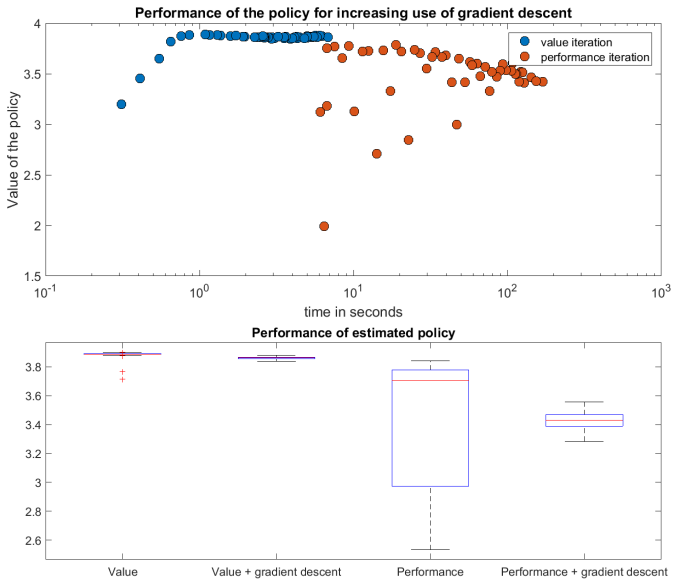


Figure 6.2: The top panel shows the value of the policy estimated by Value and Performance Iteration for an increasing use of gradient descent, plotted against the running time. The bottom panel describes the variance of the estimations obtained on a fix set of forward simulations.

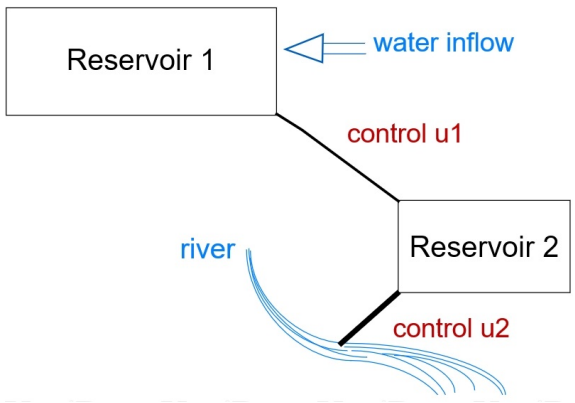


Figure 6.3: Diagrammatic representation of a system of interconnected reservoirs.

6.3.1 The model

We consider a problem of managing a system of two interconnected hydro reservoirs with turbines and pumps, see Figure 6.3. Similarly to the previous case, we model the electricity price as AR(1) process:

$$X_{n+1} = X_n + \alpha(\mu - X_n) + \sigma\xi_n, \qquad \xi_n \sim \mathcal{N}(0,1),\ i.i.d.,$$

where one time step represents 30 seconds. We aim to profit from buying and selling electricity. We use the parameters $\sigma = 1$, $\alpha = 0.1$ and $\mu = 40$.

Reservoir 1 is fed with a constant water inflow equivalent to 120MW (this assumption is reasonable given the 3 hours horizon). The connections between Reservoir 1 and Reservoir 2 is equipped with a pump and a turbine with production/consumption capacity of 600 MW (for simplicity of presentation but without any loss of numerical difficulty we assume 100% round-trip efficiency). The connection of Reservoir 2 with the river can accommodate a higher flow of water and contains a set of turbines with the maximum generation capacity of 1.2 GW. The size of Reservoir 1 is 2 GWh, the size of Reservoir 2 is 1 GWh. The numbers chosen for this example have been inspired by the facilities installed in Dinorwig, UK, and in multiple locations in Norway.

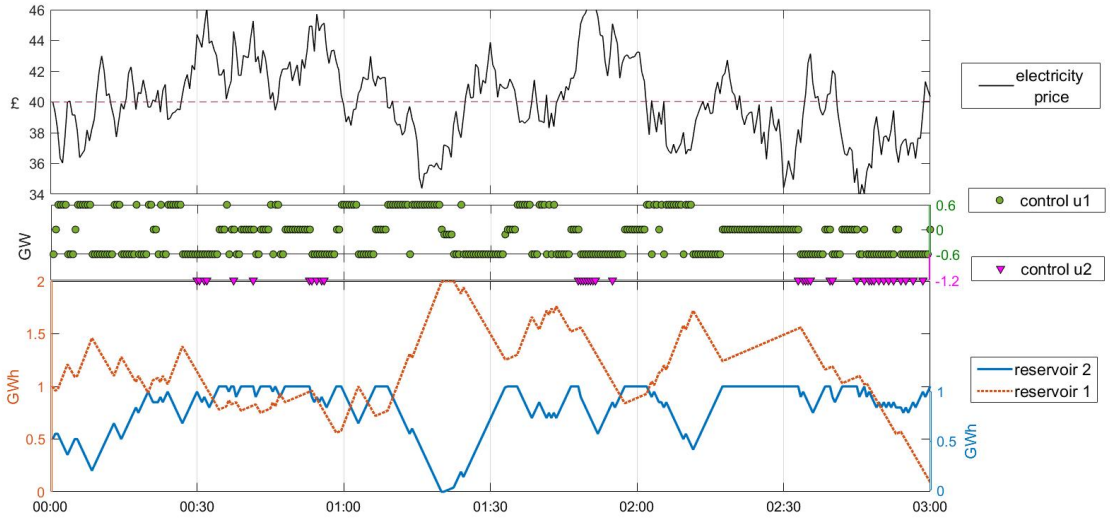


Figure 6.4: System behaviour over 3 hours period. In the top panel we display the electricity price; in the middle the control, u^1 in green and u^2 in violet, in the bottom panel, in blue inventory 2 (1 GWh capacity) and in orange inventory 1 (2 GWh capacity).

The dynamics of the energy stored in the reservoirs follow:

$$I^1_{n+1} = I^1_n + u^1_n + 0.12,$$

$$I^2_{n+1} = I^2_n - u^1_n + u^2_n,$$

where $u^1_n \in [-0.6, 0.6]$ is the flow between Reservoir 1 and 2, $u^2_n \in [0, 1.2]$ is the flow from Reservoir 2 to the river, and 0.12 is the natural inflow to Reservoir 1. We define the performance measure over three hours time horizon as follow:

$$J(n, (X_s, I^1_s, I^2_s, u^1_s, u^2_s)_{s=n}^N) = \sum_{s=1}^{N} -X_s(u^1_n + u^2_n) - A(I^1_N + I^2_N - 1.5)^2$$

where the terminal condition, with an arbitrary coefficient $A = 50$, incentivise the controller to keep the total energy stored in both reservoirs at the terminal time at 50% of the maximum of 3 GWh. An example of the optimal strategy estimated by RL is displayed in Figure 6.4

Table 6.1: Performance of estimated policies evaluated on a fixed set of 1000 paths. For comparison, the myopic (greedy, minimises current cost only) policy's performance is 274.

	Regress Later			Grid Discretisation		
value	340	355	357	290	328	357
time	0.2 s	0.5 s	6 s	4 s	7.5 s	35 s

6.3.2 Comparison

We extended our implementation of Grid Discretisation to this problem to show the effect of the dimension of inventory on its performance. Consider for example that if we take 7 discretisation points for each storage, we obtain a total of 49 possible configurations, for which the estimation of regression coefficients and computation of the optimal control has to be repeated. This leads to significantly lower computational efficiency compared to the Regress Later algorithm, see Table 6.1 which lists running time and values implied by the estimated policies for RL and GD.

6.4 Portfolio liquidation under drift uncertainty

In this section, we present an additional example of inventory problem from the field of finance. Originally this problem was solved within the project SOCMKV initiated during CEMRACS 2017, a 5-week research retreat at CIRM in Luminy, near Marseille, France. The project was supervised by Huyên Pham and Mathieu Laurière. The final paper [5] features a theoretical discussion on the Markovian Embedding of the Conditional McKean-Vlasov control problems into finite-dimensional ones and a series of examples where the performance of RL, CR and the Quantization algorithm (which is outside of the scope of this thesis) are

compared. The contribution of the author in such project resides in implementing and designing the numerical solution of the example problems using RL and CR which will be discussed in the following.

6.4.1 The model

Consider an uncontrolled Markov process governed by the following drift-less dynamics

$$X_{n+1} = X_n + X_n \sigma \xi_n$$

and an inventory process $I_{n+1} = I_n + u_n$ representing the number of shares held in portfolio. The control process u_n represents the amount of shares bought or sold at each time step. In this context, we consider an important optimisation problem in finance: portfolio liquidation. We consider the problem of an agent (trader) who has to liquidate a large number i_0 of shares in some asset, within a finite number of time-steps N, and faces execution costs and market price impact.

The dynamics chosen for the process X, which is meant to represent the price of the security we want to liquidate, have been computed to account for the non-observability of the original price process. This model, inspired by Balata et al. [5], has been originally cast in continuous time under partial information about the drift of the price process P:

$$dP_t = P_t(\beta_t dt + \sigma dW_t^0),$$

where W_t is a one dimensional Brownian motion, the drift β_t of the asset (which is typically a diffusion process governed by another independent Brownian motion) is unknown and unobservable like the Brownian motion W. The agent can actually only observe the stock prices P. In this problem, we consider the special case when $\beta_t = \beta$ is a random variable distributed according to some probability distribution ν: this corresponds to a Bayesian point of view when the agent's belief about the

drift is modelled by a prior distribution. In this case, we follow the approach in Balata et al. [5] to embed our partial observation problem into a finite-dimensional, full observation, Markov control problem. We call the discretised process driven by the filtered dynamics X, and the total liquidation cost in the filtered framework

$$J(n,(X_s,W_s,I_s,u_s)_{s=n}^N) = \sum_{s=n}^{N} F(s,W_s)u_s(X_s+f(u_s)) + F(N,W_N)g(I_N),$$

where f is an increasing function, $f(0)=0$, representing a temporary price impact, g is a loss function, i.e. a convex function with $g(0)=0$, penalising the trader when she does not succeed to liquidate all her shares, the Brownian motion process is defined as $W_{n+1} = W_n + \xi_n$ and the function:

$$F(t,w) := \int \exp\Big(\frac{bw}{\sigma} - \frac{1}{2\sigma^2}b^2 t\Big)\nu(db).$$

6.4.2 Numerical implementation

Let us now illustrate numerically the impact of uncertain Bayesian drift on the portfolio liquidation problem by considering a Gaussian prior distribution $\beta \rightsquigarrow \nu = \mathcal{N}(b_0,\gamma_0^2)$. In this case, the function F is explicitly given by:

$$F(t,w) = \frac{\sigma\gamma_0}{\sqrt{\sigma^2+\gamma_0^2 t}} \exp\left(\frac{1}{2(\sigma^2+\gamma_0^2 t)}(-b_0^2 t + 2b_0\sigma w + \gamma_0^2 w^2)\right).$$

Let us consider a linear price impact function $f(a) = \gamma a$, $\gamma > 0$, and a quadratic loss function $g(i) = \eta i^2$, $\eta > 0$.

Remark 6.4.1. *Notice that when the price process is a martingale, i.e. $b_0 = 0$, and in the limiting case when the penalty parameter η goes to infinity, corresponding to the final constraint $I_t = 0$, we see that, in continuous time, u_t^* converges to $-I_t^*/(T-t)$, hence independent of the price process, and leading to an explicit optimal inventory: $I_t^* = i_0\frac{T-t}{T}$ with constant trading rate $-i_0/T$.*

We solve the problem numerically, fixing the parameters as follows: γ=5, X_0=6, I_0=1, η=100 and σ=0.4. We run two sets of forward Monte Carlo simulations for $b_0 = 0.1$, $T = 1$ and $b_0 = -0.1$, $T = 0.5$ changing the value of γ_0 and testing the RL and CR algorithms.

In particular, we compare the performance of these algorithms against the control policy $(u_n^*)_{n=0}^{N-1}$, where u^* is the optimal strategy associated with the continuous-time portfolio liquidation problem (we indicate this strategy by OPT), for more details refer to Balata et al. [5]. We also tested a benchmark strategy (we indicate this strategy by B) which consists in liquidating the inventory at a constant rate of $-I_0/N$. The test consisted in computing the estimates $\hat{V}(0, X_0 = 6, I_0 = 1)$ associated with the different algorithms.

We display the results obtained by the different algorithms in Table 6.2 and plot them in Figure 6.5.

6.4.3 Comparison

The implementation of Regression Monte Carlo algorithms has required intense tuning and the use of the Performance Iteration technique in order to obtain satisfactory results. Paramount is, in addition, the distribution chosen for the training points in Regress Later and for the initial control in Control Randomization. The problem of finding the best set of data to provide to the backward procedure is similar in the two Regression Monte Carlo algorithms. However, little study is available in the literature; for more details on this problem in the Regress Later setting see chapter 5, [61] and [4]. In the case of RL algorithm a training measure μ_n has been chosen in order to sufficiently explore the state space in the I dimension, in particular we considered $\mu_n = \mathcal{U}[-0.5, 0.5 + \frac{T-t_n}{t_n}]$. The proofs we presented in chapter 2 can be adapted to the time-dependent training measure μ_n considering

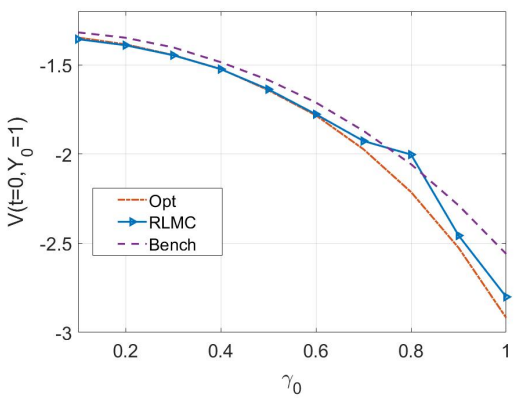
that the only difference is to update the bound $\bar{R} \mapsto \bar{R} \max_n \left\{ \left\| \frac{d\mu_n}{d\mu} \right\| \right\}$. Similarly, for CR we seek a distribution of the random control such that the controlled process I results in having a distribution similar to μ_n.

In order to choose the basis functions, we used the fact that we expect the value function to be convex in the I dimension with a minimum around the optimal inventory level and monotone in the X dimension. For RL algorithm we choose therefore the following set of basis functions: $\{x, i, i^2, xi, xi^2\}$, where we take the square function i^2 as a general approximator for convex functions around their minima (where we expect the measure μ_n to be concentrated). On the other hand, CR requires that we guess what the functional form of the conditional expectation of the value function is with respect to the control process. Considering our argument on second order polynomials approximating general convex functions, we choose to add the set $\{u, u^2, ui, ux\}$ to the set of basis functions used by RL.

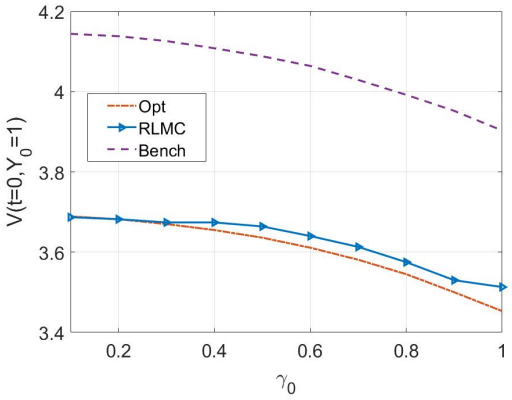
Note that we observed very high volatility in the quality of the policy estimated by control randomization. For this reason, we estimated the policy 50 times, and report in Table 6.2 the results provided by the best performing one; increasing the number of training points further affects the variability only marginally.

Table 6.2: Portfolio liquidation results. Estimations of the value functions at point ($x_0 = 6, i_0 = 1$) and time 0 provided by different algorithms.

	$b_0 = 0.1$, $T = 1$				$b_0 = -0.1$, $T = 1/2$			
γ_0	OPT	RLMC	CR	B	OPT	RLMC	CR	B
0.1	-1.347	-1.356	-1.278	-1.318	3.689	3.687	3.995	4.144
0.2	-1.385	-1.390	-1.283	-1.348	3.682	3.682	3.847	4.138
0.3	-1.445	-1.446	-1.314	-1.402	3.670	3.674	4.034	4.126
0.4	-1.523	-1.524	-1.323	-1.485	3.655	3.674	4.128	4.108
0.5	-1.642	-1.637	-1.348	-1.585	3.636	3.664	4.243	4.088
0.6	-1.783	-1.777	-1.425	-1.711	3.611	3.640	4.386	4.064
0.7	-1.973	-1.927	-1.513	-1.870	3.581	3.613	4.783	4.029
0.8	-2.213	-2.003	-1.637	-2.057	3.545	3.575	5.142	3.992
0.9	-2.526	-2.457	-1.819	-2.288	3.500	3.530	5.345	3.952
1	-2.918	-2.801	-1.806	-2.560	3.453	3.513	6.765	3.903



$b_0 = 0.1$ and $T = 1$.



$b_0 = -0.1$ and $T = 0.5$.

Figure 6.5: We display the estimated value function for the portfolio liquidation problem at point ($x_0 = 6, i_0 = 1$) and time 0, provided by RLMC, B and OPT w.r.t. γ_0. We took γ=5, X_0=6, I_0=1, η=100 and σ=0.4.

Chapter 7

Fully controlled problems

In this chapter, we discuss different examples to showcase some of the characteristics of Regression Monte Carlo, including the differences between Value and Performance Iteration which will be discussed at length in section 7.2. In particular, we present 3 different numerical applications, for reference, we list them here with growing difficulty: The first example is a linear-quadratic problem in dimension 1, which features a well known analytical solution in the continuous time case which we can use as a benchmark; the second example similarly features linear dynamics and quadratic cost but with a glitch, we add on the path of the controlled process some doors and charge a significant cost whenever these are missed. We use this problem to compare Value and Performance Iteration and offer a comment on their differences. It is interesting to notice how such small modification makes the problem so much harder. The third problem we present is a more realistic model inspired by Carmona et al. [18] on the systemic risk management by a central bank who can influence the dynamics of the cash reserve of the other banks operating under its jurisdiction.

7.1 Univariate linear-quadratic maximisation

In order to provide evidence of the convergence of Value and Performance Iteration algorithms, we briefly present a linear quadratic problem in one dimension, for which analytical solution is available for the continuous time problem (see Balata et al. [5]).

7.1.1 The model

Consider a process X driven by the following dynamics:

$$X_{n+1} = X_n + \frac{1 + X_n + u_n}{N} + \frac{\xi_n}{2N},\ X_0 = x_0,\ \xi_n \sim \mathcal{N}(0,1)$$

define the cost functional J

$$J\Big(n, (X_s, u_s)_{s=n}^N\Big) = \sum_{s=n}^{N} \frac{X_s^2 + u_s^2}{N} + X_N^2$$

and the value function $V(n,x) = \inf_{u \in \mathcal{U}} \Big\{ J\Big(n, (X_s, u_s)_{s=n}^N\Big) \Big\}$.

7.1.2 Numerical implementation

We choose $N = 100$ and solve this problem using Value and Performance iteration. We compare the value of the estimated policies with the value function of the continuous time problem. The relative distance from the continuous time solution as a function of the initial point x_0 (hereafter, for simplicity, relative error), is displayed in figure 7.1.

We explore further the effect of the time discretisation on the solution produced by Value Iteration by letting N vary and studying the behaviour of the relative error. In table 7.1 we display the estimated value of the policy and relative error for

N	value	error
50	1.13	4.2%
100	1.1489	1.92%
500	1.1654	0.51%
1000	1.1676	0.32%
5000	1.694	0.17%

Table 7.1: Convergence of Value Iteration to the continuous time solution, for $X_0 = 0$, 1.1714 as $N \to \infty$. All experiments have been run on the same set of 50000 simulations producing similar 95% confindence intervals 0.5% wide.

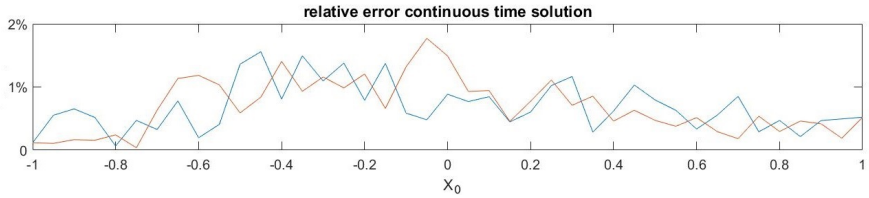


Figure 7.1: We show, in relative terms, the distance between the value function of the Linear Quadratic continuous time problem and the value of the policies estimated by the algorithms, Value Iteration in blue and Performance Iteration in orange, as a function of the initial condition x_0

$N = 50, 100, 500, 1000, 5000$. The Forward Evaluation has been run on a common set of 50000 paths to minimise the effect of randomness on the comparison; a 95% confidence interval of ± 0.025 has been identified for all examples. Figure 7.2 offers a graphical representation of the convergence for increasing values of N.

7.2 Control of a robot through a set of doors

In this experiment, introduced in example 1.1.1, we propose a toy problem quite simple to understand, but whose optimal policy is difficult to learn for the

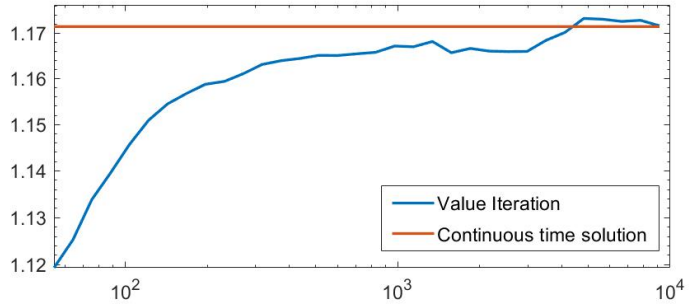


Figure 7.2: We show the convergence of Value Iteration to the continuous time solution, for $X_0 = 0$, 1.1714 as $N \to \infty$. All experiments have been run on the same set of 50000 simulations producing similar 95% confindence intervals 0.5% wide.

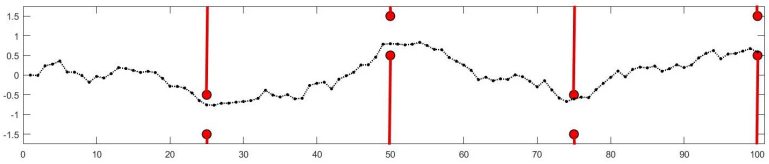


Figure 7.3: Graphical representation of the scheme of doors (7.2.1), which acts as a system of doors through which we want to drive the robot. Superimposed an example of controlled trajectory.

algorithms. Notice that this problem is mathematically very similar to the previous one, with the addition of a discontinuous cost function at four instants of time over the whole optimisation horizon. To help intuition, imagine we are controlling a robot through a system of rooms connected via some doors indicated by $[d_\tau^-, d_\tau^+]$. The robot is advancing over some unstable debris, causing its actual movement to be stochastic around the inputs provided by the controller.

7.2.1 The model

Consider a controlled autoregressive process

$$X_{n+1} = \left(X_n + \frac{u_n}{100} + \frac{1}{10}\xi_n\right) \vee -2 \wedge 2, \; n = 0, \ldots, N$$

and the task of guiding a robot through a sequence of rooms, as illustrated in figure 7.3, minimising the use of the control. Notice that for simplicity we have modelled the temporal dimension of the process as a spacial one, under the assumption that the robot moves at variable speed, advancing one unit of space towards the end of the last room ($n = N$) each unit of time. In order to model this problem in the framework of chapter 1, we introduce following cost functional

$$J(n, (X_s, u_s)_{s=n}^N) = \sum_{s=n}^{N} a\mathbb{1}_{\{u_s \neq 0\}} + b(u_s)^2 + c\mathbb{1}_{\{s=\tau_s\}}\mathbb{1}_{\{X_s \notin [d_{\tau_s}^-, d_{\tau_s}^+]\}}, \tag{7.2.1}$$

where a and b represent a fix and a quadratic-proportional cost for using the control, while c is the penalty for hitting the wall.

7.2.2 Numerical implementation

For this problem we select the set of basis function given by $\{1,\ x,\ x^2\}$, while we will test different choices of the training measure μ. We fix the following parameters: $a = 0$, $b = 1$, $c = 100$, $N = 100$, $\tau_s = 25, 50, 75, 100$ and the doors have width 1 and disposed as in Figure 7.3.

The purpose of this problem is to show the difference between Value and Performance Iteration. As explained in section 5.1, it has the main characteristics to do so. Notice that you can refer to figure 5.1 to see the effect of the measure μ on the approximation of a function similar to the shape of the doors $\mathbb{1}_{\{X_N \notin [d_{\tau_s}^-, d_{\tau_s}^+]\}}$. We would like to pick the training measure that induces the best policy. In order to improve the numerical results, and given the peculiar structure of this problem, we introduce a time-dependent training measure μ_n. The intuition is that the measure we choose should guide the training points through the rooms, inducing an effective policy. We use a heuristic technique to generate a sensible training measure μ_n: we first run an iteration of the backward procedure using a stationary uniform

training distribution μ, we then run the forward procedure and use the distribution $\mu_n = \mathcal{L}\{X_n\}$ estimated from it to run the following backward iteration.

7.2.3 Exploration vs. exploitation

The choice of the training measure μ_n, in this problem, well exemplifies the concept of exploration and exploitation. In order to solve this problem, in fact, both are necessary. The problem has been already discussed in section 5.1 at the functional approximation level, where the idea of learning the value function over a bigger portion of the state space is countered by the decreasing accuracy of the approximation in the area where the optimally controlled process would be driven. Contrary to most problems where we can find a good balance between exploration and exploitation, in the case of this problem multiple runs of the algorithm are needed as both goals are difficult to achieve simultaneously. The idea of exploration and exploitation when multiple runs of the algorithm are available does no longer influences the approximation of value function on a functional approximation level, but rather, in a more subtle way, through our use of the information gathered during previous runs of the algorithm. We use a training measure $\mu_n = \mathcal{N}(b_n, \sigma)$, where σ controls the trade off between exploration and exploitation at the functional approximation level, and b_n is computed as an average of the process controlled by the policy estimated at the previous iteration and accounts for our exploitation of the knowledge gathered during the exploration undertaken at previous iterations.

7.2.4 Value vs. Performance

Our results are displayed in Figures 7.4 and 7.5. From the top panel in figure 7.4 we can observe how the two algorithms approximate the value function during the backward procedure; as anticipated in section 5.1 Performance Iteration, contrary

to Value Iteration, can adapt the shape of the approximation even without receiving external inputs between one door and the other. More schematically, this can be observed from the time-dependent position of the minima of the two estimated value functions displayed in the bottom panel. Figure 7.5, on the other hand, displays the control maps in the top panel, and the distribution of the pathwise performance of the two policies, in the bottom one. From the comparison of the control maps, we immediately see the consequences of the different estimations observed in Figure 7.4. From the distribution of the pathwise-performance, we can conclude that the policy estimated by Performance Iteration is of higher quality than the Value Iteration one. It is also possible to understand the behaviour of the two: Performance Iteration invests more on the control, avoiding frequent impacts with the wall, while Value Iteration saves on control (the first mass of probability is at lower values than Performance Iteration), but experiences more impacts with the wall, causing a much higher average cost.

7.3 A model of interbank systemic risk with partial observation

In this section, we discuss a problem originally formulated in continuous time and in a partial information framework. Even though we focus only on the discrete counterpart, obtained as described in Balata et al. [5], we provide a short description of the original problem in order to help understanding its financial interpretation.

7.3.1 The model

We consider the following model of systemic risk inspired by Carmona et al. [18]. The log-monetary reserves of $\mathcal{B}$ banks lending to and borrowing from each other are

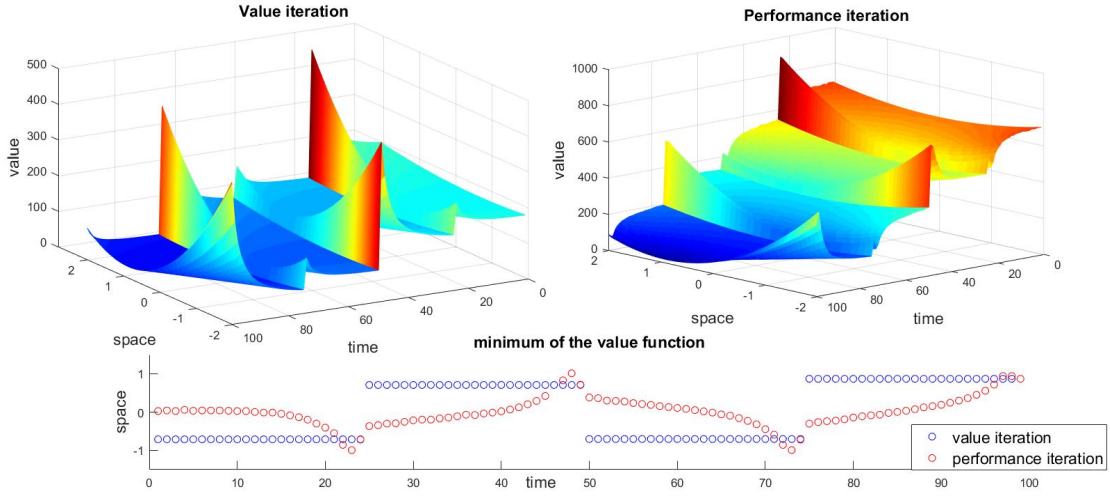


Figure 7.4: On the top the value function estimated during the backward procedure, on the left we can observe that Value Iteration produces an estimate with lower error (≈ 300 rather than ≈ 800), however, on the right, Performance Iteration seems to carry more information about the optimal policy. Such claim is confirmed in the panel on the bottom where we display the minimum of the two value functions at any time step, notice how Performance Iteration captures a more responsive shape, compared to Value Iteration.

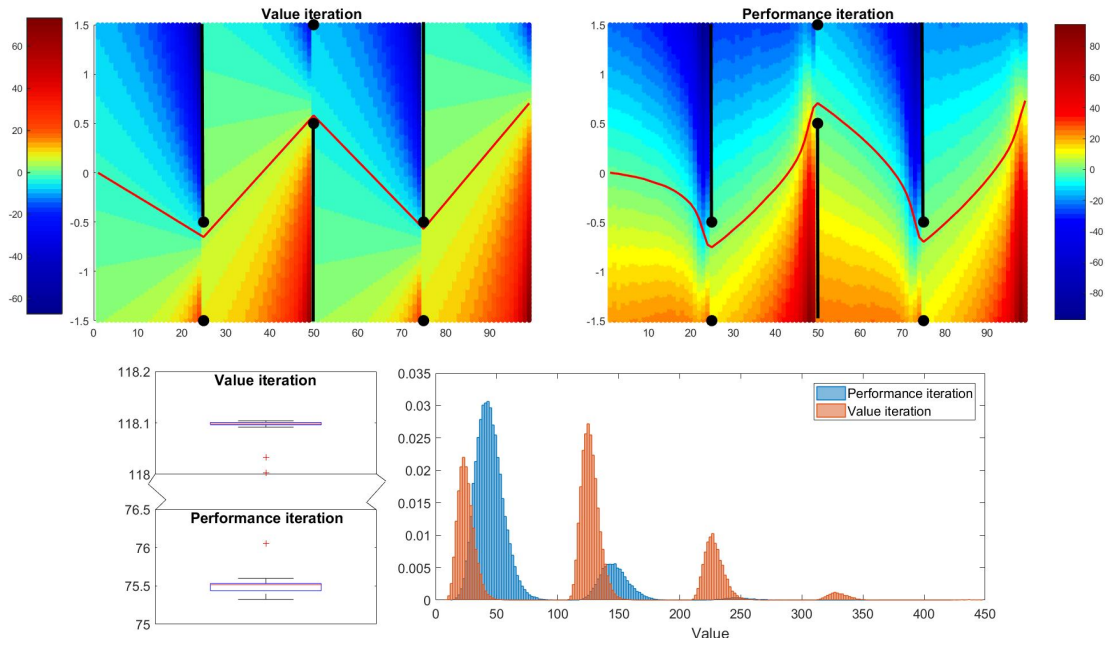


Figure 7.5: The two panels at the top display the control policy estimated by the two algorithms; Value Iteration on the left, Performance Iteration on the right. Notice, at first glance, that the policy estimated by Performance Iteration seems to be more flexible than the one produced by Value Iteration. In the bottom panel, we display the empirical density function of the performance of the policies estimated by the two algorithms. It can be noticed that controlling the robot with the Value Iteration policy leads to more impacts but lower use of the control (the first and second masses of probability are at lower values than Performance Iteration) indicating that Value Iteration estimated a myopic policy.

governed by the system

$$dX_t^i = \frac{\kappa}{\mathcal{B}} \sum_{j=1}^{\mathcal{B}} (X_t^j - X_t^i)dt + \sigma X_t^i(\sqrt{1-\rho^2}dW_t^i + \rho dW_t^0), \quad i = 1, \dots, N;$$

where W^i, $i = 1, \dots, \mathcal{B}$, are independent Brownian motions that represent the idiosyncratic risk of each bank; W^0 is a common noise independent of W^i that represent the systemic risk; the volatility $\sigma > 0$ is a given real parameter; $\rho \in [-1, 1]$ represents the level of concentration of the banks and their exposure to the systemic risk. X_0^i, $i = 1, ..., \mathcal{B}$ are i.i.d. r.v. that represent the initial level of reserves. The mean-reversion coefficient $\kappa > 0$ models the strength of interaction between the banks, where bank i can lend to and borrow from bank j with an amount proportional to the difference between their reserves.

Let us now consider a central bank, viewed as a social planner, who only observes the common noise and not the reserves of each bank, and can influence the strength of the interaction between the individual banks, through the control process u_t. In the asymptotic regime when $\mathcal{B} \to \infty$, the theory of propagation of chaos implies that the reserves X^i of individual banks become independent and identically distributed, conditionally on the common noise W^0. The reserve of the representative bank in the asymptotic regime is then driven by

$$dX_t = (\kappa + u_t)(\mathbb{E}[X_t|W^0] - X_t)dt + \sigma X_t(\sqrt{1-\rho^2}dB_t + \rho dW_t^0), \quad X_0 \sim X_0^1,$$

for some Brownian motion B independent of W^0. The objective of the central bank is to minimize the departure of the reserve of a representative bank from the average level along with the spending on stimulus and compliance cost for institutions when stringent regulation applied, here assumed to be quadratic with respect to its effectiveness. In formulas, the cost for the central bank to implement a policy $(u_t)_{t \geq n}$ is

$$J(n, (X_s, u_s)_{s \geq n}) = \int_n^T \left(\frac{1}{2}u_t^2 + \frac{\eta}{2}(X_t - \mathbb{E}[X_t|W^0])^2 \right) dt + \frac{c}{2}(X_T - \mathbb{E}[X_T|W^0])^2,$$

where $\eta > 0$ and $c > 0$.

The problem described so far is a McKean Vlasov control problem under partial observation, a problem that falls outside the scope of this thesis and range of applicability of Regression Monte Carlo methods to the best knowledge of the author. Notice, however, that in this particular case the problem can be embedded into standard control problem, of the kind introduced in chapter 1. We set $\bar{X}_t = \mathbb{E}[X_t|W^0]$ and $Y_t = \mathbb{E}[(X_t - \bar{X}_t)^2|W^0]$. The cost functional is then written as

$$J(n, (\bar{X}_s, Y_s, u_s)_{s\geq n}) = \int_n^T \left(\frac{1}{2}u_t^2 + \frac{\eta}{2}Y_t\right) dt + \frac{c}{2}Y_T, \tag{7.3.2}$$

where the dynamics of $\bar{X}$ and Y are governed by

$$\begin{aligned} d\bar{X}_t &= \sigma\rho\bar{X}_t dW_t^0, \quad \bar{X}_0 = x_0 = \mathbb{E}[X_0] \\ dY_t &= \Big[\Big(\sigma^2 - 2(\kappa + u_t)\Big)Y_t + \sigma^2(1-\rho^2)\bar{X}_t^2\Big]dt + 2\rho\sigma Y_t dW_t^0, \quad y_0 = \mathbb{V}(X_0). \end{aligned} \tag{7.3.3}$$

We have then reduced the problem to a control problem in dimension two with state variables $(\bar{X}, Y)$.

In order to solve this problem numerically, we write a discrete approximation of (7.3.3). Taking $N = 100$ time steps on a uniform grid we obtain:

$$\begin{aligned} \bar{X}_{n+1} &= \bar{X}_n(1 + \frac{\sigma\rho}{10}\xi_n), \quad \bar{X}_0 = x_0 = \mathbb{E}[X_0] \\ Y_{n+1} &= Y_n + \Big[\Big(\sigma^2 - 2(\kappa + u_n)\Big)Y_n + \sigma^2(1-\rho^2)\bar{X}_n^2\Big]\Delta t + 2\rho\sigma Y_n\sqrt{\Delta t}\xi_n, \\ Y_0 &= y_0 = \mathbb{V}(X_0), \end{aligned} \tag{7.3.4}$$

and define the value function as:

$$V(n, x, y) = \min_{u\in\mathcal{U}_{\{n:n\}}} \left\{\frac{1}{200}u_n^2 + \frac{\eta}{200}Y_n + \mathbb{E}_{n,x,y,u}\Big[V(n+1, X_{n+1}, Y_{n+1})\Big]\right\}$$

7.3.2 Numerical implementation

For the implementation of the RL algorithm, we decided to use polynomial basis functions up to degree 2. This choice allows us to compute the optimal control

analytically as a function of the regression coefficients thereby gaining speed and precision as discussed in remark 5.2.3. For CR, we used basis functions up to degree 3 in all dimensions to obtain more stable results.

The training measure in RL was initially chosen to be a normal on each dimension, however, as we know that the dimension Y represents the conditional variance of the original process X, we centred μ at zero and considered only training points $Y_n^m \geq 0$. In CR, on the other hand, we would need to carefully choose the distribution of the random control, so that the process Y does not become negative. Notice in fact that the Euler approximation, contrary to the original SDE describing Y, is not bounded to be positive and we would, therefore, need to carefully choose a control that keeps Y positive. To simplify the implementation, we modified the Euler approximation of (7.3.3) to feature a reflexive boundary at zero. Such a feature allows training the estimated control policy to not overshoot when trying to drive the process Y to zero, without having Y to become negative.

7.3.3 Comparison

For this problem, in the absence of an analytical solution, we decided to compare the estimations of the value function at time 0 provided by our algorithms with a numerical approximation of the continuous time solution, based on finite difference scheme computed with the software Mathematica.

We computed $\hat{V}(0, x_0 = 10, i_0 = 0)$ using RL and CR methods by considering a sample of size 500 000, and using the following parameters $T = 1$, $\sigma = 0.1$, $\kappa = 0.5$ and $X_0 = 10$.

In Table 7.2 we display the numerical results of two experiments: we took $\eta = 10$, $c = 100$ and $\eta = 100$, $\rho = 0.5$ and vary the value of ρ in the first case, and vary the value of c in the second one.

ρ	RLMC	CR	Bench
0.1	8.88	9.12	8.94
0.2	8.73	8.98	8.77
0.3	8.42	8.69	8.48
0.4	8.02	8.25	8.06
0.5	7.61	7.73	7.51
0.6	6.93	6.97	6.79
0.7	5.94	6.07	5.87
0.8	4.86	4.82	4.67

$c = 100$ and $\eta = 10$.

c	RLMC	CR	Bench
0	7.79	7.78	7.79
1	7.88	7.87	7.88
5	8.22	8.23	8.23
10	8.63	8.64	8.62
25	9.69	9.76	9.62
50	11.08	11.27	10.97

$\rho = 0.5$ and $\eta = 100$.

Table 7.2: Results for the systemic risk problem. Estimations of the value function at the point ($x_0 = 10$) at time 0 provided by different strategies. We took $T = 1$, $N = 100$, $\sigma = 0.1$, $\kappa = 0.5$, $X_0 = 10$.

Figure 7.6 shows two examples of a realisation of the randomness; in each we have a path of the original process $(X_n)_{n=1}^N$ controlled by RLMC (curve "RLMC"), a path naively controlled by $u = 0$ (curve "uncontrolled"), and finally a path of the conditional expectation $(\bar{X}_n)_{n=1}^N$ (curve "$E(X|W)$"). One can see in these two examples that the estimated control is as follows: do nothing when the terminal time is far, i.e., take $u = 0$ to avoid running costs, then catch $\bar{X}$ when the terminal time is approaching, to minimize the terminal cost.

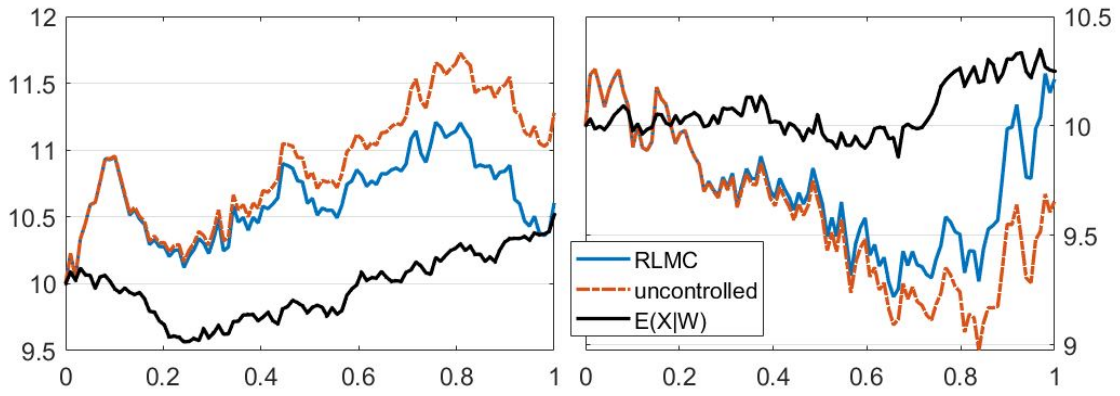


Figure 7.6: Two examples of a realisation of the randomness; in each we have a path of the original process $(X_n)_{n=1}^N$ controlled by RLMC (curve "RLMC"), a path naively controlled by $u = 0$ (curve "uncontrolled"), and finally a path of the conditional expectation $(\bar{X}_n)_{n=1}^N$ (curve "$E(X|W)$").

Part III

Sustainable Energy Systems

Introduction

In this final part of the Thesis we present a study of sustainable energy systems and, in particular, microgrids. Microgrids are self-sustainable systems that aim to increase the penetration of renewable energy generation without affecting the reliability of the power supply. In chapter 8, we present a review of the current energy systems and introduce the reader to some of its most important components. Particular emphasis is given to energy storage, a device that allows to smooth out the random fluctuations in power generation caused by renewable generators. In chapter 9 we introduce a mathematical model on which we base our study of microgrid systems, then presented in chapter 10.

Energy systems have been a central topic during my Ph.D. and have motivated the development of the algorithms presented in Part I. As it will be discussed more in depth in chapter 8, there is a compelling necessity for our society to transition from fossil fuel-based energy supply to renewable generation and sustainability. This transition, however, requires substantial investments, which can be justified only by applying the latest techniques in modelling and optimisation, to study the best design and management practices.

The content of this Part of the Thesis has been published in two papers [1, 6] and presented at international conferences and meetings. The first part of this project has also inspired a TEDx talk, as well as a presentation to MPs at the Parliament, during STEM for Britain 2018.

The main contribution to the literature contained in this part involves the stochastic modelling and optimisation of a microgrid system, both under an almost sure no-blackout constraint as well as a probabilistic one. We also provide a sensitivity analysis of the optimal policy, and the cost, with respect to the main factors influencing the functioning of the power network.

Chapter 8

The current energy landscape

8.1 Introduction

It is nowadays common understanding that the practices of generation and consumption of energy embraced so far were, and still are, not sustainable, causing the environment where we live to progressively become less suitable for life. Raises in average temperatures, draughts and poor air quality, threaten our food supply as well as our own settlements (see sea rise and desertification). On the other hand, many countries are experiencing a long-awaited process of industrialisation, which western economies have gone through one century ago, and are faced by a trade-off between development and sustainability.

We claim that development and sustainability do not represent a trade-off, but rather both objectives can be achieved through a well-engineered set of incentives, pilot projects and international regulatory environment. When a country, a city, or a village, plans for the development of their energy system, they should account for the externalities generated by such designs. Future environmental and healthcare costs of most fossil fuel-based energy generation are often not priced in the assessment of

the optimal energy mix, causing to underweight the investment in renewable forms of energy production.

Even though the costs of renewable generators has decreased considerably over time, thanks to the scale of adoption reached by solar panels and wind turbines, their full integration in energy system is hindered by their unpredictable and uncontrollable nature. To compensate for times when the demand for electricity is high while supply is low, grid operators have to maintain dispatchable generation capacity online at all times. The lack of reliability reduces the economic value of renewable projects motivating the research and development of managing techniques for such plants, that should take full advantage of the zero marginal cost of generation.

At a high level, a straight forward solution to the dispatchability problem is to shift power generated at one time, to some time in the future when it is needed. Different technologies nowadays allow for such load shifting, but they are all very costly, calling again for more efficient use of the available resources and a careful evaluation of their economic potential over time.

The rest of the chapter is organised as follow: in section 8.2 we provide an overview of different renewable energy sources, while in section 8.3 we discuss different energy storage devices. We conclude this introductory chapter with an overview of the challenges that are involved in the energy transition and possible solutions.

8.2 Renewable generation

In this section, we introduce some of the most common technologies used to harvest the renewable energy available in our environment. We will focus in particular on the two most iconic ones: wind and solar.

8.2.1 Wind

Wind power represents the first example of sustainable energy source harnessed by humankind. Initially used to produce mechanical energy in windmills to ground grains and pump water, and in sail-boats to induce propulsion, it is now employed in generators to rotate magnets within a magnetic field, thereby inducing an electric current. The first known installation of a wind turbine to generate electricity dates back to 1887, but it wasn't until 1980 that commercial solutions started to appear in the US and Europe.

In spite of the relatively recent adoption of this technology, the technological progress has been outstanding, with both installed capacity and cost improving exponentially. The latter for example, measured as EUR per KW of power, has dropped by 95%, from approximately 55EUR in 1980 (adjusted for inflation) to 2.5EUR today.

The main commercial advantage of wind power, and renewables in general, is the zero-marginal-cost of electricity produced. The possibility of generating energy at no additional cost (except the wearing off over time) comes with the lack of control over the power produced at any given time, that depends on the wind speed. Wind speed is relatively constant, year on year, but it is subject to severe fluctuations in the short term, preventing a complete transition to a wind-powered economy.

A branch of research to which considerable effort has been devoted is the modelling of the power produced by a wind turbine. Such stochastic models rely on a precise description of the wind speed, which is then converted in electricity generated through a deterministic formula. For more insight on wind power modelling and forecasting refer to Foley et al. [30]. We present a model of wind power generation in section 10.5.1.

8.2.2 Solar

Solar energy, under certain points of view, should be regarded as the only energy source available. Solar radiations are used by plants to generate the energy that sustains them and, through a process of millennia, accumulate and transform into fossil fuels through the dead bodies of plants and animals. The direct extraction of solar energy and conversion into electricity by mankind is relatively recent and dates back to 1880s, with a widespread use starting from 1990s.

Two main technologies are used to convert solar radiation into electrical energy: photovoltaic and solar concentration. While the former directly converts solar radiation into electric current exploiting the photovoltaic effect in small and modular silicon-based cells, the latter functions similarly to all thermal power plants, where some fuels are used to heat up water and generate steam that in turn rotates some turbines. In the case of solar concentration, a wide array of mirrors or lens are used to concentrate sunlight into a beam that heats up water, or other substances.

The widespread use of solar panels can be explained by the strong incentives that have been offered by governments around the world to promote their adoption, as well as their distributed nature. Even though some small scale solutions are available for harvesting wind power, it is much more common to install solar panels on rooftops, which do not require maintenance and do not have moving parts, guaranteeing a seemingly unnoticeable electricity generation.

8.2.3 Other sources

Nowadays, the portfolio of renewables in use is highly concentrated into hydroelectric, wind and solar. Promising technologies, on the other hand, are not being promoted as much as they need to achieve a more balanced range of generating

units. Geothermal energy, ocean thermal, tidal and wave energy, residual heat from industry, as well as new designs for harvesting solar and wind energy, can provide, along with more developed renewables, a mix of sustainable energy sources that complement one another. A mixed pool of energy sources allows to smooth out the overall generation power, making it simpler for the grid operator to match the fluctuations in demand. Further discussion on alternative energy generation is, however, outside the scope of this thesis.

8.3 Energy storage

Energy storages can take many different forms. In the widest sense, even fossil fuel can be seen as a medium to store solar energy released in a far past. In this discussion we will, however, focus on storage facilities intended to transfer an amount of energy, readily available today, to a given point in the future, might this be for few minutes (flywheels) or months/years (hydro-reservoirs/gas caves). In particular, the recent trend in renewable electricity generation calls for an increasing study of management and design of electricity storages. Differently from other forms of energy, electricity can not be stored directly, but needs to be transformed: potential energy when stored in hydro-reservoirs, kinetic energy when stored in flywheels and via increasing pressure when stored in compressed air/gas storages, electrochemical energy when stored in batteries, etc.

As mentioned in the introduction, the importance of storage devices in the future design of the electric grid cannot be understated. In order to accommodate an increasing share of renewable generation, characterised by an intermittent and not easily forecastable generation output, we need a buffer that allows shifting power from times and places with high generation to times and places with high demand. This buffer can either be provided by extending transmission lines and thermal

generators to be used as reserves, or by increasing the total capacity of grid-connected electricity storages. In particular, in the case of microgrids described in chapter 9, which are attracting increasing interest from both industry and academia, storage devices are fundamental in order to guarantee the possibility of running in islanded mode.

Anuta et al. [2] claim that widespread use of electricity storage has been limited by high costs, lack of deployment experience and the barriers created by the regulatory framework. However, in the UK alone, the benefits that extensive use of storage could bring are estimated to about 10 billion pounds by 2050. Today, 99% of the electricity storage is represented by pumped hydroelectric storage (PHS) with a total power of 150 GW worldwide. Anuta et al. [2] claim also that storage is often uneconomical if used for a single application and therefore there is need to develop a viable business model to allow the investors to exploit the potential services that storage can provide (black start, power quality, reserves, voltage regulation, frequency regulation, renewable smoothing dispatch, increase in asset utilisation, peak shaving, price arbitrage).

An interesting study in Mishra et al. [57] looks at the integration of different electricity storage technologies at different levels of the grid. In particular, the study considers cheap compressed air electricity storages (CAES) and different types of fast reacting batteries to be installed at home, at the distribution level, and at the transmission level. The study found that if just one technology on one level was to be considered then the best choice would have been lithium-ion batteries deployed at the costumers' houses. The optimal configuration, however, includes slow reacting CAES deployed at higher levels. Finally, the author identifies the following as most important characteristic of an electric storage device which determines the optimal location and mode of use: capacity, max power, efficiency, self-discharge rate, life-cycle, power ramp up.

So far, the most studied kind of facilities have been gas storages, because they are widely used around the world and because gas, differently to electricity, is much more easily storable. However, recently, much attention has been given to electricity storages as well, because they represent a very important asset in the future of the electric grid.

8.3.1 Natural gas storage facilities

Differently from pure electric energy, natural gas can be stored more easily, even though it still requires a dedicated infrastructure for transportation and large investments for the storage facilities. So, since it is not a usual commodity, natural gas trading nowadays heavily relies on the storage facilities, both for the winter-summer oscillation in consumption and for the quite inelastic extraction rate of natural gas.

In practice, natural gas is stored in huge natural caves, salt domes, emptied gas or oil fields, and aquifers, even though many small projects have used different technologies and solutions. It should be noted that despite the market for the liquefied natural gas (LNG) is growing, and since after liquefaction the gas can be stored as a traditional commodity, these problems are not of primary interest here.

An interesting problem linked to this kind of facilities, given their particular physical characteristics and the complicated dependence on energy prices, is their economic valuation. Swindle et al. [74] reports that most of the value is in the ability of the gas storage to monetize short time scale mean reversion in gas price. For this reason, the valuation heavily depends on the variance structure of gas Futures (derivatives contracts) prices.

8.3.2 Pumped hydro storage

Hydro-electric power plants are the source of most of the renewable electric energy produced; these facilities use natural basins, like lakes and rivers or a system of artificial dams, to exploit the potential energy of a big amount of water efficiently released into turbines to generate electricity. Some of these plants are formed by different basins located at different heights which are then connected with a system of pipes equipped with turbines for electricity production. Pumps create load and allow to pump the water back to a point with greater potential energy, effectively storing electric energy.

Modelling these facilities can be difficult, the topology of the dams and the system of pipes that connects them, makes every facility around the world unique. In addition, as each basin has to be modelled separately, and the problem can grow in dimensionality relatively quickly. As we discussed in chapter 3, simulation methods for the endogenous (inventory) process are required in order to efficiently solve these problems. In a full discretisation setting, see Felix and Weber [29] for more details on the problem of valuing a multiple hydro reservoirs storage systems.

8.3.3 Batteries

Batteries are devices that store electricity by accumulating and releasing chemical force generated within each of the many cells they are composed by. The widespread adoption of batteries as energy storage in the consumer market can be justified by the reduced size of the units. Under the point of view of power system management, on the other hand, other characteristics have attracted the attention of industry and academia. Batteries offer amongst the highest round-trip efficiency (energy is consumed in the process of storing), along with high power and an amount of energy that allows delivering such rated power for far longer than comparable solutions.

Different technologies have been developed, offering a wide range of specification, and are now adopted for power quality applications (along with flywheels) as well as bridging power and energy management (along with, but at a considerably lower scale, PHS).

The high modularity of batteries, as well as their standardized characteristics, suggest that the modelling of a storage plant should be easier than an hydro-reservoir, as the combination of all modules can be represented as one, unique inventory. On the other hand, however, the electrochemical system of storage has its own physics which involves different and challenging difficulties.

One of the most important constraints introduced by the battery is the wearing off that manifest itself in the form of decreasing the total capacity of the inventory, as a function of the number of complete recharge cycles and operating temperature. It is often reported, [26] [7], that the wearing off, in addition, should not be modelled by some throughput computation (quantity of energy charged and discharged) because much more impact is attributable by the amount of time the battery is used at high power. Mathematically, this means that we can not anymore keep track only of the single information on the state of charge I_n, rather, we will need to know the power injection or withdraw at any time in the past in order to take into account the trade-off between immediate benefit and shorter lifetime.

It should be noticed here that many studies on electricity storage in the framework of dynamic programming are technology agnostic and do not specify what kind of facility they are considering, effectively working at a high level of abstraction. However, in the following, we present some examples of realistic modeling of a battery; note that these studies have been completed in collaboration with EDF R&D and somehow represent a good insight into the state of the art in industry. In particular, we notice the lack of interest in the price arbitrage use of the battery and an interest in the ancillary services these devices can offer: Haessig et al. [37]

studies aging limitation in order to find control strategies that enhance the life of the battery; it does so effectively by creating a trade-off between profits and wearing off of the battery. The authors model the aging process as an Ah-throughput converted in an equivalent number of cycles. In other words, the authors assume that the battery has a finite amount of energy which can flow through it before it comes at the end of its lifetime. Defining then an expected calendar life for the device, the average power the battery should use, at any time in order to live for the requested amount of time, can be computed. The tradeoff, then, is created by defining a parameter for a short term horizon which indicates from how far in the future you can "borrow" excess power so that, in the end, in the short term horizon the average planned wearing off is preserved. Haessig et al. [36] present an aging-aware model of a NaS battery used to assist a wind energy producer in meeting her committed generation, on which she has agreed based on a short time horizon forecast on wind speed. The paper addresses the problem of modelling the battery by means of its modularity describing it as a particular assembly of basic energy cells.

8.3.4 Flexible alternatives

One of the most common flexible storage solution is the so-called CHP (combined heat and power) which is a device that receives as input gas and electricity and provides in exchange power and heat. The device is provided with thermal storage which can be charged at the most suitable times. Kitapbayev et al. [47], for example, studies CHP for district heating. In particular, they use a risk-sensitive approach where the risk aversion parameter is tuned in order to obtain a certain required CVaR, or expected shortfall, on the costs.

A more interesting, flexible alternative to storage, is the idea of demand response programs and load control which create virtual storage. The former acts on longer time-scales and relies on customer adoption, who can participate in the peak shaving

activity under some compensation scheme. The latter involves automatic short time-scale control of particular appliances in buildings whose regulation of power could drastically reduce the high-frequency fluctuation of the demand without affecting the utility of the users of the building. Meyn et al. [56] shows an application in which the rotational speed of the fans in big buildings can be regulated in order to improve the reliability of the grid. Refrigerators and water pumps in swimming pools could be used similarly, thanks to the inertia that these systems enjoy.

Note that an extensive discussion about energy storage under the technological, engineering and economic point of view can be found in the recent book Huggins [43].

8.4 Challenges and solutions

The integration of additional renewable generation, as mentioned earlier, is hindered by the difficulties in managing the grid when the ratio of classic thermal generation to renewables decreases. In addition to the dispatchability of traditional power plants, which simplifies the scheduling of online plants, renewables also lack power inertia. Power inertia is a phenomenon created by the big masses (magnets) attached to the turbines of power plants, that induces them to keep rotating at a constant angular speed. The frequency of rotation is directly transmitted to the electricity, whose frequency has to remain constant in order to guarantee a safe supply of power. High inertia guarantees that the mismatch between load and supply changes slowly, as the rotational speed of the turbines is influenced by changes in frequency. Renewable sources like solar panels do not have a strong magnetic field that can increase system inertia, and wind turbines, that recently started producing synthetic inertia, are not sufficient to replace thermal power plants, breaking down the coupling with the rest of the electricity grid. In order to address these concerns, it has often been proposed

the idea of creating smaller regions of the grid that are able to manage themselves, and can be connected according to the requirements of the grid operator.

A microgrid is a network of loads and energy generating units, that often include renewable sources like solar panels and wind turbines, alongside more traditional forms of thermal electricity production. These microgrids can be part of the main grid or isolated. Communities in rural areas of the world have long now enjoyed the installation of isolated microgrid systems that provide a reliable and often environment-friendly source of electricity to meet their power needs.

The elementary purpose of a microgrid is to provide a continuous electricity supply from the variable power produced by renewable generators while minimising the installation and running costs. In this kind of systems, the uncertainty of both, the load and the renewable production are high, and its negative effect on the system stability can be mitigated by including a battery to the microgrid. Energy storage devices ensure power quality, including frequency and voltage regulation (see Hayashi et al. [39]) and provide backup power in case of any contingency. A dispatchable unit, in the form of a diesel generator for example, is also used as a backup solution, and to provide base-load power.

In order to enable these services to be developed, a large amount of data must be combined into a smart multi-energy system. Data from local sources (such as smart thermostats and energy meters) are combined with data from forecasts (like weather forecasts for wind and solar farms) and from energy markets where gas, power and other energy forms are traded. A multi-energy market model can reduce the costs of energy generation and distribution for society, as a consequence of more effective coordination with the production of renewable energy. The purpose of the hardware and software components involved is to make energy consumption within those systems more efficient, more flexible and more cost effective. Research and implementation of new technologies is of paramount importance to develop platforms

that can control a fully integrated multi-energy system. We provide an example of such control algorithms in chapter10, where we show how to combine, in the most efficient ways, the different components of a sustainable energy system by applying stochastic optimisation techniques. We present the mathematical modelling of a microgrid in chapter 9.

Chapter 9

Microgrid modelling

9.1 Introduction

In this chapter, we model a microgrid serving a small group of customers in islanded mode, meaning that the network is not connected to the main national grid. The system consists of an intermittent renewable generator unit, a conventional dispatchable generator, a battery storage system and the loads. Both the load and the intermittent renewable production are stochastic, and we use a Markov process to model directly the residual demand, that is, the difference between the load and the renewable production. We then set up a stochastic optimisation problem, whose goal is to minimize the cost of using the dispatchable generator plus the cost of curtailing renewable energy in case of excess production, subject to the constraint of ensuring reliable energy supply. The numerical examples illustrate the performance of the optimal policies, provide insights on the optimal sizing of the battery, and compare the policies obtained by stochastic optimisation to the industry standard, which uses deterministic scheduling/optimisation.

The optimisation problem arising from the search for a cost-effective control strategy

has been extensively studied. Three recent survey papers Liang and Zhuang [50], Olivares et al. [63], Reddy et al. [66] summarize different methods used for optimal usage, expansion and voltage control for the microgrids. Heymann et al. [41, 42] transform the optimisation problem associated with the microgrid management into an optimal control framework and solve it using the corresponding Hamilton Jacobi Bellman equation. Besides proposing an optimal strategy, the authors also compare the solution of the deterministic and stochastic representation of the problem. However, similarly to most PDE methods, this approach suffers from the curse of dimensionality and, as a result, it is difficult to scale. In contrast to existing approaches, the Regression Monte Carlo method used in this chapter is more easily scalable and works well in moderately large dimensions.

Identifying the optimal mix, the size and the placement of different components in the microgrid is an important challenge to its large scale use. The papers Mashayekh et al. [53, 54] use mixed-integer linear programming to address the design problem and test their model on a real data set from a microgrid in Alaska. In a similar work, Olatomiwa et al. [62] studied the economically optimal mix of PV, wind, batteries, and diesel for rural areas in Nigeria. In Haessig et al. [38], optimal battery storage sizing is deduced from the autocorrelation structure of renewable production forecast errors. In this paper, we propose an alternative approach for the optimal sizing of the battery energy storage system, assuming stochastic load dynamics and a fixed lifetime of the battery. Our in-depth analysis of the system behaviour leads to practical guidelines for the design and control of islanded microgrids.

Finally, several authors Collet et al. [22], Ding et al. [24, 25] used stochastic control techniques to determine optimal operation strategies for wind production/storage systems with access to energy markets. In contrast to these papers, in the present study, energy prices appear only as constant penalty factors in the cost functional when an electricity generator is employed, and the main focus is on the stable

operation of the microgrid avoiding blackouts.

The rest of the chapter is organized as follows: In section 9.2 we describe the microgrid model and introduce the different components of the system, in section 9.3 we translate the problem of managing the microgrid into a stochastic optimisation problem and present the dynamic programming equation that we then solve numerically. In section 9.4 we present an extension where the blackout constraint imposed in the original problem is probabilistically relaxed.

9.2 Model description

In this section, we will discuss the topology of the microgrid, its operation, components, and their respective dynamics. Although we discuss a simplified microgrid model, more complicated topologies can be studied using straightforward generalizations of the methods presented in this thesis.

Consider a microgrid serving a small, isolated village. Most of the power supplied to the village is generated by units whose output has zero marginal cost, is intermittent and uncontrolled. Additional power is supplied by a controlled generator whose operations come alongside a cost for the microgrid owner (either the community itself or a power utility). Often the intermittent units include solar panels and wind turbines, while the controlled unit is often a diesel generator. In order to fully exploit the free power generated by the renewable units at times when production exceeds the demand, microgrids are equipped with energy storage devices. These can be represented by a battery energy storage system.

The introduction of the battery in the system not only allows for the inter-temporal transfer of energy from times when demand is low, to times when it is higher but also introduces an element of strategic behavior that can be employed by the system

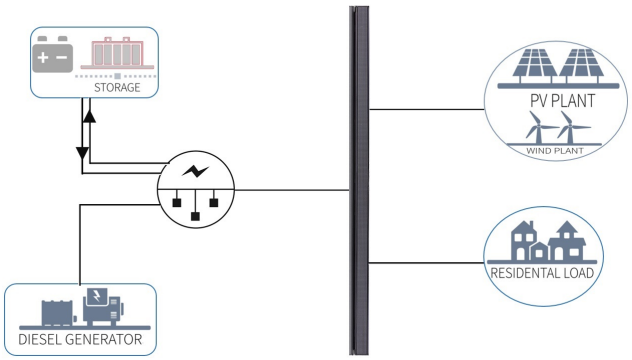


Figure 9.1: The figure above shows an example of microgrid topology that contains all the elements in our model. The network is arranged as follows: photovoltaic panels and wind turbines provide renewable generation, a diesel generator provides dispatchable power for the village and a battery storage system is used to inject or withdraw energy.

controller, to minimize the operational costs. Without energy storage, the diesel generator has to be run at all times when the demand exceeds production. When a battery is installed, intensity and timing of output from the diesel generator can be adjusted to move the level of charge of the battery towards the most cost-effective levels.

In figure 9.1 we propose a schematic description of the system which might help the reader to familiarize themselves with the microgrid layout, whose components are described more in depth in the following sections.

9.2.1 Residual demand

Consider two stochastic processes L_n and R_n, the former represents the demand/load and the latter the production through the renewable generators. Notice that both processes are uncontrolled and they represent, respectively, the unconditional withdrawal or injection of power in the system (constant during time step). For the purpose of managing the microgrid, the controller is interested only in the net effect

of the two processes denoted by the process X_n:

$$X_n = L_n - R_n \; ; \quad t \in \{0, 1, \ldots, N\}. \tag{9.2.1}$$

Remark 9.2.1. *The state variable X_n represents the residual demand of power at each time n, such that for $X_n > 0$, we should provide power through the battery or diesel generator and for $X_n < 0$ we can store the extra power in the battery.*

The process X_n is driven by the following difference equation, starting from an initial point $X_0 = x_0$:

$$X_{n+1} = X_n + b(\Lambda_n - X_n)\Delta t + \sigma_n \sqrt{\Delta t}\, \xi_n \; ; \quad n \in \{0, 1, \ldots, N\} \tag{9.2.2}$$

where $\xi_n \sim \mathcal{N}(0, 1)$, Δt is the amount of time before new information is acquired, b is the mean reversion speed, σ_n the variance of the process and Λ_n is the time-dependent mean reversion level.

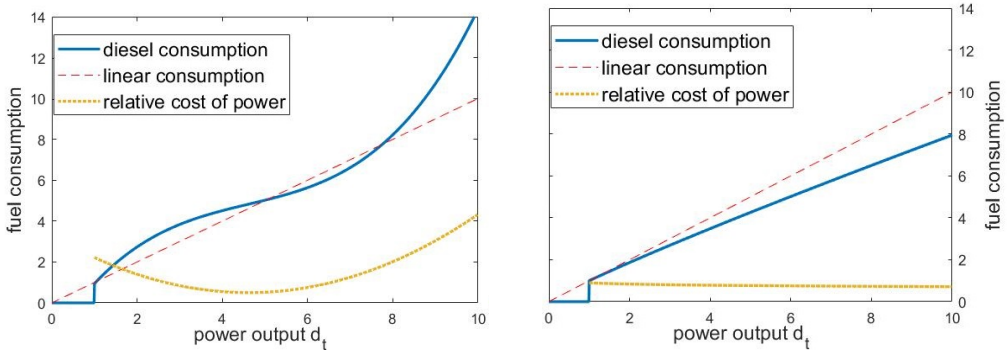
Remark 9.2.2. *In real applications, the function Λ_n should represent the best forecast available for future residual demand at the time of the estimation of the policy.*

9.2.2 Diesel generator

The Diesel generator represents the controlled dispatchable unit. The state of the generator is represented by $m_n = \{0, 1\}$. If $m_n = 0$ then the diesel generator is OFF, while it is ON when $m_n = 1$. When the engine is ON, it produces a power output denoted by $d_n \in [d_{min}, d_{max}]$ at time n, for $d_{min} > 0$.

Notice that, in addition, when the engine is being turned ON, an extra amount of fuel is burned in order for the generator to warm up and reach working regime. We model the cost of burning extra fuel with a switching cost $\mathcal{K}$ that is paid every time

the switch changes from 0 to 1. The fuel consumption of the diesel generator is modelled by an increasing function $\rho(d_n)$ which maps the power d_n produced during one-time step into the quantity of fuel necessary for such output. Denoting by P_n the price of diesel at time n, the cost of producing d_n KW of power for one time step, is $P_n\rho(d_n)$. For simplicity, we take a constant price of the fuel $P_n = p$. Two examples of efficiency functions ρ are described in figure 9.2.



$\rho(d) = \frac{(d-6)^3+6^3+d}{10}$ $\rho(d) = d^{0.9}$

Figure 9.2: The panels above show two examples of efficiency function (litres/KW), on the left $\rho(d) = \frac{(d-6)^3+6^3+d}{10}$, typical of a generator designed to operate at medium regime, on the right $\rho(d) = d^{0.9}$, typical of a generator designed to operate a full capacity.

9.2.3 Dynamics of the battery

The storage device is directly connected to the microgrid and therefore its output is equal to the imbalance between the residual demand X_n and diesel generator output d_n, when this is allowed by the physical constraint. The battery, therefore, is discharged in case of insufficiency of power and charged when the diesel generator and renewables provide a surplus of power.

Notice then that an energy storage has a limited amount of capacity after which it can not be charged further, as well as an "empty" level below which no more power

can be provided from the battery. We denote the state of charge by the controlled process I_n^d which is described by the following equation:

$$I_{n+1}^d = I_n^d - B_n^d \Delta t, \quad n \in \{0, 1, \ldots, N-1\}, \quad I_0^d = w_0 \tag{9.2.3}$$

where $I_n^d \in [0, I_{max}]$, the battery output/input $B_n^d \in [B^{\min}, B^{\max}]$, $B^{\min} < 0$ and $B^{\max} > 0$. For simplicity, we assume that the battery is 100% efficient. Notice that we used superscript d on B^d and I^d to highlight the dependence of these processes on the controlled diesel output d_n.

Let us denote the power output of the battery by B_n^d and its power rating by $B^{\max}$ and $B^{\min}$, where $B^{\max}$ and $B^{\min}$ represent respectively the maximum output and input. Thus:

$$B_n^d = \frac{I_n^d - I_{\max}}{\Delta t} \vee \left(B^{\min} \vee (X_n - d_n) \wedge B^{\max} \right) \wedge \frac{I_n^d}{\Delta t} \tag{9.2.4}$$

The case where $B_n^d < 0$, represents that the battery is charging while the case where $B_n^d > 0$, represents that the battery is supplying power.

Intuition tells us that the bigger the battery, the less diesel will be needed to run the operations of the microgrid. This is true because a bigger battery would allow storing for later, using a bigger proportion of the excess power produced by the renewables. Batteries, however, are very expensive, and the cost per KWh of capacity scales almost linearly for the kind of devices we consider in this paper (parallel connection of smaller batteries), hence it is important to find the optimal size of battery for the needs of each specific microgrid.

9.2.4 Management of the microgrid

The purpose of the microgrid is to provide a cheap and reliable source of power supply to, at least, match the demand. Therefore, we search for a control policy

for the diesel generator which minimizes the operating cost and produces enough electricity to match the residual demand. In order to assess how well we are doing in supplying electricity, we introduce the controlled imbalance process S_n defined as follows:

$$S_n = X_n - B_n^d - d_n \quad n \in [0, N] \tag{9.2.5}$$

Ideally, the owner of the microgrid would like to have $S_n = 0 \quad \forall\, t$. This situation represents the perfect balance of demand and generation. When $S_n > 0$ we observe a *blackout*, residual demand is greater than the production meaning that some loads are automatically disconnected from the system. The situation $S_n < 0$ is defined as a curtailment of renewable resources and takes place when we have a surplus of electricity and the battery is full.

We treat the two scenarios, blackout and curtailment asymmetrically. To ensure no-blackout, $S_n \leq 0$, and regular supply of power, we impose a constraint on the set of admissible controls:

$$\begin{aligned} S_n &\leq 0 \\ \text{i.e. } d_n &\geq X_n - B_n^d. \end{aligned} \tag{9.2.6}$$

However, for $S_n < 0$ i.e. surplus of electricity, we penalize the microgrid using a proportional cost denoted by C. A large penalty would lead to low level of curtailment and can be thought of as a parameter in the subsequent optimisation problem. In section 9.4 and 10.4, we also consider a more realistic case where we impose a constraint on the probability of blackout, rather than the hard constraint just introduced.

9.3 Stochastic optimisation problem

We state now the stochastic control problem for the diesel generator operating in a microgrid system as described in section 9.2. In practice, we seek a control that

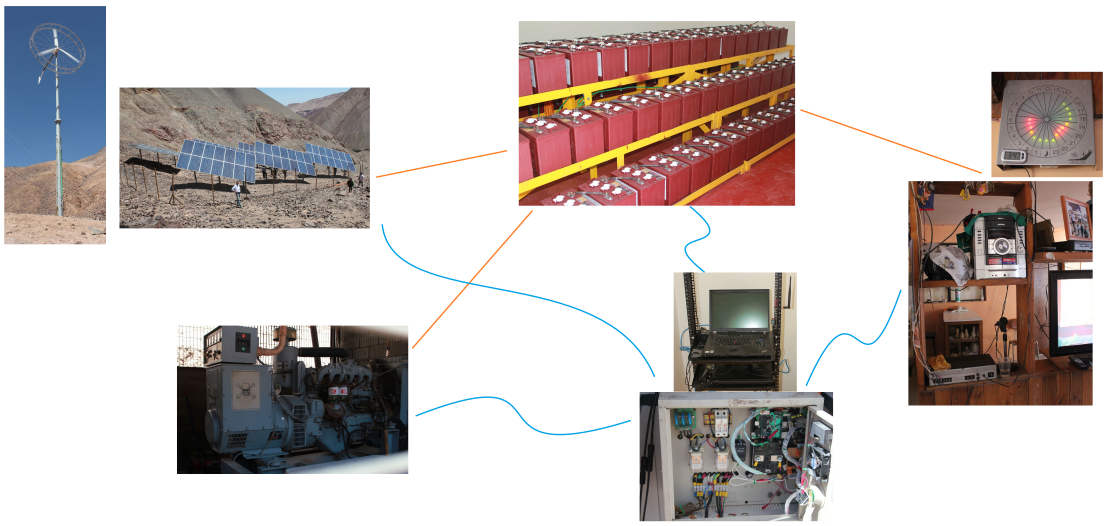


Figure 9.3: The figure above is a pictorial description of the layout of the microgrid installed in Huatacondo, Atacama desert, Chile. The picture has been obtained by pasting together photos taken during one of the field trips of the team following the project. Credit to Chris Marnay, Berkeley National Laboratory, for the photos.

minimizes the cost of diesel usage $p\rho(d)$, the switching cost $\mathcal{K}$ and the curtailment cost $C|S_n|\mathbb{1}_{\{S_n<0\}}$, under the no black-out constraint $S_n \leq 0$.

Let us define the pathwise value J, given by

$$J(n,(X_t,I_t,m_t;d_t)_{t=n}^N) = \sum_{s=n}^{N-1} \mathbb{1}_{\{m_{s+1}-m_s=1\}}\mathcal{K} + \pi\rho(d_s) + C|S_s|\mathbb{1}_{\{S_s<0\}} + g(I_N^d). \tag{9.3.7}$$

As a consequence, we define the value function as:

$$V(n,x,w,m) = \min_{(d_u)_{u=n}^N} \left\{ \mathbb{E}\left[J(n,(X_t,I_t,m_t;d_t)_{t=n}^N) \middle| X_n = x, I_n^d = w, m_n = m \right] \right\} \tag{9.3.8}$$

$$\begin{aligned}
\text{subject to} \quad & d_n \geq X_n - B_n^d \quad \forall t; && \text{(9.3.9a)}\\
& d_n \in [d_{min}, d_{max}] \cup \{0\}; && \text{(9.3.9b)}\\
& B_n^d = \frac{I_n^d - I_{\max}}{\Delta t} \vee \Big(B^{\min} \vee (X_n - d_n) \wedge B^{\max}\Big) \wedge \frac{I_n^d}{\Delta t}; && \text{(9.3.9c)}\\
& m_n = \mathbb{1}_{d_n \neq 0}; && \text{(9.3.9d)}
\end{aligned}$$

where (9.3.9a) represents the black-out constraints, expressed in terms of the power produced by the diesel generator, (9.3.9b) represents the minimum and maximum power output of the generator, (9.3.9c) models the physical constraints of the battery: maximum input/output power and maximum capacity and (9.3.9d) encodes the state of the diesel generator.

From equation (9.3.8), we can write the associated dynamic programming formulation which helps understand the structure of the problem composed of two optimal control sub-problems: an optimal switching problem between being in the regime ON or OFF, and another absolutely continuous control problem assuming the regime is ON. The equation reads as follows:

$$\begin{aligned}
V(n,x,w,0) = \min_{d \in \mathcal{U}_t} \Big\{ & \pi\rho(0) + C|S_n|\mathbb{1}_{\{S_n<0\}} + \mathcal{C}(n,x,w,0;d), \\
& \min_{d \in \mathcal{U}_t} \Big(\mathcal{K} + \pi\rho(d) + C|S_n|\mathbb{1}_{\{S_n<0\}} + \mathcal{C}(n,x,w,1;d) \Big) \Big\}, \\
V(n,x,w,1) = \min_{d \in \mathcal{U}_t} \Big\{ & \pi\rho(0) + C|S_n|\mathbb{1}_{\{S_n<0\}} + \mathcal{C}(n,x,w,0;d), \\
& \min_{d \in \mathcal{U}_t} \Big(\pi\rho(d) + C|S_n|\mathbb{1}_{\{S_n<0\}} + \mathcal{C}(n,x,w,1;d) \Big) \Big\}
\end{aligned} \tag{9.3.10}$$

where

$$\mathcal{C}(n,x,w,m;d) = \mathbb{E}[V(n+1, X_{n+1}, I_{n+1}, m_{t+1}) | X_n = x, I_n = w, d_n = d, m_n = m],$$

is the conditional expectation of the future costs and $\mathcal{U}_n$ is the collection of admissible

controls d at each time step t, i.e.

$$\mathcal{U}_n := \{d_n : \text{equations (9.3.9a) - (9.3.9c) are satisfied and } d_n \text{ is adapted to } \mathcal{F}_n\}. \tag{9.3.11}$$

In order to ensure that the set of admissible controls is nonempty we introduce the following assumption:

Assumption 3. *The diesel generator is powerful enough to supply demand at all times, i.e there is always a control d that satisfies the blackout constraint.*

Remark 9.3.1. *We enforce assumption 3 by redefining the residual demand process with a truncated version of* (9.2.1)*, such that $\tilde{X}_n = \min(X_n, X_{\max})$ is the residual demand. In practice, this is reasonable because the maximum power that could be required from the microgrid is known apriori and the diesel generator is generally sized to the maximum capacity installed on the system. For the sake of notational simplicity, we will drop the tilde on the variable $\tilde{X}_n$ from the following sections.*

Note that (9.3.10) provides a direct technique to solve problem (9.3.8), iterating backward in time from a known terminal condition and solving a static, one period, optimisation problem at each time step. The only difficulty in this procedure lies in the estimation of conditional expectations of the future value function, which can not be computed exactly. Refer to chapter 2-3 for a presentation of numerical methods suitable to solve this kind of problems.

9.4 Relaxing the no-blackout constraint

In this section, we introduce an extension of the original model where we relax the hard constraint on blackouts and substitute it with one on the probability of a blackout event between decision times. Recall that the control decisions are made

at discrete epochs $\{0, 1, \ldots, N\}$, however, in this problem, we assume that these decisions affect the state of the system continuously. As a result, choosing the control d_s involves minimising the cost of running the microgrid, as well as controlling the probability of blackout (i.e. the controller fails to match the residual demand) at intermediate intervals $[n, n+1)$. This section presents an introduction to the problem originally included in Balata et al. [6].

The stochastic control problem in this model reads:

$$\begin{aligned} V(n,x,i,m) &= \min_{(d_s)_{s=n}^{N}} \left\{ \mathbb{E}\left[\sum_{s=n}^{N-1} \mathbb{1}_{\{m_{s+1}-m_s=1\}} \mathcal{K} + \pi\rho(d_s) + g(I_N^d) \right] \right\} \\ &\text{subject to } \mathbb{P}(\sup_{s\in[n,n+1)} S_s > 0 | \mathcal{F}_n) < p \quad \forall\, n. \end{aligned} \tag{9.4.12}$$

The admissibility condition is in terms of the supremum functional

$$p_n(x,i,d_n) := \mathbb{P}(\sup_{s\in[n,n+1)} S_s > 0 | \mathcal{F}_n).$$

Because $p_n()$ is not (in general) available analytically, the admissibility condition $p_n < p$ is implicit.

Define the admissible control set $\mathcal{U}(x,i) = \{d \in \{0\} \cup [d_{\min}, d_{\max}] : p_n(X_n, d_n) < p \text{ and } I_{n+1} \in [0, I_{\max}]\}$.

Remark 9.4.1. *The admissible set $\mathcal{U}$ for this problem has the special structure of being an interval: if $d \in \mathcal{U}_n(x,i)$, then $\forall\ \tilde{d} > d, \tilde{d} \in \mathcal{U}_n(x,i)$. Hence, we may represent $\mathcal{U}(x,i) = [d^{\min}(x,i), d_{\max}]$ in terms of the minimal admissible diesel output $d^{min}(x,i)$.*

In Figure 9.4 we present the minimum admissible control $d^{\min}(x)$ under a constraint of $p = 0.01$ probability of blackout, conditional on (X, I). We also present a path for $(X_n, I_n)_{t\geq 0}$ using a *myopic* strategy where the controller employs the minimum admissible control at each point, $d_n := d_{min}(X_n, I_n)\ \forall n$. Notice how for the most part, $d^{\min}(x,i) = 0$ is trivially admissible so that $\mathcal{U}(x,i) = [0, \bar{d}]$ and the blackout

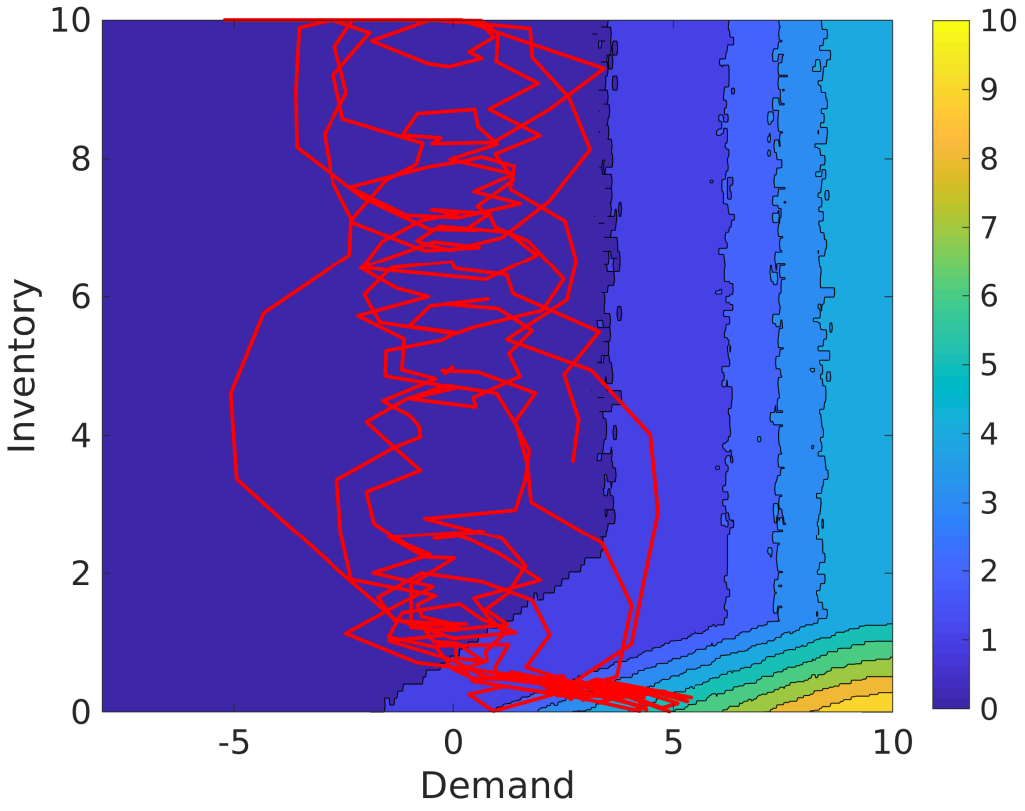


Figure 9.4: Contour plot for minimum admissible diesel output. Red line represents a path following a myopic strategy choosing the minimum admissible control.

constraint is not binding. This is not surprising, as blackouts are only possible when X_n is large and positive and the battery is close to empty, $I_n \simeq 0$. Thus, except for the lower-right corner, any control is admissible. As a result, only a small subset of the state space actually requires additional effort to estimate the admissible set $\mathcal{U}(x, i)$.

Chapter 10

Guidelines for optimal microgrid management

10.1 Introduction

In this chapter, we illustrate the results of the numerical computations and we analyse the behaviour of the model of microgrid that we have outlined in the previous chapter. Through our methodology, expressed by the different experiments and optimisation procedures devised, we seek to guide the practitioner towards a full analysis of a real microgrid system. Our work tries to introduce novel simulations techniques that can increase the efficiency of a system that is part of the critical transition towards sustainable energy management.

Before exploring the optimisation results, we provide a list of the base parameters chosen for the numerical experiments; notice that the ”s” column indicates whether a sensitivity analysis is run for such a parameter. For the meaning of the parameters refer to section 9.2.

parameter	value	s
T	$100h$	
Δt	$0.25h$	
b	0.5	*
σ	2	*
Λ_t	0, $\forall t$	

parameter	value	s
I_{max}	10 KWh	*
$\rho(d)$	$\frac{(d-d^*)^3+(d^*)^3+d}{10}\frac{litre}{\text{KW}}$	
d^*	6 KW	
p	1 €	
$g(i)$	0, $\forall i$	

parameter	value	s
d_{min}	1KW	
d_{max}	10KW	
K	5 €	*
C	0 €	*

According to the parameters table above, and recalling remark 9.3.1, the residual demand has the following dynamics:

$$X_{n+1} = \left(X_n(1 - 0.5\Delta t) + \sigma\sqrt{\Delta t}\xi_n\right) \wedge 10, \quad t \in \{0, 1, \ldots, T-1\}, \tag{10.1.1}$$

where $\xi_t \sim \mathcal{N}(0, 1)$. We decided to use such simple dynamics for illustrative purposes in order to make the sensitivity of the optimal control policy to the remaining parameters more straightforward to understand.

The rest of the chapter is organised as follow: in section 10.2 we study the optimal sizing of the battery, the impact of increasing penetration of renewable generation and the effect of switching and curtailment cost. In section 10.3 we compare the performance of the policy estimated in the stochastic setting versus an industry-standard policy obtained from a deterministic model and show the superior performance of the former, in particular during those days when the system is more critically overloaded. We progress with section 10.4 where we relax the *no-blackout* constraint by introducing a bound on the probability of blackout between decision

times. In this framework we show the effect on costs and empirical frequency of blackout of the constraint in a stationary and seasonal model, the latter being calibrated on real data obtained from a microgrid in Huatacondo, Chile. Section 10.5 that follows, explores the differences between the islanded system described so far and the framework where the microgrid is connected with the main electricity grid and, rather than using a diesel generator, power can be bought at a variable market price. It is interesting to see, in this scenario, how the policy is free to engage in price arbitrage or minimise cost to satisfy demand, according to the statistical properties of the demand and price processes. We conclude the chapter with section 10.6, where the previous analysis are elaborated into a concise set of guidelines for microgrid design and management.

10.2 System behavior

The aim of the section is to build a solid understanding of the behaviour of the microgrid in order to get an insight into the optimal design of the system. We decided to study the following aspects of the grid: battery capacity, represented by I_{max}; different proportions of renewable production, via the variance σ and the mean reversion b; tunable behaviour of the policy, via the switching cost K and curtailment cost C.

In order to be able to carry out our analysis, without introducing cumbersome economic and engineering details regarding the microgrid components, we have to make very simplistic assumptions. Our aim is, however, to guide the reader through a methodology that can be replicated to study real-world microgrid systems.

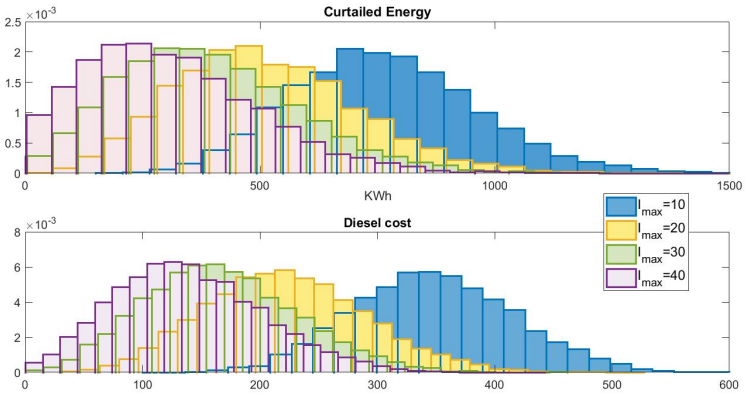


Figure 10.1: In the figure above we show histograms for different levels of battery capacity. In the top panel, we display the estimated probability density of the curtailed energy, while in the bottom panel the estimated density of the cost of operating the diesel generator. Notice that the decrease in cost and curtailed energy per kWh of additional capacity is smaller for high capacity batteries.

10.2.1 Battery capacity

We study first the behaviour of the system relatively to changes in the capacity of the battery. We would expect to observe a negative correlation between the quantity of diesel consumed and the battery size. We display in Figure 10.1 both the quantity of energy curtailed and the cost of running the diesel generator for different values of the battery capacity. We can observe that, as expected, increasing the size of the battery leads to lower diesel usage thanks to the higher proportion of renewable energy that is retained within the system. As the capacity of the battery reaches 30/40 KWh, we start observing a decrease in the cost-reduction per kWh of additional capacity suggesting that further analysis should be run in order to understand up to which size it is worth to pay to add storage capacity to the system.

We show now how to infer information about the optimal sizing of the battery, minimizing the trade-off between the installation cost of a bigger battery and the

reduced use of the diesel generator. Consider however that including battery ageing in the stochastic control problem is outside the scope of this analysis, but rather in this section we present only a post-optimisation analysis. Assuming that the microgrid runs under similar conditions for the next 10 years, we can quickly estimate the total throughput of energy for the different battery capacities. Consider now that a battery has not an infinite lifetime (see chapter 8), but rather it should be scrapped after, say, equivalent 4000 cycles (amount of energy for one full charge and discharge). Under the previous assumptions, we can compute how many batteries would be necessary to cover the next 10 years of operations. Similarly, using the data relative to the usage of the diesel generator for different levels of capacity, we can compute the operating cost of the diesel generator over the same time period. Further, exploiting the assumption about the lifetime of a battery, we obtain the cost of running the grid for 10 years as a function of the number of batteries. To conclude, assuming a linear cost of 400 EUR/KWh of capacity, we work out the installation cost of the different-size storage devices.

Once this information is collected we search for the minimum of the sum of installation and running cost and, in turn, we compute the optimal capacity. Figure 10.2, on the left, displays a graphical summary of the procedure just described and shows that, in our problem, the optimal size of the battery is 14 KWh under the current set of assumptions. Further, we study how much our result is affected by the cost per KWh of capacity, repeating the procedure above. We find that, as expected, when the cost increases, the size of the optimal battery decreases. Figure 10.2, on the right, displays such behaviour.

10.2.2 Renewable penetration

In this section, we want to investigate how robust the microgrid is to higher penetration of renewable generation, or, in other words, to what extent the algorithm

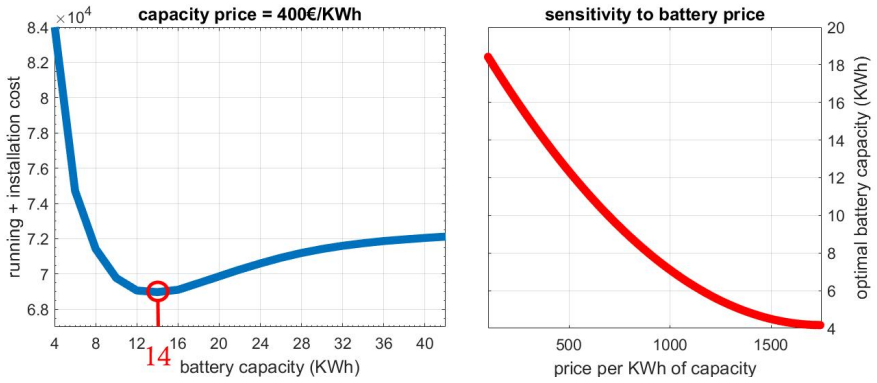


Figure 10.2: In the figure above we compute the total cost of installing and running the grid for ten years, assuming we replace the battery every 4000 cycles, and plot it against the battery capacity (left panel). From the corresponding minimum we can work out the optimal battery capacity and, further, compute the sensitivity of such result with respect to the cost per KWh of capacity.

can cope with increasing randomness and decreasing predictability of the system. To model this phenomena we assume that greater penetration of renewables can be modeled by increasing both the parameters for variance σ and the mean reversion rate b.

In order to establish the real added value provided by our stochastic optimisation algorithm, we compare the estimated policy with a heuristic myopic control which can be reproduced in our model solving the dynamic programming equation (9.3.10) taking constant conditional expectation with respect to the control. We show the value of the two control policies as a function of the increasing learning difficulty in Figure 10.3 where we observe the value of accounting for statistical estimation of future conditional expectations when taking decisions increases.

In figure 10.3 we present cost of diesel as a function of σ for stochastic and myopic policy. Since increasing σ alters the variance of the distribution, we define the mean reversion rate $b = \sigma^2/(2c)$ in order to ensure that the stationary distribution of the process (X_n) is constant while we increase σ. The stochastic policy leads to at least

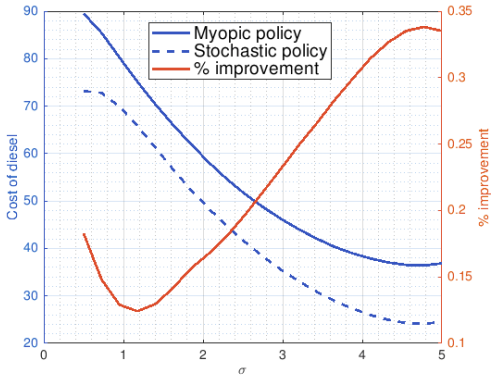


Figure 10.3: The figure represents the cost of the diesel usage for stochastic and myopic policy as a function of σ. The orange curve represents the percentage improvement in cost, as a proportion of the cost of the myopic policy.

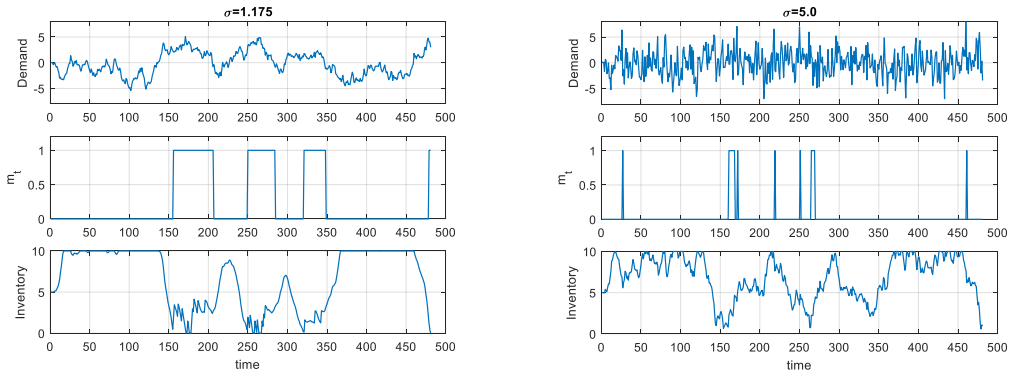


Figure 10.4: The figures in the left and right panel represent demand, diesel usage and the inventory dynamics for low and high σ respectively. It is important to mention that the mean reversion rate was chosen as $b = \sigma^2/8$, in order to ensure a constant variance of the process regardless of σ. Notice the low usage of the diesel generator in the figure on the right compared to the one on the left.

12% reduction in the cost of the diesel usage, compared to the myopic policy, and the difference magnifies with increasing "fluctuations" in the process. The decreasing relationship of the cost with σ signifies the importance of the battery storage system in the microgrid which absorbs the sharp change in the demand. In figure 10.4 we compare the demand for two different levels of the σ, the dynamics of the diesel generator and the inventory. Notice significantly less usage of the diesel for high fluctuations, $\sigma = 5$, compared to $\sigma = 1.175$.

The results of this experiment are affected by the over-pessimistic assumption of modeling greater penetration of renewables with an increasingly unpredictable, and eventually completely random, residual demand process. This sort of analysis can, however, provide insight into how much (weather and load) forecasting capability will be necessary for a given level of renewable penetration.

10.2.3 Switching and curtailment

We conclude this section by analyzing the dependence of the system behavior on two key parameters in the model: switching cost K and curtailment cost C. Switching cost is a system's property and the microgrid controller has little freedom over, however, the controller can significantly reduce the amount of curtailed energy by choosing the appropriate curtailment cost. In figure 10.5, we observe that increasing the curtailment cost reduces the total curtailed energy by approximately 4%. However, it comes at the cost of inefficient use of the diesel generator, which is represented on the right in the figure 10.5. The histograms represent the difference of the cost of diesel usage (blue) and the energy curtailed (orange) between the solution obtained for C=20 and the one obtained for C=2. Positive diesel cost depicts inefficient usage of the diesel at C=20 compared to C=2. Depending upon the specific cost functional for the diesel, the controller can use C as a parameter for better optimisation.

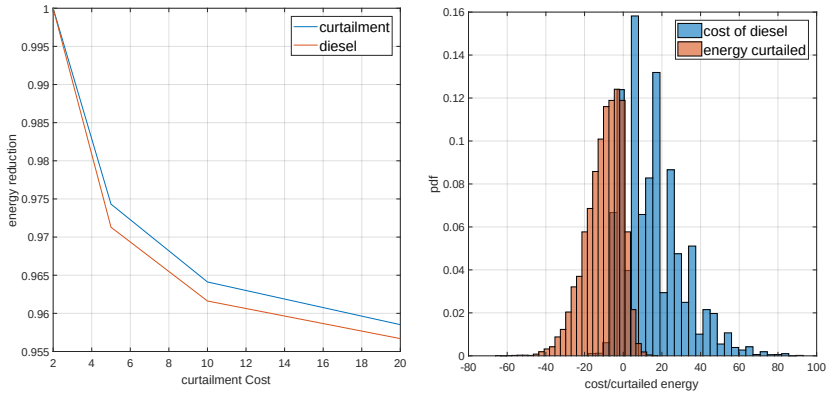
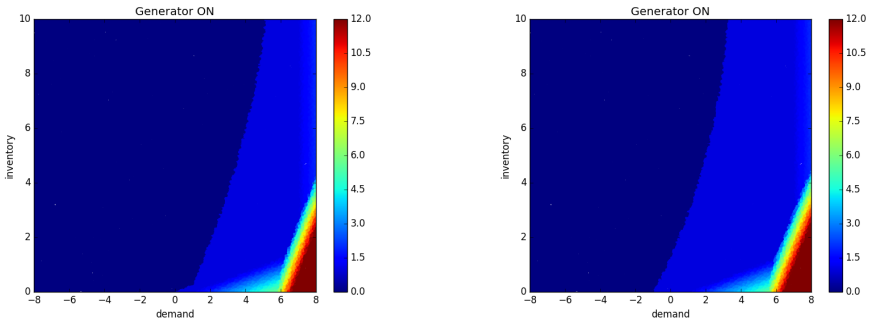


Figure 10.5: Line plot on the left represents the impact of curtailment cost on the total curtailed energy for different C as a proportion of curtailed energy at C=2. The histogram on the right represents the difference in the cost of diesel and the curtailed energy between C=20 and C=2. Notice the increase in curtailment cost leads to reduced curtailed energy but at the expense of inefficient diesel usage.

The optimal policy when the generator is ON $m_n = 1$ is significantly altered depending upon the switching cost. For example, in figure 10.6, we present the control maps associated with K=2 and K=5. As expected, larger switching cost disincentivises the controller to switch OFF the diesel generator once it's ON. However, we don't observe "significant" change in the control policy due to an increase in switching cost when the generator is OFF.

10.3 Comparison with deterministically trained policy

In this section we compare our stochastic optimisation algorithm with a deterministically trained policy. The latter is widely used in online optimisation where the solution is computed with respect to the best forecast available at a given time. We emulate this situation by computing the optimal set of actions for a



$K = 2$ $K = 5$

Figure 10.6: Figure on the left represents the control map for switching cost $K = 2$, while the figure on the right represents the control map for $K = 5$ when the generator is ON. Notice the increase in area for light blue (corresponding to $d = 1$) in the figure on the right because of increased switching cost.

particular deterministic demand trajectory at different levels of the inventory. We assume that the forecast of the demand is given by:

$$X_{n+1} = X_n + 0.5(6\sin(\frac{\pi t}{12}) - X_n)\Delta t; \quad t \in \{0, 1, \ldots, T-1\}. \tag{10.3.2}$$

Equation (10.3.2) implies periodicity of one day in the residual demand and is equivalent to $\sigma = 0$, $b = 0.5$ and $\Lambda_t = 6\sin(\frac{\pi t}{12})1$ in (9.2.2). Zero variance in the residual demand curve leads to a deterministic optimal control problem, rather than a stochastic control problem we have presented in section 5.

Notice that the deterministic optimal control problem results in a sequence of control maps $d_n : (w, m) \to [d_{min}, d_{max}] \cup 0$. As a result, although the policy has been trained on a deterministic residual demand, it dynamically adapts itself to different inventory levels and state of the diesel generator, when tested in a stochastic environment. We present the modified pseudocode in algorithm 7. There are two key differences compared with the algorithm presented in chapter 3: first, we use a one-dimensional projection of the value function and second, we replace regression with interpolation since there is no randomness left in the problem.

Algorithm 7 Regression Monte Carlo algorithm for deterministic demand

Simulate $\{X_n\}_{t=1}^{N}$ according to its dynamics;

Discretize I_n into M levels indexed by j s.t. $\{I_n^j\}_{j=1}^{M}$;

Initialize the value function $V(T, I_T^j, m_T) = g(I_T^j), \quad \forall j = 1, \ldots, M$ and $m_T = \{0, 1\}$;

for $t = N - 1$ to 1 **do**

  Find interpolation function $\mathcal{B}(t + 1, I_{n+1}, m)$ for $\{V(t + 1, I_{n+1}^j, m_{t+1})\}_{j=1}^{M}$ for each $m = 0, 1$

  Compute the set of admissible controls as $\mathcal{U}_t$

  for $j = 1$ to M **do**

    for $m = 0$ to 1 **do**

      $F = \mathcal{B}(t + 1, I_n^j, 0)$

      Compute the value function

$V(t, I_n^j, m) =$

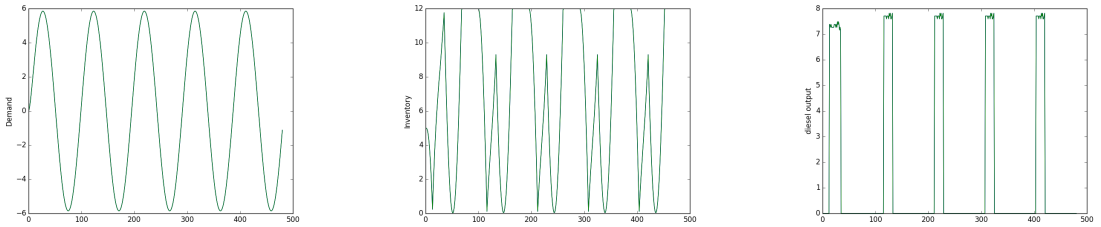
if $0 \in \mathcal{U}_t$:

$$\min_{d \in \mathcal{U}_t \setminus \{0\}} \left\{ \pi\rho(d) + CS_t \mathbb{1}_{\{S_n < 0\}} + \mathcal{B}(t + 1, I_n^j - B_t^d, 1) \right\} + K\mathbb{1}_{\{m=0\}} \wedge F$$

otherwise :

$$\min_{d \in \mathcal{U}_t} \left\{ \pi\rho(d) + CS_t \mathbb{1}_{\{S_n < 0\}} + \mathcal{B}(t + 1, I_n^j - B_t^d, 1) \right\} + K\mathbb{1}_{\{m=0\}}$$

output: control policy $\{\mathcal{B}(t, \cdot, \cdot)\}_{t=2}^{N}$.



Demand Inventory Diesel Output

Figure 10.7: The image illustrates the dynamics of the inventory and control for the deterministic control problem. The figure on the left represents the demand in equation (10.3.2), the optimal control of the diesel in the central figure and the corresponding dynamics of the inventory in the right figure.

In order to understand the solution of the deterministic problem, in figure 10.7 we present the dynamics of optimal control and inventory. As expected, diesel switches on when the demand is high and it keeps it running just long enough that the battery is empty before it faces negative residual demand to charge the battery. Moreover, there is a substantial curtailment of energy since the battery is not large enough to store all the excess energy.

In order to quantify the gain due to formulating the microgrid management problem as a stochastic control rather than traditional deterministic control, we compare the performance of the deterministically trained strategy of this section to its stochastic counterpart developed in this paper. While the deterministic control problem was solved using the residual demand curve (10.3.2), the stochastic control problem was fed in with the residual demand curve (10.3.3). Finally, we test both the strategies on fresh out-of-sample paths following the residual demand (10.3.3).

$$X_{n+1} = \left(X_n + 0.5(6\sin(\frac{\pi n}{12}) - X_n)\Delta t + 2\sqrt{\Delta t}\xi_n \right) \wedge 10 \; ; \quad n \in \{0, 1, \ldots, N-1\} \tag{10.3.3}$$

In figure 10.8, we present the histogram of the cost from the stochastic policy and

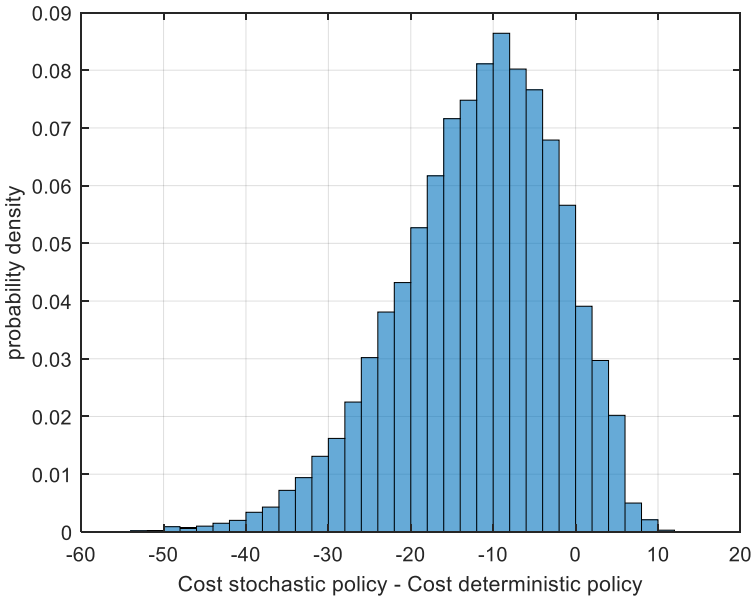


Figure 10.8: Difference of the Cost of Stochastic and deterministic policy for K=5

Switching Cost	K=2	K=5	K=10
Deterministic	138.56	162.63	201.52
Stochastic	131.86	150.49	178.22
% difference	4.84%	7.46%	11.56%

Table 10.1: Comparison between stochastic and deterministic policy: switching costs

the deterministic policy pathwise for 10,000 out-of-sample paths. As evident, most of the distribution lies on the negative side, implying gain due to stochastic policy. To measure this difference, in table 10.1, we quantify the gain of the stochastic policy for different switching cost. For switching cost of K=5, we observe that the stochastic policy is 7.5% better than the deterministic policy. As the switching cost increases, mistakes made by deterministic policy become more expensive leading to higher percentage difference.

Finally, Figure 10.9 displays the behavior of inventory and the cost along with a

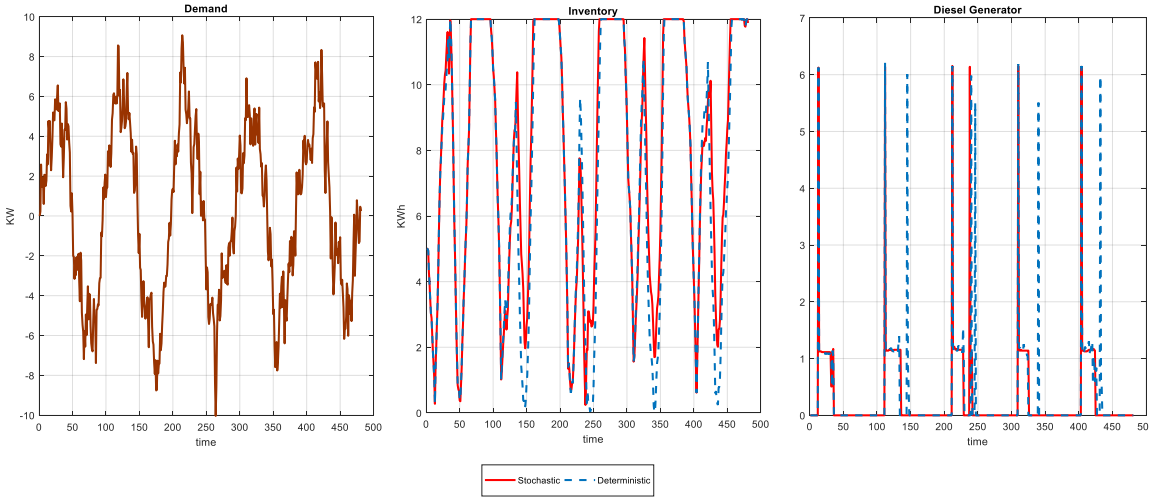


Figure 10.9: The figure above presents the pathwise comparison of stochastic and deterministic policy for the same demand on the left panel. The center panel represents dynamics of the inventory due to control on the right panel. Particularly notice the difference in switching times for the diesel in the deterministic policy and stochastic policy.

random trajectory of residual demand. In blue we show the stochastically trained control policy and in orange the deterministically trained. The stochastic policy has fewer switches of the diesel generator and thus lower costs. The spikes in the cost function for the deterministic policy is due to poor management of the inventory and thus inefficient usage of the microgrid.

10.4 Probabilistic constraint

In this section, we study the model introduced in section 9.4, where the constraint on blackouts is implemented through a probabilistic bound on the supremum of the imbalance process between the discrete decision times. Denote by $G_n(x, i, d)$ the random variable $\sup_{s\in[n,n+1)} S_s$.

The main difficulty in solving these kinds of models, compared to the original one, is that the set of admissible controls is not immediately available, but needs to be estimated. In the next section, we shortly describe different estimation techniques, for details refer to Balata et al. [6].

10.4.1 Admissible set estimation

In this section we propose different approaches to estimate the admissible set of controls $\mathcal{U}_n$ in equation (9.4.12). We follow three main themes to estimate the admissible set:

- **Probability estimation:** Given a state $X_n = x$, $I_n = i$ and $d \in \mathcal{U}_n$, we estimate, via simulation or via logistic and Gaussian process regression, the probability
$$p(x, i, d) := \mathbb{P}\Big(G_n(x, i, d) > 0\Big).$$
It follows that $d \in \mathcal{U}_n(x, i) \Leftrightarrow \hat{p}(x, i, d) < p$.

- **Density estimation:** Alternatively, we can first learn the distribution of the random variable $G_n(x, i, d)$, and then analytically infer the probability $\mathbb{P}\Big(G_n(x, i, d) > 0\Big)$ from the corresponding cumulative distribution function. Practically this is done by proposing a parametric family $\{f_\theta\}$ of distributions for G_n, fitting the underlying parameters θ based on an empirical sample of G_n's and then evaluating the resulting analytical probability $\mathbb{P}\Big(G_n(\hat{\Theta}(x, i, d)) > 0\Big)$. This approach potentially allows for "universal" solution across a range of constraint levels p.

- **Quantile estimation:** We may target the estimation of the quantile $q(x, i, d)$ of $G_n(x, i, d)$ for each $d \in \mathcal{D}^u$ which allows us to assess whether $d \in \mathcal{U}_n$ by checking $q(x, i, d) \leq 0$. Thus, instead of estimating $p(x, i, d)$ we learn $q(x, i, d)$

via quantile regression, SVM, empirical ranking and conditional tail average (conservative). The admissible sets $\mathcal{U}_n(x, i, d)$ is then defined as:

$$\mathcal{U}_n(x, i) := \Big\{ d : \hat{q}(x, i, d) < 0 \Big\}$$

10.4.2 Numerical implementation

We approximate the continuation value using piecewise continuous approximation with three degrees in (X) combined with interpolation in other dimensions (with discretisations of inventory $M_I = 10$). For the estimation of the admissible set $\mathcal{U}_n$, we approximate it using the methods described in section 10.4.1. We discretize the control space $[1, 10]$ into 51 values, while we use a fixed set of $M' = 20,000$ out-of-sample simulations to evaluate the performance of each method.

Numerical Gold Standard: In the absence of analytic benchmark, we use a so-called "empirical gold standard" (naive nested Monte Carlo estimation) to compare the output from the different estimation techniques. We employ a total simulation budget of $100 \times 100 \times 102 \times 10000 \approx 10^{10}$ to compute sample averages on a fine three-dimensional grid and estimate the probability of blackout. To estimate the continuation value, we use a piecewise linear approximation and grid size $1500 \times 15 \times 15$.

We consider now two situations: a stationary residual demand process, and the case with seasonality, as estimated from data measured in Huatacondo, Chile.

10.4.3 Stationary net-demand

In this subsection, we assume that the dynamics of the net-demand process is given by equation (9.2.2), where $\Lambda_n = 0, \ \forall n$. In this scenario, we need to estimate the admissible set $\mathcal{U}_0(\cdot)$ only once as a pre-processing step before starting the sequential

estimation of the continuation value function. While simplistic, this method helps us evaluate the performance of different admissible set estimation methods and the relative performance remains similar as we extend the model to assume a more realistic dynamics in section 10.4.4.

Table 10.2 lists other input parameters i.e. capacity of the battery $I_{\max}$, maximum charging rate $B_{\min}$, maximum discharging rate $B_{\max}$ and switching cost K for the model. The costs v for different methods by running the microgrid for 48 hours and

$b = 0.5$, $K = 5$
$I_{\max} = 10$ (kWh), $B_{\min} = -6$, $B_{\max} = 6$ (kW)
$T = 48$ (hours), $\Delta t = 0.25$ (hours)

Table 10.2: Parameters for the microgrid example. Chance constraint at level $p = 5\%$.

the corresponding estimated probability of constraint violations w_{freq}, is given in left panel of figure 10.10. The red star represents the cost from using the numerical gold standard and the circles represent the performance of different estimation methods for the admissible controls set on two-dimensions: cost and estimated probability of constraint violations. Recall the we choose $p = 5\%$. Although the figure as presented may have us believe equal importance for the two dimensions, we want to emphasize that it is not the case. Consider two hypothetical methods with the following values on the two dimensions: (i) zero cost and a large probability of constraint violations, or (ii) large cost with zero probability of violations. While method (ii) is acceptable, method (i) is not due to the large probability of constraint violations. In the absence of an exact threshold of acceptability (we can only estimate the accuracy of our solution), we consider methods closer to the numerical gold standard to be better than others. In particular, notice that the numbers reported in figure 10.10 do not represent the probability of a blackout, but rather the probabilities that the blackouts probability are greater than the chosen threshold p. With GPR and

logistic regression, we observe 0.4% and 0.9% estimated probability of violations, while the latter jumps up to 7.1% for quantile regression, 9.7% for SVM, 8.9% for CVaR and 9.5% for Ranking. Between GPR and Logistic regression, while GPR has a lower estimated probability of violations, it results in higher costs. As a result, we consider both as equally good methods. While all the methods seemed promising a priori, admissible set estimation via probability-based methods clearly seems to outperform quantile-based methods. Although it is not clear to us the source of this difference in the two types of methods, our experience on hypothetical dataset suggests that sample estimates of probability $\hat{p}(x, d)$ have significantly lower bias compared to sample estimates of quantiles $\hat{q}(x, d)$, thus partially explaining the difference. Notice that the results imply that, if one wants to reduce the probabilities of breaching the constraint, depending on the method employed, should choose a more conservative parameter p also incurring in higher running costs as a result.

Next, we test the sensitivity of the cost in terms of the violations threshold p (employing logistic regression $\hat{\mathcal{U}}_{LR}$) in the right panel of figure 10.10. Increasing the probability threshold decreases the cost as the set of admissible controls $\mathcal{U}$ monotonically increases in p. For example, any admissible control at $p = 1\%$ threshold is also feasible for $\forall p > 1\%$, thus the cost at 1% threshold should be greater than or equal to cost at, say, 10% threshold. In the same figure, we also present the realized frequency of blackouts as a function of the probability threshold. Notice that as $p \to 1$, the realized probability of blackouts $\to 20\%$ which indicates that the constraint is binding only about 20% of the time over the optimally controlled inventory paths.

10.4.4 Calibration on real data

Unlike in the previous example where we assumed stationary net-demand, in practice, it is quite common to observe seasonality in the net-demand process

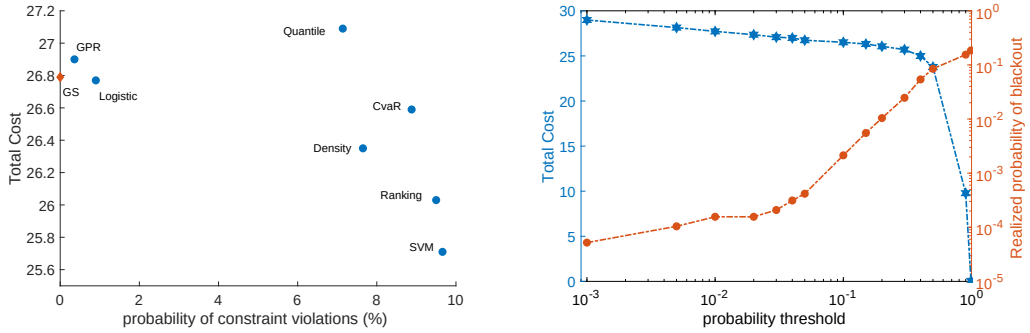


Figure 10.10: Left panel: Trade-off between cost and frequency of constraint violations w for the stationary model. Right panel: Total cost v (left axis, line with stars) and realized frequency of blackouts w (right axis, line with circles) at any time-step as a function of p employing the logistic regression method.

i.e. during the day when the output from the renewables is high, net-demand is negative and during morning/evening when demand exceeds the renewable output, net-demand is positive. Thus, we incorporate seasonality in the net-demand process (9.2.2).

We estimate the model using iterative methodology described in [41] and the data from a solar-powered microgrid in Huatacondo, Chile. We estimate the function Λ_n and σ_n by computing the mean and variance of the residual demand at 15-minute intervals using data from spring 2014 i.e. estimating the following sequence $\{\Lambda_1, \Lambda_2, \ldots, \Lambda_{96}\}$ and $\{\sigma_1, \sigma_2, \ldots, \sigma_{96}\}$. The estimated parameters are presented in the left panel of figure 10.11. We can observe that during the day i.e. $n \in [12, 20]$ (noon- 8:00 pm) the net-demand is negative $\Lambda_n < 0$ and is positive $\Lambda_n > 0$ for morning and night. Moreover, the variance σ_n is also higher during the day due to the unpredictable nature of electricity consumption. The mean reversion parameter was estimated to be $b = 0.3416$.

To visualize the interplay of the net-demand, inventory and optimal control, in the

center panel of figure 10.11, we present the average trajectories of the three processes by running the microgrid for 2 days i.e. 48 hours. During the morning hours when the net-demand is high and the battery is empty, the controller uses the diesel generator to match the net-demand process. Similarly, during the day when the renewable output is high and the net-demand is negative, the controller switches off the diesel generator and the battery charges itself. However, the non-trivial region is when the average net-demand changes sign, either from positive to negative around noon or negative to positive in the evening. During the former time-interval, the average optimal control process is either $\{0, 1\}$ (recall that minimum diesel output is 1), which is possible only due to the stochastic nature of net-demand. Similarly, during the evening when the net-demand becomes positive (as the renewable output declines), the controller quickly ramps up the diesel output to match the net-demand process. Finally, in the right panel of figure 10.11, we present the optimal control process and the net-demand process with the 2-standard deviation bands, in terms of the forward trajectories of $(X_s)_{s=0}^N$. As expected from the center panel, the time-periods around ramp-up or ramp-down of the diesel generator is when the control process differs from the net-demand process and experiences the greatest path-dependency and dispersion.

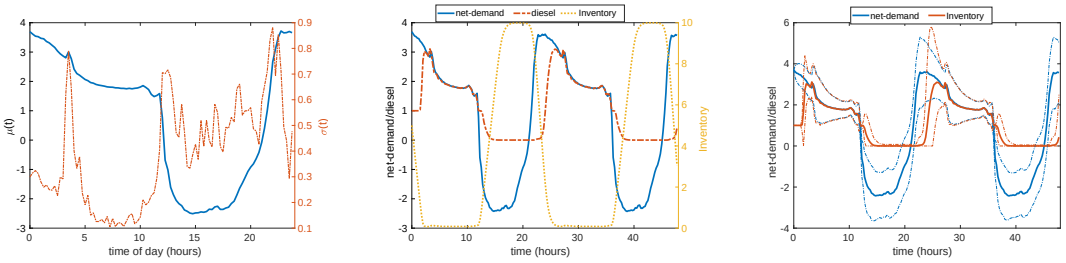


Figure 10.11: Left panel: Calibrated model parameters for a microgrid in Huatacondo, Chile; Center panel: Average values of net-demand, inventory and optimal control processes using the gold standard. Right panel: 2-Std error bands for net-demand and optimal control.

Method	Total Cost	w_{freq}	w_{Av}
Gold Standard INMC	53.39	0	0
Logistic regression	53.78	0.82%	0.11
GP regression	54.04	1.19%	0.14
Empirical percentile	52.82	22.2%	0.23
Support vector machine	38.87	45.39%	0.96

Table 10.3: Cost of running the microgrid $\hat{V}(0, \Lambda_0, 5)$, probability of constraint violations $w_{freq} = w(0, \Lambda_0, 5)$ and average violations w_{Av}. Recall that w_{freq} measures how often the constraint is breached, not the magnitude of the breach. w_{Av}, on the other hand, indicates the average missing power that would have constrained the probability of blackout below 5%

Next, to compare the performance of different methods, we present in table 10.3 the cost, average number of violations and the intensity of the violations. Comparing table 10.3 with figure 10.10 indicates that incorporating seasonal net-demand process does not change the relative order of performance between the methods, however, it does lead to a higher cost. This is expected as the diesel generator has to be used through the mornings and the evenings to match demand. This observation is similar to the results obtained in [20], in the context of natural gas storage, where authors observed higher valuation by incorporating seasonal effects in the model.

As in the previous example, we continue to observe the overall performance of logistic regression and Gaussian process regression to be very similar to our numerical gold standard despite significantly lower simulation budget. When fixing $p = 5\%$, the probability of violating the constraint for the different methods are 0.82% for Logistic, 1.19% GPR, but 22.2% and 44% for Ranking and SVM respectively. To illustrate the results, in figure 10.12 we present the average optimal control and the minimum admissible control had we been using the gold standard at those locations

for each of these methods. The solid blue line is expected to be above the dashed blue line, as it implies that on an average the method does not violate the constraint. Reverse order i.e. solid blue line below the dashed blue line is undesirable as the method will violate the constraint on an average. We observe that on an average logistic regression and Gaussian process regression do not violate the constraints but SVM quite obviously violates the constraint as the dashed-line is significantly higher than the solid line at several time-steps. Furthermore, the conservative nature of GP is also evident via the large difference between the average minimum admissible control and the average optimal control.

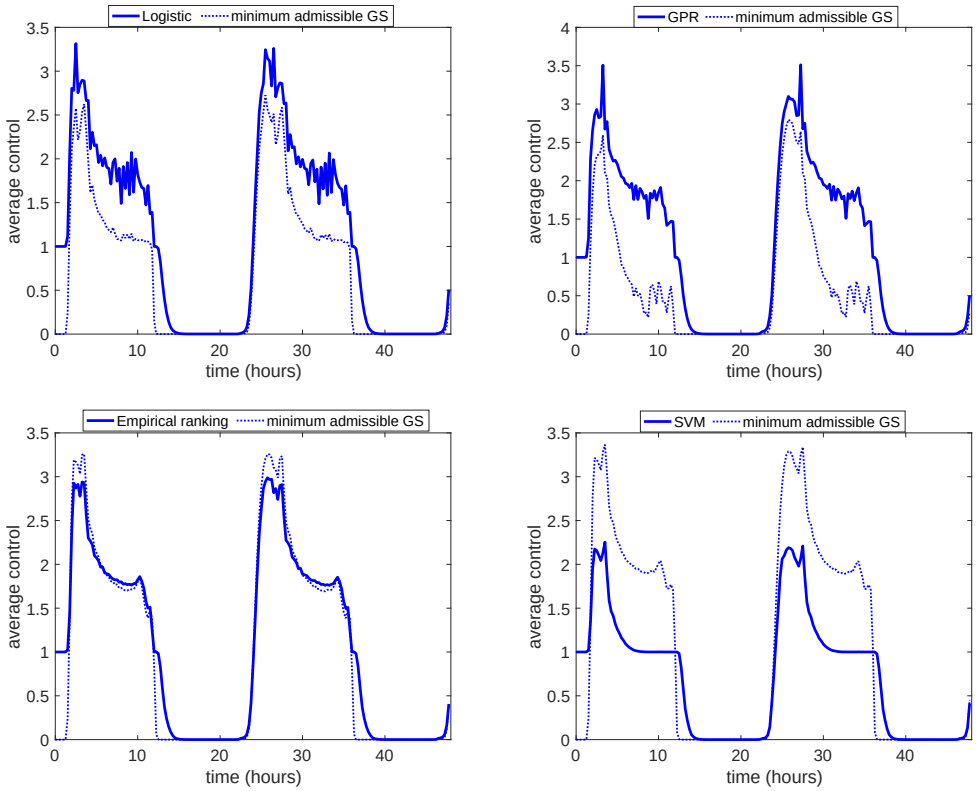


Figure 10.12: Average optimal control and the minimum admissible control using gold standard. We can clearly see how bad VaR and SVM are through these plots.

10.5 Non-islanded mode

After having analysed an islanded set-up for the microgrid in the previous sections, we now explore how the connectivity with a centralised grid changes management and design practices. In practice, we replace the diesel generator with a connection to the grid, from which electricity can be bought, but also sold.

For the model in this example we take inspiration from Scott et al. [68] and consider an electricity system composed of an intermittent energy supply represented by a wind turbine, a load or electricity demand represented by a group of residential buildings, a battery storage device and an interconnection with the main grid. Energy can flow through the different components of the system as indicated by the arrows in Figure 10.13; note that the power produced from the wind turbine is directly transferred to the load and used to satisfy the demand, the power in excess is transferred to the storage device which can sell it to the main grid or save it for later use in the battery. When the power from the wind turbine is insufficient to satisfy the demand, the shortfall is fulfilled either (or both) from the storage device discharging the battery or/and by buying electricity from the grid. Note that we assume that the main grid can always provide the power we require.

The operator has control on every flow of energy, making the dimensionality of the control process equal to five; we denote these flows at time n by X_n^{WD},X_n^{WB},X_n^{BD},X_n^{GB} and X_n^{GD}, where the notation $X^{i,j}$ indicates energy flowing from i to j and we denote the components as follow: W for the wind turbine, D for the demand, G for the grid interconnection and B for the battery. We constrain each flow to be positive except for X_n^{GB} which can assume negative values indicating that the battery is selling energy to the grid. Finally, we consider our problem stated on an hourly basis over a time horizon of two weeks.

10.5.1 Modelling choices

As in Scott et al. [68], we make the following modelling choices.

Wind Power

We compute the power in MWh produced by a wind turbine with rated power $2.1\,MW$ in one hour as a function of the wind speed w_n (average over an hour in m/s):

$$W_n = 10^{-6}\, C_p\, \rho\, A\, \max(w_n, \omega)^3,$$

where $C_p = 0.4$ is the power efficiency, $\rho = 1.225\, kg/m^3$ is the density of air, $\omega = 14m/s$ represents the rated output speed and $A = 50^2\pi$ is the area swept by the blades.

In particular, as commonly done, we model the centred square root of the wind speed (which we consider constant) as an AR(1) process:

$$\begin{aligned} w_n &= (y_n + \mu)^2 \\ y_{n+1} &= 0.7633 y_n + 0.4020 \xi_n, \quad \xi_n \sim \mathcal{N}(0,1) \end{aligned}$$

The coefficients are calibrated from 2010 real data registered in Maryneal, Texas; see Scott et al. [68] for details.

Electricity Price

The electricity price is modelled as is Scott et al. [68] as an exponential of a mean reverting process superimposed on a weekly seasonal component Y^{week} estimated from the data recorded in Texas in 2010:

$$\begin{aligned} P_{n+1} &= e^{Y_{n+1}} - 27.2531 + Y^{week}_{n+1} \\ Y_{n+1} &= Y_n + 0.2055(4.1995 - Y_n) + 0.11856 \xi_n + J_n, \quad \xi_n \sim \mathcal{N}(0,1) \end{aligned}$$

where $J_n = \xi_n^j 1_{\{U_n<0.017\}}$ is the jump component, $\xi_n^j \sim \mathcal{N}(0, 0.4229)$ and $U_n \sim \mathcal{U}(0,1)$.

Note that we introduce a fixed 10% transmission surcharge on the use of the grid for both supplying demand and exchanging energy with the battery. This is a requirement in order to make the optimal control unique, otherwise, it would be equivalent to supply power to the load from the battery through X^{BD} or through $X^{GB} \to X^{GD}$.

Electricity Demand

It is well known that temperature is one of the main drivers of the electricity demand of residential buildings, in fact, during periods of high or low temperature households tend to use air conditioning. The electricity demand is therefore modelled as a function of temperature; in Scott et al. [68] it is suggested to use an order 5 polynomial to fit the observed pairs of demand and temperature. Since estimated coefficients were not provided and in order to be consistent with the other data, we estimated our coefficients using observations recorded in Austin, Texas in 2010 (`http://www.noaa.gov/`). We obtained the following model for the demand:

$$\tilde{D}_n = 6784.9728 - 235.5911\, T_n - 2.2869\, T_n^2 + 0.8897\, T_n^3 - 0.0204\, T_n^4 + 0.000105\, T_n^5.$$

To scale down the demand to a few buildings we set $D_n = \frac{\tilde{D}_n}{30000}$. We model the temperature as a simple mean reverting process $T_n = \tilde{T}_n + \mu_n$, where μ_n represents a daily seasonal component estimated from the data and

$$\tilde{T}_{n+1} = -0.92\tilde{T}_n + 2.14\xi_n, \quad \xi_n \sim \mathcal{N}(0,1).$$

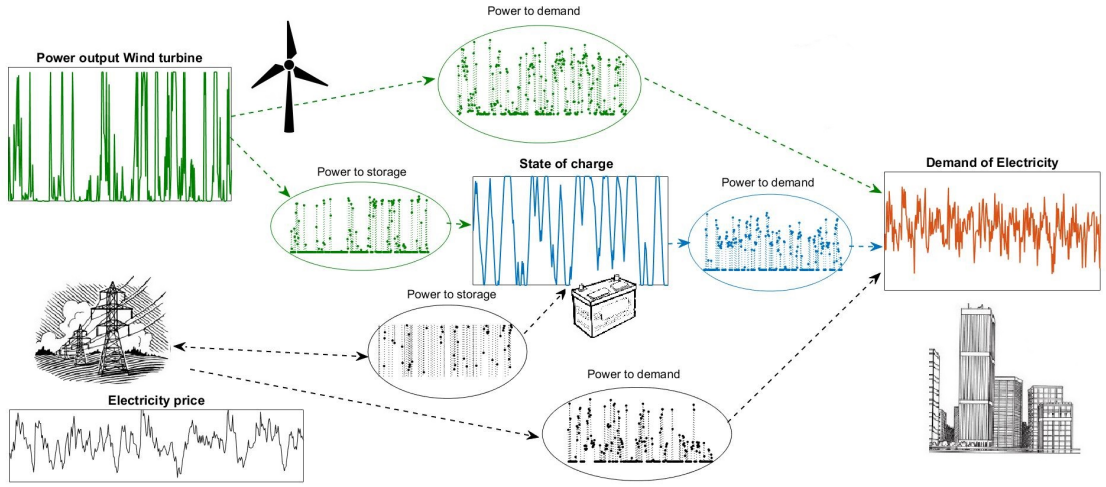


Figure 10.13: The diagram above shows components of the system. Boxes show evolution of state variables. Ellipses display energy flows among the components. Note that the algorithm directly controls the connection grid-battery and grid-demand as other flows are uniquely determined by those two.

The Battery

The most interesting component of the system is an energy storage device; it is represented by a battery with rated capacity $I_{\max} = 20\text{MWh}$ and maximum power in charge $B^{\max} = 2.1\text{MW}$ and discharge $B^{\min} = -2.1\text{MW}$. For simplicity of exposition we assume the battery is 100% efficient and its use does not take into account its ageing. The level of charge of the battery evolves as follow:

$$I_{n+1} = I_n + X_n^{GB} + X_n^{WB} - X_n^{BD}.$$

10.5.2 Problem formulation

We look at the system on an hourly basis during the first two weeks of January, which corresponds to 336 hourly time steps. We aim to provide power to the residential buildings minimising the cost of buying electricity from the grid. Since the problem

is stated in discrete time we will deal with energy rather than power. Notice that the following set of constraints is in place (we dropped the subscript n):

$$X^{GD} + X^{WD} + X^{BD} = D \tag{c.1}$$

$$X^{WD} = \min\{W, D\} \tag{c.2}$$

$$X^{WB} + X^{WD} = W \tag{c.3}$$

$$X^{GD} \geq 0 \tag{c.5}$$

$$X^{BD} \in [0, -\Pi_{\min}] \tag{c.6}$$

$$X^{GB} + X^{WB} - X^{BD} \in [B^{\min}, B^{\max}] \cap [-I, I_{\max} - I] \tag{c.7}$$

The first constraint (c.1) is natural and says that the demand has to be matched at all times. Constraints (c.2) and (c.3) regulate the operations of the wind turbine: all its power is delivered to the load first and the excess is transferred to the battery. As long as the current level of charge allows it the power is accumulated; the excess power is sold to the grid (for a price reduced by the 10% transmission surcharge), c.f. (c.7). Constraint (c.5) enforces the natural condition that power should only flow from the grid to the load. Constraint (c.6) limits the discharging rate of the battery enforcing a bound on the energy flowing from the battery to the demand. Finally, the last constraint (c.7) guarantees that the inventory stays between empty and full and that the maximum charging/discharging rates are maintained.

Note that the constraints (c.1)-(c.7) actually reduce the dimension of the controlled process from five to two (X^{GB} and X^{GD}), as the other components can be computed consequently. Recall that $X^{GD} \geq 0$ while the sign of X^{GB} depends on whether the battery buys or sells energy to the grid.

We can then write the profit of an electricity provider who owns the wind turbine and the battery and has contracted a given load, as electricity sold minus electricity

bought (net of transmission surcharge):

$$J\left(\left(D_s, W_s, P_s, B_s, I_s, X_s^{GB}, X_s^{GD}\right)_{s=1,\dots,N}\right) = \sum_{n=1}^{N} P_n\, D_n - 1.1 P_n (X_n^{GB} + X_n^{GD}).$$

Define the value function of the problem as the solution to the minimisation problem:

$$V(n, d, w, p, i) =$$
$$\max_{X \in \mathcal{U}} \left\{ \mathbb{E}\Big[J\Big((D_s, W_s, P_s, B_s, I_s, X_s^{GB}, X_s^{GD})_{s=n,\dots,N} \Big) | (D_n,\, W_n,\, P_n,\, I_n) = (d, w, p, i) \Big] \right\}, \tag{10.5.4}$$

where $\mathcal{U}$ is the set of admissible controls specified by constraints (c.1)-(c.7).

10.5.3 Numerical solution

We solve the problem (10.5.4) applying the Regress Later algorithm introduced in chapter 2. For this problem the choice of linear and quadratic basis functions, i.e. $\{D,\, W,\, P,\, D^2,\, W^2,\, P^2,\, I,\, D\,I,\, P\,I,\, W\,I,\, I^2\}$, works satisfactorily. The numerical complexity of the algorithm stems mainly from a non-trivial two dimensional constrained optimisation problem for the control which is solved at each time step and for each training point.

Once the estimated optimal policy has been computed we run it forward in time to assess its value; in Figure 10.13 we display a particular realisation of all the flows of energy among the components. It can be observed that the battery stores the wind power when this exceeds the demand for electricity and transfers it to the demand instead of buying electricity from the grid at a high price. Over the two weeks of interest the system considered has an average demand of electricity of 320 KW compared with an average production of wind power of 350 KW meaning that the system is a net exporter of electricity. Figure 10.15 displays some of the interactions within the system. The scatter plot on the left shows the energy flowing to the

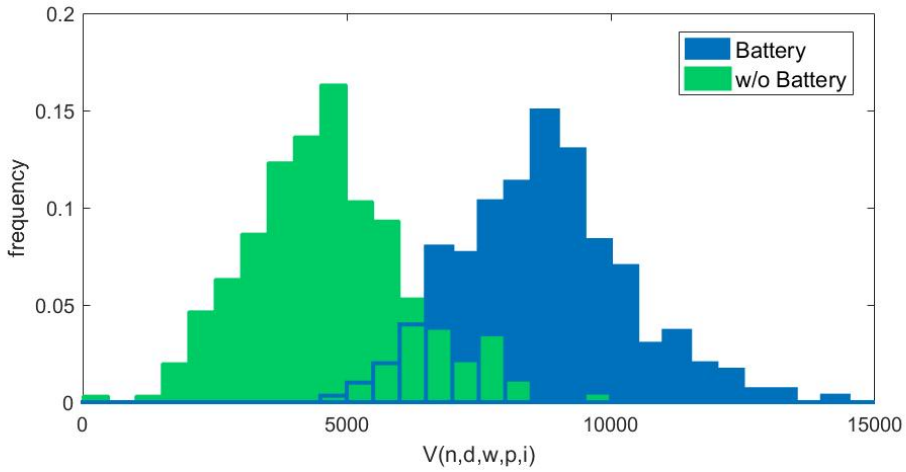


Figure 10.14: The empirical probability distributions of the performance J at time 0 with the initial data $(d, w, p, i) = (0.15, 0, 35, 10)$, representing the profits obtained in the system with (blue) and without (green) the use of the battery. There are 30 bins each 500 units wide.

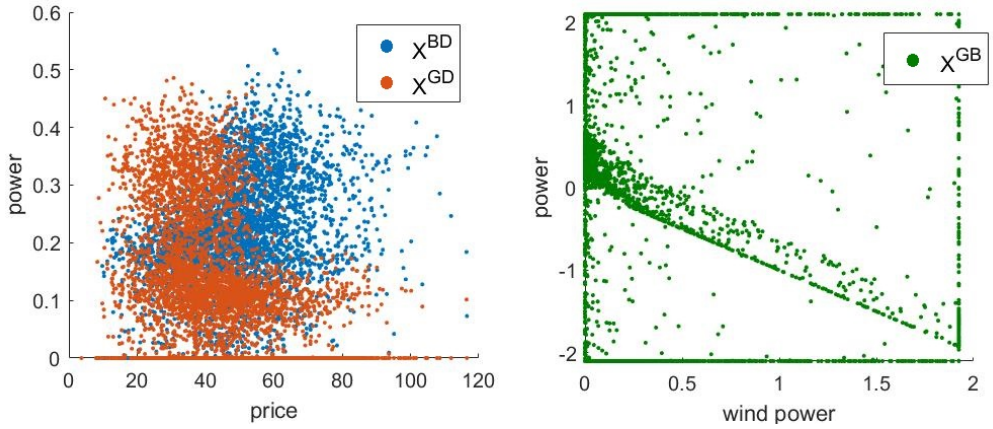


Figure 10.15: This graphs display interactions within the system in the form of scatter plots. The left plot shows X^{GD} and X^{BD} against the price revealing that the first tends to provide power when price is low, the latter when price is high. The right plots show how the connection battery-grid (X^{GB}) is influenced by the wind power; note that the high level of wind often results in selling energy to the grid in order to decrease the level of charge.

demand from the battery (blue) and from the grid (orange). There is a general tendency for the power to be bought from the grid when its price is low while it is supplied mainly by the battery when the price is high. The scatter plot on the right displays the interactions between wind power and transfers through the connection battery-grid. We observe that when the wind power is high the battery tends to sell energy to the grid in order to maintain an acceptable level of inventory.

The solution of the optimisation problem of this subsection enables the study of the economic value of using a battery to assist the operations of a wind turbine. To this end, we evaluate the estimated policy on a new and larger set of trajectories of the underlying processes (wind, demand and price). This is further compared to the profits made in a system with no battery installed, i.e., all excess wind energy immediately sold to the grid. We find that the 25% and 75% quantiles of the profits for operating the system without the use of the storage over two weeks are $3595\,USD$, and $5498\,USD$, respectively. When the flexibility provided by the storage is optimally used the respective quantiles are $7556\,USD$ and $9598\,USD$, showing an increase in the profits of about 90%. Further details about the distribution of profits are displayed in Figure 10.14. Considering the inventory behaviour displayed in Figure 10.13 as "typical" we can compute the total throughput of energy to be 490 MWh in the two weeks. Assuming further that the battery sustains 4000 full cycles we obtain an estimated life-span of 320 weeks or approximately 6 years. Assuming further a stationary flow of profit during the life of the battery we conclude that the installation of the storage alongside the wind turbine will bring additionally $640000\,USD$ in profits (assuming zero scrap value and zero interest rate).

10.6 Managing a microgrid: final guidelines.

In this section we reflect upon the experiments described in this chapter and, as a result, present some concise guidelines for microgrid management.

In section 10.3 we learn that the use of stochastic-based optimisation, compared with traditional industry standard techniques, reduces cost between 5% and 10% and, in particular, the increase in performance is particularly evident during those days in which the system is under the most stressful conditions. This result suggests that, for a microgrid operator, an investment in advanced control software and in hiring specialised personnel is, in the long-term, a profitable choice. The gains from a stochastic formulation increase as the forecasting capabilities decrease, suggesting that in a world with growing renewable penetration, this kind of optimisation becomes a requirement for viable microgrid management.

Section 10.2.1 and 10.5 show, in both islanded and fully connected configuration, the paramount importance of a battery storage in the microgrid system. An investment in this device allows to exploit more efficiently the power produced by renewable generators and to reduce the reliance on dispatchable units or external devices. The high cost of this component, however, requires careful optimisation of the sizing, we found that avoiding over-sizing can reduce costs by about 3%, while the effect of under-sizing is often more severe.

Section 10.5 explores the design where the microgrid is connected to the main power grid. Such configuration shows the extent to which a battery device can be useful, even when renewable power does not have to be curtailed (as in the islanded case), but can be sold at a variable price, helping to optimise the interplay between distributed generation and power trading. It is widely recognised however, even though outside the scope of this investigation, that when operating in a connected configuration, battery device can also provide very remunerative ancillary services

like frequency regulation.

Finally, in our analysis of probability constrained blackouts, in section 10.4, we showed the cost of reliability in a microgrid system (figure 10.10). It is possible to maintain the probability of blackout below the required threshold encountering only a modest increase in cost. This experiment also allowed us to, once again, prove the importance of the battery, which allows to make mistakes (being over-conservative) without paying a dear price in terms of usage of the dispatchable generator.

The microgrid operator, in conclusion, should recognise the importance of sophisticated control and modelling software to manage and design the system and ensure a successful financial outcome, maintaining the most appropriate level of reliability.

Bibliography

[1] Clemence Alasseur, Alessandro Balata, Sahar Ben Aziza, Aditya Maheshwari, Peter Tankov, and Xavier Warin. Regression Monte Carlo for microgrid management. *arXiv:1802.10352*, 2018.

[2] Oghenetejiri Harold Anuta, Phil Taylor, Darren Jones, Tony McEntee, and Neal Wade. An international review of the implications of regulatory and electricity market structures on the emergence of grid scale electricity storage. *Renewable and Sustainable Energy Reviews*, 38:489–508, 2014.

[3] Alessandro Balata and Jan Palczewski. Regress-Later Monte Carlo for optimal control of Markov processes. *arXiv:1712.09705*, 2017.

[4] Alessandro Balata and Jan Palczewski. Regress-Later Monte Carlo for optimal inventory control and applications in energy. *arXiv:1703.06461*, 2017.

[5] Alessandro Balata, Côme Huré, Mathieu Laurière, Huyên Pham, and Isaque Pimentel. A class of finite-dimensional numerically solvable McKean-Vlasov control problems. *ESAIM, Proceedings CEMRACS*, 2018.

[6] Alessandro Balata, Michael Ludkovski, Aditya Maheshwari, and Jan Palczewski. Regression Monte Carlo for probability constrained optimal control problems. *to appear*, 2019.

[7] Anthony Barre, Benjamin Deguilhem, Sebastien Grolleau, Mathias Gerard, Frederic Suard, and Delphine Riu. A review on lithium-ion battery ageing

mechanisms and estimations for automotive applications. *Journal of Power Sources*, 241:680–689, 2013.

[8] Christian Bayer, Martin Redmann, and John Schoenmakers. Dynamic programming for optimal stopping via pseudo-regression. *arXiv:1808.04725v2*, 2018.

[9] Christian Bender and Jessica Steiner. *Least-Squares Monte Carlo for Backward SDEs*, pages 257–289. Springer Berlin Heidelberg, Berlin, Heidelberg, 2012.

[10] V. E. Benes, L. A. Shepp, and H. S. Witsenhausen. Some solvable stochastic control problems. *Stochastics*, 4:39–83, 1980.

[11] Eric Beutner, Antoon Pelsser, and Janina Schweizer. Fast Convergence of Regress-Later estimates in Least Squares Monte Carlo. *arXiv:1309.5274*, 2013.

[12] Alexander Boogert and Cyriel de Jong. Gas storage valuation using a Monte Carlo method. *The Journal of Derivatives*, 15(3):81–98, 2008.

[13] Philippe Briand and Céline Labart. Simulation of BSDEs by Wiener chaos expansion. *The Annals of Applied Probability*, 24(3):1129–1171, 2014.

[14] Mark Broadie and Paul Glasserman. A stochastic mesh method for pricing high-dimensional American options. *Journal of Computational Finance*, 7(35): 35–72, 2004.

[15] Mark Broadie, Paul Glasserman, and Zachary Ha. Pricing American options by simulation using a stochastic mesh with optimized weights. In Stanislav P. Uryasev, editor, *Probabilistic Constrained Optimization*, volume 49 of *Nonconvex Optimization and Its Applications*, pages 26–44. Springer, 2000.

[16] Rene Carmona and Michael Ludkovski. Pricing asset scheduling flexibility using optimal switching. *Applied Mathematical Finance*, 15(5-6):405–447, 2008.

[17] Rene Carmona and Michael Ludkovski. Valuation of energy storage: an optimal switching approach. *Quantitative Finance*, 10(4):359–374, 2010.

[18] Rene Carmona, Jean-Pierre Fouque, and Li-Hsien Sun. Mean field games and systemic risk. *Communications in Mathematical Sciences*, 13(4):911–933, 2015.

[19] Jacques F. Carriere. Valuation of the early-exercise price for options using simulations and nonparametric regression. *Insurance: Mathematics and Economics*, 19(1):19–30, 1996.

[20] Zhuliang Chen and Peter A. Forsyth. A semi-Lagrangian approach for natural gas storage valuation and optimal operation. *SIAM Journal on Scientific Computing*, 30(1):339–368, 2008.

[21] Emmanuelle Clement, Damien Lamberton, and Philip Protter. An analysis of a least squares regression method for American option pricing. *Finance stoch*, 6(4):449–471, 2002.

[22] Jérôme Collet, Olivier Féron, and Peter Tankov. Optimal management of a wind power plant with storage capacity. HAL preprint hal-01627593, 2017. URL `https://hal.archives-ouvertes.fr/hal-01627593`.

[23] Michel Denault, Jean-Guy Simonato, and Lars Stentoft. A simulation-and-regression approach for stochastic dynamic programs with endogenous state variables. *Computers & Operations Research*, 40(11):2760–2769, 2013.

[24] Huajie Ding, Zechun Hu, and Yonghua Song. Stochastic optimization of the daily operation of wind farm and pumped-hydro-storage plant. *Renewable Energy*, 48:571–578, 2012.

[25] Huajie Ding, Zechun Hu, and Yonghua Song. Rolling optimization of wind farm and energy storage system in electricity markets. *IEEE Transactions on Power Systems*, 30(5):2676–2684, 2015.

[26] Matthieu Dubarry, Nicolas Vuillaume, and Bor Yann Liaw. From single cell model to battery pack simulation for li-ion batteries. *Journal of Power Sources*, 186(2):500–507, 2009.

[27] Daniel Egloff. Monte Carlo algorithms for optimal stopping and statistical learning. *The Annals of Applied Probability*, 15(2):1396–1432, 2005.

[28] Emmanuel Gobet. *Monte Carlo Methods and Stochastic Processes: From Linear to Non-Linear*. Chapman and Hall/CRC, 2016.

[29] Bastian Joachim Felix and Christoph Weber. Gas storage valuation applying numerically constructed recombining trees. *European Journal of Operational Research*, 216(1):178–187, 2012.

[30] Aoife M. Foley, Paul G. Leahy, Antonino Marvuglia, and Eamon J. McKeogh. Current methods and advances in forecasting of wind power generation. *Renewable Energy*, 37:1–8, 2012.

[31] Paul Glasserman and Bin Yu. Simulation for American options: regression now or regression later? In Harald Niederreiter, editor, *Monte Carlo and Quasi-Monte Carlo Methods*, pages 213–226. Springer Berlin Heidelberg, 2002.

[32] Paul Glasserman and Bin Yu. Number of paths versus number of basis functions in American option pricing. *The Annals of Applied Probability*, 14(4):2090–2119, 2004.

[33] Kossi Gnameho, Mitja Stadje, and Antoon Pelsser. A regress-later algorithm for backward stochastic differential equations. *arXiv:1706.07986*, 2017.

[34] Emmanuel Gobet, Jean-Philippe Lemor, and Xavier Warin. A regression-based Monte Carlo method to solve backward stochastic differential equations. *The Annals of Applied Probability*, 15(3):2172–2202, 2005.

[35] Pierre Haessig, Thibaut Kovaltchouk, Bernard Multon, Hamid Ben Ahmed, and Stephane Lascaud. Computing an optimal control policy for an energy storage. *arXiv:1404.6389*, 2013.

[36] Pierre Haessig, Bernard Multon, Hamid Ben Ahmed, Stephane Lascaud, and Lionel Jamy. Aging-aware NaS battery model in a stochastic wind-storage simulation framework. In *PowerTech, 2013 IEEE Grenoble*, pages 1–6. IEEE, 2013.

[37] Pierre Haessig, Hamid Ben Ahmed, and Bernard Multon. Energy storage control with aging limitation. In *PowerTech, 2015 IEEE Eindhoven*, pages 1–6. IEEE, 2015.

[38] Pierre Haessig, Bernard Multon, Hamid Ben Ahmed, Stéphane Lascaud, and Pascal Bondon. Energy storage sizing for wind power: impact of the autocorrelation of day-ahead forecast errors. *Wind Energy*, 18(1):43–57, 2015.

[39] Naoki Hayashi, Masaaki Nagahara, and Yutaka Yamamoto. Robust AC voltage regulation of microgrids in islanded mode with sinusoidal internal model. *SICE Journal of Control, Measurement, and System Integration*, 10(2):62–69, 2017.

[40] Onesimo Hernandez-Lerma and Jean Bernard Lasserre. *Discrete-time Markov control processes: basic optimality criteria*. Applications of Mathematics. Springer-Verlag New York, 1996.

[41] Benjamin Heymann, J. Frédéric Bonnans, Pierre Martinon, Francisco J. Silva, Fernando Lanas, and Guillermo Jiménez-Estévez. A stochastic continuous time model for microgrid energy management. In *2016 European Control Conference (ECC)*, pages 2084–2089, 2016.

[42] Benjamin Heymann, J. Frédéric Bonnans, Pierre Martinon, Francisco J. Silva, Fernando Lanas, and Guillermo Jiménez-Estévez. Continuous optimal control

approaches to microgrid energy management. *Energy Systems*, 9(1):59–77, 2018.

[43] Robert A. Huggins. *Energy Storage*. Springer, Cham, 2016.

[44] Saul D. Jacka. A finite fuel stochastic control problem. *Stochastics*, 10(2): 103–113, 1983.

[45] Shashi Jain and Cornelis W. Oosterlee. The stochastic grid bundling method: Efficient pricing of Bermudan options and their greeks. *Applied Mathematics and Computation*, 269:412–431, 2015.

[46] Idris Kharroubi, Nicolas Langrené, and Huyén Pham. A numerical algorithm for fully nonlinear HJB equations: an approach by control randomization. *Monte Carlo Methods and Applications*, 20(2):145–165, 2014.

[47] Yerkin Kitapbayev, John Moriarty, and Pierluigi Mancarella. Stochastic control and real options valuation of thermal storage-enabled demand response from flexible district energy systems. *Applied Energy*, 137:823–831, 2015.

[48] Harold J. Kushner and Paul G. Dupuis. *Numerical Methods for Stochastic Control Problems in Continuous Time*. Applications of Mathematics. Springer-Verlag New York, 2001.

[49] Nicolas Langrené, Tanya Tarnopolskaya, Wen Chen, Zili Zhu, and Mark Cooksey. New regression Monte Carlo methods for high-dimensional real options problems in minerals industry. In *21st International Congress on Modelling and Simulation*, pages 1077–1083, 2015.

[50] Hao Liang and Weihua Zhuang. Stochastic modeling and optimization in a microgrid: A survey. *Energies*, 7(4):2027–2050, 2014.

[51] Francis A. Longstaff and Eduardo S. Schwartz. Valuing American options by simulation: A simple least-squares approach. *Review of Financial Studies*, 14 (1):113–147, 2001.

[52] Michael Ludkovski and Aditya Maheshwari. Simulation methods for stochastic storage problems: A statistical learning perspective. *arXiv:1803.11309*, 2018.

[53] S. Mashayekh, M. Stadler, G. Cardoso, M. Heleno, S. Chalil Madathil, H. Nagarajan, R. Bent, M. Mueller-Stoffels, X. Lu, and J. Wang. Security-constrained design of isolated multi-energy microgrids. *IEEE Transactions on Power Systems*, 33(3):2452–2462, 2018.

[54] Salman Mashayekh, Michael Stadler, Gonçalo Cardoso, and Miguel Heleno. A mixed integer linear programming approach for optimal der portfolio, sizing, and placement in multi-energy microgrids. *Applied Energy*, 187(Supplement C): 154–168, 2017.

[55] Jr. H. P. McKean. Appendix : a free boundary problem for the heat equation arising from a problem of mathermatical economics. *Industrial Management Review*, 6:32–39, 1965.

[56] Sean P. Meyn, Prabir Barooah, Ana Busic, and Yue Chen. Ancillary service to the grid using intelligent deferrable loads. *IEEE Transactions on Automatic Control*, 60(11):2847–2862, 2015.

[57] Aditya Mishra, Ramesh Sitaraman, David Irwin, and Ting Zhu. Integrating energy storage in electricity distribution networks. In *e-Energy '15 Proceedings of the 2015 ACM Sixth International Conference on Future Energy Systems*, pages 37–46, 2015.

[58] Manuel Moreno and Javier F. Navas. On the robustness of least-squares Monte

Carlo (LSM) for pricing American derivatives. *Review of Derivatives Research*, 6(2):107–128, 2003.

[59] Selvaprabu Nadarajah and Nicola Secomandi. Relationship between least squares Monte Carlo and approximate linear programming. *Operations Research Letters*, 45:409–414, 2017.

[60] Selvaprabu Nadarajah, François Margot, and Nicola Secomandi. Comparison of least squares Monte Carlo methods with applications to energy real options. *European Journal of Operational Research*, 256(1):196 – 204, 2017.

[61] Selvaprabu Nadarajah, François Margot, and Nicola Secomandi. Comparison of least squares Monte Carlo methods with applications to energy real options. *European Journal of Operational Research*, 256(1):196–204, 2017.

[62] Lanre Olatomiwa, Saad Mekhilef, A.S.N. Huda, and Olayinka S. Ohunakin. Economic evaluation of hybrid energy systems for rural electrification in six geo-political zones of nigeria. *Renewable Energy*, 83(Supplement C):435 – 446, 2015.

[63] Daniel E Olivares, Ali Mehrizi-Sani, Amir H Etemadi, Claudio A Cañizares, Reza Iravani, Mehrdad Kazerani, Amir H Hajimiragha, Oriol Gomis-Bellmunt, Maryam Saeedifard, Rodrigo Palma-Behnke, et al. Trends in microgrid control. *IEEE Transactions on smart grid*, 5(4):1905–1919, 2014.

[64] Warren B. Powell. *Approximate Dynamic Programming: Solving the curses of dimensionality*, volume 703. John Wiley & Sons, 2007.

[65] William H. Press, Saul A. Teukolsky, and Brian P. Flannery. *Numerical Recipes*. Cambridge University Press, 2007.

[66] S. Surender Reddy, Vuddanti Sandeep, and Chan-Mook Jung. Review of

stochastic optimization methods for smart grid. *Frontiers in Energy*, 11(2): 197–209, Jun 2017.

[67] L. Chris G. Rogers and David Williams. *Diffusions, Markov Processes and Martingales: Volume 1 Foundations*, volume 1. Cambridge University Press, 2000.

[68] Warren R. Scott, Warren B. Powell, and Somayeh Moazeni. Least squares policy iteration with instrumental variables vs. direct policy search: Comparison against optimal benchmarks using energy storage. 2014.

[69] Nicola Secomandi. Optimal commodity trading with a capacitated storage asset. *Management Science*, 56(3):449–467, 2010.

[70] Albert N. Shiryaev. *Optimal Stopping Rules*. Springer-Verlag Berlin Heidelberg, 2008.

[71] J. L. Snell. Applications of martingale system theorems. *Transactions of the American Mathematical Society*, 73(2):293–312, 1952.

[72] Halil Mete Soner and Wendell H. Fleming. *Controlled Markov Processes and Viscosity Solutions*. Springer-Verlag New York, 2006.

[73] Lars Stentoft. Assessing the least squares Monte-Carlo approach to American option valuation. *Review of Derivatives Research*, 7(2):129–168, 2004.

[74] Glen Swindle, Hyungsok Ahn, and Albina Danilova. Storing arb. *Wilmott*, 2002.

[75] Nizar Touzi. *Optimal Stochastic Control, Stochastic Target Problems, and Backward SDE*. Springer, 2013.

[76] J. N. Tsitsiklis and B. Van Roy. Regression methods for pricing complex american-style options. *Trans. Neur. Netw.*, 12(4):694–703, July 2001.

[77] A. Wald. Sequential tests of statistical hypotheses. *The Annals of Mathematical Statistics*, 16(2):117–186, 1945.

[78] Henry Wolkowicz and George P. H. Styan. *Bounds for eigenvalues using traces*, volume 29. Elsevier Inc., 1980.

[79] Daniel Z. Zanger. Convergence of a least squares Monte Carlo algorithm for bounded approximating sets. *Applied Mathematical Finance*, 16(2):123–150, 2009.

[80] Daniel Z. Zanger. Quantitative error estimates for a least-squares Monte Carlo algorithm for American option pricing. *Finance and Stochastics*, 17(3):503–534, 2013.

[81] Guangzhi Zhao and Matt Davison. Optimal control of hydroelectric facility incorporating pump storage. *Renewable Energy*, 34(4):1064–1077, 2009.