SAPIENZA
UNIVERSITÀ DI ROMA

An optimal control approach to Reinforcement Learning

Scuola di dottorato Vito Volterra

Dottorato di Ricerca in Matematica – XXXIV Ciclo

Candidate

Andrea Pesare
ID number 1616730

Thesis Advisor

Prof. Maurizio Falcone

October 2021

Thesis defended on 25 January 2022
in front of a Board of Examiners composed by:

Prof. Michele Piana (chairman)
Prof.ssa Benedetta Morini
Prof.ssa Silvia Noschese

An optimal control approach to Reinforcement Learning
Ph.D. thesis. Sapienza – University of Rome

This thesis has been typeset by LaTeX and the Sapthesis class.

Version: 25 January 2022

Author's email: pesare@mat.uniroma1.it

Abstract

Optimal control and Reinforcement Learning deal both with sequential decision-making problems, although they use different tools. In this thesis, we have investigated the connection between these two research areas. In particular, our contributions are twofold.

In the first part of the thesis, we present and study an optimal control problem with uncertain dynamics. As a modeling assumption, we will suppose that the knowledge that an agent has on the current system is represented by a probability distribution π on the space of possible dynamics functions. The goal is to minimize an average cost functional, where the average is computed with respect to the probability distribution π. This framework describes well the behavior of a class of model-based RL algorithms, which build a probabilistic model (here represented by π) of the dynamics, and then compute the control by minimizing the expectation of the cost functional with respect to π. In this context, we establish some convergence results for the value function and the optimal control. These results constitute an important step in the convergence analysis of this class of RL algorithms.

In the second part, we propose a new online algorithm for dealing with LQR problems where the state matrix $\widehat{A}$ is unknown. Our algorithm provides an approximation of the dynamics and finds a suitable control at the same time, during a single simulation. It is based on an integration between RL and optimal control techniques. A probabilistic model is updated at each iteration using Bayesian linear regression formulas, and the control is obtained in feedback form by solving a Riccati differential equation. Numerical tests show how the algorithm can efficiently bring the system to the origin, despite not having full knowledge of the system at the beginning of the simulation.

Contents

Notation

Abbreviations

BLR	Bayesian linear regression
BNN	Bayesian Neural Network
CE	Certainty equivalence
DP	Dynamic programming
GP	Gaussian Process
HJB	Hamilton-Jacobi-Bellman
LQR	Linear-Quadratic Regulator
LR	Linear regression
LS	Least squares
MBRL	Model-based Reinforcement Learning
MDP	Markov Decision Process
ODE	Ordinary differential equation
OC	Optimal control
PDE	Partial differential equation
RL	Reinforcement Learning

Mathematical symbols

$\mathbf{0}_n$	the zero matrix in $\mathbb{R}^{n\times n}$
$\|A\|_2$	the 2-norm $\|A\|_2 \coloneqq \sup\left\{x^T A y : \ \ x, y \in \mathbb{R}^n, \ \ |x| = |y| = 1\right\}$, for $A \in \mathbb{R}^{n\times n}$
$B(x_0, r)$	the open ball $\{x \in \mathbb{R}^n : |x - x_0| < r\}$
$BUC(X)$	the set of bounded and uniformly continuous functions $f\colon X \to \mathbb{R}$
$\mathscr{B}_X$	the collection of all Borel sets on a given topological space X
$C(X;Y)$	the space of continuous functions $f\colon X \to Y$
$C_b(X)$	the set of bounded continuous functions $f\colon X \to \mathbb{R}$
$C^1(X)$	the set of continuously differentiable functions $f\colon X \to \mathbb{R}$
$d_2(A, A')$	the distance $d_2(A, A') \coloneqq \|A - A'\|_2$ between two matrices $A, A' \in \mathbb{R}^{n\times n}$ induced by the 2-norm
δ_x	the Dirac delta measure centered at x
$\mathbb{E}_\pi[Y]$	the expected value of a random variable Y with respect to π, given a probability space $(\Omega, \mathcal{F}, \pi)$
$\|f\|_\infty$	the supremum norm $\sup_{x\in D} \|f(x)\|$ of a function $f\colon D \to \mathbb{R}^n$
$\|f\|_{\infty,K}$	the supremum norm $\sup_{x\in K} \|f(x)\|$ of a function $f\colon D \to \mathbb{R}^n$ restricted over a compact set $K \subset D$

$\|f\|_p$	the L^p-norm $\|f\|_p \coloneqq \left(\int_s^T \|f(t)\|^p \, dt\right)^{\frac{1}{p}}$, for $f \in L^p([t_0, T]; \mathbb{R}^n)$
$\|f\|_{W^{1,p}}$	the $W^{1,p}$-norm $\|f\|_{W^{1,p}} \coloneqq \|f\|_p + \left\|\frac{df}{dt}\right\|_p$
I_n	the identity matrix in $\mathbb{R}^{n \times n}$
$L^p([t_0, T]; \mathbb{R}^n)$	the Lebesgue space, for $p \in [1, \infty)$, $\left\{f \colon [t_0, T] \to \mathbb{R}^n \,\middle|\, f \text{ meas. and } \int_s^T \|f(t)\|^p \, dt < \infty\right\}$
$\mathscr{L}$	the Lebesgue σ-algebra on a given topological space
$\mathcal{N}(m, \Sigma)$	the multivariate Gaussian distribution on $\mathbb{R}^n$ with mean vector $m \in \mathbb{R}^n$ and covariance matrix $\Sigma \in \mathbb{R}^{n \times n}$
$ones(m, n)$	the matrix in $\mathbb{R}^{m \times n}$ with all elements equal to 1
$\mathcal{P}(X)$	the space of probability distributions on a given metric space (X, d)
Π	the set of all stationary deterministic policies
Π_t	the set of all time-dependent deterministic policies
$\pi^N \overset{*}{\rightharpoonup} \pi$	π^N weak-$\star$ converges to π, i.e. $\int_X f(x) d\pi^N(x) \to \int_X f(x) d\pi(x), \ \forall f \in C_b(X)$
$\mathbb{R}^+$	the set of non-negative real numbers
$\mathbb{R}^n$	the Euclidean n-dimensional space
$\mathbb{R}^{m \times n}$	the set of $m \times n$-dimensional real-valued matrices
$supp(\pi)$	the support of a measure π
$\|v\|$	the Euclidean norm of a vector $v \in \mathbb{R}^n$
$W^{1,p}([t_0, T]; \mathbb{R}^n)$	the Sobolev space $\left\{f \in L^p([t_0, T]; \mathbb{R}^n) \,\middle|\, \frac{df}{dt} \in L^p([t_0, T]; \mathbb{R}^n)\right\}$, for $p \in [1, \infty)$
$W_1(\pi, \pi')$	the 1-Wasserstein distance between two probability distributions π and π' [141]

Chapter 1

Introduction

1.1 Context and motivations

Reinforcement Learning (RL) [136, 22] is one of the three basic Machine Learning paradigms, together with Supervised Learning and Unsupervised Learning [24]. In RL, an agent interacts with a dynamic, stochastic, and possibly unknown environment, intending to find an action-selection strategy, or policy, to optimize a certain long-term performance. The term *reinforcement* comes from behavioral psychology, where it indicates the mechanism by which a human or animal tends to repeat a certain behavior if it is followed by a reward. Other names for RL are *approximate dynamic programming* [113, 18] and "neuro-dynamic programming", which is also the title of one of the main references for RL [22]. Dynamic programming is a class of methods which we will discuss later. The term *neuro* here stands for neural networks or, more generally, for any function approximator; in the methods presented in the book, function approximators are used to solve large-scale problems. In the intro of that book, we find the following definition of RL: *"The methods of this book allow systems to learn how to make good decisions by observing their own behavior (simulation), and use built-in mechanisms for improving their actions through a reinforcement mechanism"* [22].

One can consider two main RL philosophies: the first one, called *model-based* [77, 144, 32, 71], usually concerns the reconstruction of a model from the data trying to mimic the unknown environment. That model is then used to plan and to compute a suboptimal policy. The second RL philosophy, called *model-free*, employs a direct approximation of the value function and/or a policy based on a dynamic-programming-like algorithm, without using a model to simulate the unknown environment. Model-free methods include the famous Monte Carlo methods [136], Temporal-Difference Learning [134, 125] and Q-Learning [146] and more recent ones [95, 126, 89, 62]. An excellent overview of the two approaches can be found in [136].

The general framework of this thesis is the application of control theory to the RL problem. Let us recall some classical approaches to optimal control problems. *Optimal Control (OC)* [50, 11, 143] dates back to the 1950s, thanks to the work of Richard Bellman [15, 16]. The optimal control problem consists in governing a dynamic system through the influence of a control or input. The main goal is to

find the *"best"* or *optimal* control which minimizes a certain cost functional. This research area has been of great importance since the Cold War and the Space Race between the United States and the Soviet Union [110]. During those years, the American and Russian mathematical schools have developed two distinct methods of solving the same problem in parallel. American scholars, including Richard E. Bellman (1920–1984) and Rufus P. Isaacs (1914–1981), developed the theory of Dynamic Programming. At the same time, on the other side of the world, Lev S. Pontryagin (1908–1988), Vladimir G. Boltyanskii (1925-2019) and the Russian school proposed some necessary conditions for the optimal control problem, known as the Maximum Principle [112].

Dynamic Programming (DP) [15, 16, 21] considers a family of optimal control problems with different initial times and states and looks at the relationship between these problems. The main ingredient is the value function $V(t, x)$, defined as the minimum of the cost functional. It solves a functional equation, called the Bellman equation. In the continuous-time problem, the Bellman equation is a nonlinear partial differential equation of Hamilton-Jacobi type. Once the value function has been obtained, it provides optimal feedback control, i.e. a state-dependent optimal control; this is a valuable property as it makes the control system stable with respect to random disturbances. For this reason, it was possible to apply dynamic programming also to the stochastic variants of the problem [117] and to differential games [69]. *Pontryagin's Maximum Principle* [112, 50] translates into some necessary conditions that the optimal control, the optimal trajectory and a costate variable must satisfy. In the continuous-time case, they consist of a boundary value problem for a system of ordinary differential equations, which, once solved, provides an optimal open-loop control, that is, dependent only on time. Controls of this type cannot adapt to changes in the system or in the presence of small disturbances, and this makes them unstable.

Although the Dynamic Programming approach is preferable from a theoretical point of view as it provides sufficient conditions and a synthesis of optimal feedback control, it has always been very difficult to apply it to realistic problems, since it is very expensive from a computational point of view. It suffers from the so-called *"curse of dimensionality"* (an expression coined by Bellman himself [16]), which means that the computational cost necessary to solve the Bellman equation grows exponentially with respect to the system dimension. In contrast, Pontryagin's method requires solving merely a boundary value problem of linear size in terms of dimensions, which is not a difficult task (see e.g. [5]). This led to the development of suboptimal solution methods to the Bellman equation, known as *approximate dynamic programming* [113, 18], that could overcome the curse of dimensionality. Later, we started calling these methods as Reinforcement Learning [136, 20].

Optimal control and RL are strongly connected [137, 20], as they deal with similar problems; in fact, both can be regarded as *sequential decision problems*, in which one has to make decisions in sequence, trying to optimize not only the immediate rewards but also the future, delayed ones [12]. Recently, Powell proposed a unified framework for all the family of sequential decision problems, including OC and RL [114, 115]. More precisely, RL deals with control problems in which the dynamics of the system is uncertain. Over the years, various techniques have been proposed for working with uncertain models, including adaptive control [7, 82],

dual control [48, 148, 49], robust control [151, 2] and average cost control [122, 23]. In 1992, Sutton and coauthors identified RL with *direct adaptive optimal control* [137]. In recent years, many researchers have tried to link these two research areas, so that each could benefit from the other. For example, Bertsekas compares the two approaches starting from the common aspect of dynamic programming and focusing on discrete-time control systems, which are the natural setting in RL [20]. Recht, on the other hand, offers a tour of Reinforcement Learning from the point of view of continuous-time control [120], following up on some interesting works on continuous-time RL [97, 40, 85, 87]. Finally, Harrison, in his recent lecture notes, presents "a unified treatment of OC and RL, with an emphasis on model-based RL" [65]. This thesis aims to take this connection a step further.

About ten years ago, PILCO was introduced [37, 36], an innovative and disruptive model-based method, from which many subsequent model-based RL methods have been inspired. Rather than exploiting the data to construct a dynamics approximating the partially known environment, PILCO makes use of them to construct a probability distribution (more precisely, a Gaussian Process) on a class of dynamical systems. At each iteration, this distribution is updated to fit the data set. After the model update, PILCO takes the *policy improvement* step which boils down to solving an averaged optimal control problem, where the averaging distribution is the one extrapolated from the data at the previous experiments. That approach has the advantage of considerably reducing the model bias, i.e. the fact that an inaccurate model can produce suboptimal controls, which is one of the main shortcomings of model-based RL [8]. A general, rigorous framework capturing PILCO as well as other Bayesian model-based RL approaches (see, e.g. [31, 56, 73, 32, 71, 144]) has been developed in [98, 99]. In particular, it is important to mention that the framework developed in [98] is closely related to the averaging control framework and Riemann-Stijlties optimal control [92, 23, 105, 122, 153]. Starting from the framework proposed in [98, 99], in this thesis we propose and study an average cost optimal control problem, proving interesting convergence results. Then, we propose a new practical algorithm.

1.2 Contributions

The contribution of the thesis is twofold. In the first part, consisting of Chapters 3 and 4, we analyze from the theoretical point of view an average cost optimal control problem, which is identified with some probabilistic model-based RL (MBRL) methods. The aim is to establish a stronger link between this class of RL methods and the framework introduced in [98]. The model bias issue says that if the model is not accurate, the algorithm will produce a suboptimal policy. Our work instead wants to show that if the model *is accurate*, but not exact, the found policy is almost optimal, and estimate how far it is from the optimal one. In the second, more numerical part, consisting of Chapter 5, we propose a new algorithm for a Linear-Quadratic Regulator (LQR) problem with partially unknown dynamics. The algorithm borrows the structure of a typical model-based RL method and also makes use of Bayesian linear regression, a tool widely used in machine learning. So, while in the first part of the dissertation we use control theory to demonstrate the

convergence of some RL methods, in the second part we use RL tools to solve an optimal control problem.

First part: average cost optimal control problems

In Chapter 3, we consider a finite-horizon optimal control problem of a physical system, namely

$$\begin{cases} \text{minimize } \left\{ \int_{t_0}^{T} \ell(x(t), u(t))dt + h(x(T)) \right\} \\ \text{over } u : [t_0, T] \to U \subset \mathbb{R}^m \text{ measurable such that} \\ \dot{x}(t) = \widehat{f}(x(t), u(t)), \qquad t \in [t_0, T] \\ x(t_0) = x_0 \end{cases} \tag{1.1}$$

where ℓ and h are given, known cost functions and $x_0 \in \mathbb{R}^n$ is a given vector. However, in this context, we will assume that the system dynamics $\widehat{f}$ is *unknown* and the knowledge we have about $\widehat{f}$ is merely represented by a probability distribution π constructed over a space of functions X (with $\widehat{f} \in X$) by using the data available from the physical system. This situation is in accordance with the PILCO [37] setting, which is "not focusing on a single dynamics model", but makes use of "a probabilistic dynamics model, a distribution over all plausible dynamics models that could have generated the observed experience" ([36], pg. 34). Such a modelling setting allows us to define the averaged optimal control problem

$$\begin{cases} \text{minimize } \left\{ \int_X \left[\int_{t_0}^{T} \ell(x_g(t), u(t))\, dt + h(x(T)) \right] d\pi(g) \right\} \\ \text{over } u : [t_0, T] \to U \subset \mathbb{R}^m \text{ measurable such that} \\ \dot{x}_g(t) = g(x_g(t), u(t)), \quad g \in X, \quad t \in [t_0, T], \\ x_g(t_0) = x_0 \end{cases} . \tag{1.2}$$

If indeed the real physical system is driven by the equation

$$\dot{x}(t) = \widehat{f}(x(t), u(t)), \qquad t \in [t_0, T],$$

for a certain function $\widehat{f}$, then, it is reasonable to expect that an increase of the experience will produce a more accurate distribution π over X. This fact can be translated into the assumption that the probability distribution is "close" (in a precise sense that will be specified in the sequel) to the Dirac delta $\delta_{\widehat{f}}$ concentrated at the true dynamics $\widehat{f}$ when enough experience of the environment (here represented by $\widehat{f}$) is gained.

We would like to stress that our goal, in this first part of the dissertation, is not to propose a new algorithm to find an optimal policy but rather to consider a class of existing algorithms and motivate their good performances. In particular, we aim to provide insight into the convergence of Bayesian-like RL algorithms in which a recursive construction of probability measures is carried on (further considerations on the connection with RL are given in Section 3.2). Here, by "convergence", we mean the convergence of the optimal policy obtained by estimating the underlying dynamics using data from the real system (the one constructed in the so-called *policy improvement* step) towards the optimal policy obtained by solving problem (1.1). More precisely, the questions we will tackle are:

1) Is the value function related to the optimal control problem (1.2) close to the value function associated to problem (1.1) when π is close to $\delta_{\widehat{f}}$ w.r.t. the Wasserstein distance (see equation (3.6) for the formal definition)?

2) Under the same assumptions over π, is the optimal control of (1.2) close to the optimal control of (1.1)?

In Chapter 3 we will provide a positive answer to question 1) for a general, nonlinear control system. In Chapter 4 we will deal with a specific problem, the LQR problem, for which we can prove also the convergence of the optimal control.

Second part: a new online algorithm for the LQR problem with partially unknown dynamics

In Chapter 5 we change perspective and attack a numerical problem. We consider an LQR problem where the state matrix $\widehat{A}$ is unknown. Our goal is to control the system by bringing the state towards the origin, and at the same time to reconstruct matrix $\widehat{A}$ based on the observed data. The existing algorithms for this type of problem are based on offline learning: first, the system is observed through one or more simulations, and this allows to reconstruct the dynamics, and only then is the actual control applied. We propose a new algorithm to solve the problem which works online, meaning that it reconstructs the dynamics *and* controls the system in a single run. Our algorithm takes contributions from both OC and RL. In particular, it can be considered a Bayesian RL method, but at the same time, it uses the LQR solution based on the Riccati equation, a standard tool from control theory. Although we still don't have proof of the convergence for this method, it seems to work well on some numerical tests.

1.3 Organization

We now summarize the contents of the main chapters of the dissertation:

Chapter 2: From optimal control to Reinforcement Learning. We give an overview of Optimal Control and Reinforcement Learning, touching on various related topics. To begin with, we present the optimal control problem in various settings, that include continuous and discrete-time problems, the LQR problem, and Markov Decision Processes. We continue talking about dynamic programming, which is first discussed in general and then applied to various problems. Further on, we review some approaches to control problems with uncertain dynamics, such as adaptive control and robust control. Finally, we present Reinforcement Learning, discuss the main similarities and differences with optimal control, and introduce a class of MBRL algorithms that will be analyzed in subsequent chapters.

Chapter 3: An average cost optimal control problem for modeling model-based RL. We present the framework we have worked on, that is an average cost optimal control problem. In particular, in this chapter, we consider a generic nonlinear control system, where the dynamics is uncertain and the belief about the

dynamics is represented by a probability distribution. After having explained in detail how this problem can be identified with some probabilistic model-based RL methods, we prove the convergence of the value function and verify the validity of the result on some numerical tests.

An abridged version of this chapter will appear in the conference proceeding

[109] A. Pesare, M. Palladino, and M. Falcone, *Convergence of the value function in optimal control problems with unknown dynamics*, in 2021 European Control Conference (ECC), IEEE, to appear

Chapter 4: Convergence results for an average cost LQR problem. We consider the framework of Chapter 3 in the specific case of the linear-quadratic problem. For this type of problem, we manage to prove not only the convergence of the value function but also the convergence of the optimal control, using some necessary conditions of the Pontryagin style valid for the average cost problems. When the number of possible dynamics is finite, then, we also prove the convergence of feedback control, through the convergence of the Riccati matrix of an augmented system.

This chapter is based on the results presented in

[108] A. Pesare, M. Palladino, and M. Falcone, *Convergence results for an averaged LQR problem with applications to Reinforcement Learning*, Mathematics of Control, Signals, and Systems, 33 (2021), p. 379–411

Chapter 5: A new online algorithm for an LQR problem with partially unknown dynamics. We present a new online algorithm to solve an LQR problem where the state matrix is unknown. After presenting the problem, we describe the algorithm by discussing some implementation details. Finally, we test it on several numerical tests.

The main contents of this chapter will appear in the conference proceeding

[104] A. Pacifico, A. Pesare, and M. Falcone, *A new algorithm for the LQR problem with partially unknown dynamics*, in 2021 Large-Scale Scientific Computing (LSSC), Springer International Publishing, to appear

Chapter 2

From optimal control to Reinforcement Learning

We begin our dissertation by retracing a historical path that starts from the 1950s with the birth of optimal control and dynamic programming and reaches today, with the development of advanced Reinforcement Learning methods. This chapter is not intended to be exhaustive, but aims to insert the thesis in a broader research context and provide some basic knowledge that will be useful in the following chapters. The structure of the chapter is inspired by [65].

In Section 2.1 we introduce the fundamental concepts of optimal control, both in continuous and in discrete time. In Section 2.2 the main ideas of dynamic programming will be illustrated for several variants of the problem, providing examples of DP algorithms. In Section 2.3 we will discuss some previous attempts to solve optimal control problems in uncertain systems. In the end, in Section 2.4 we will introduce Reinforcement Learning and compare it to optimal control.

2.1 The optimal control problem

2.1.1 Continuous-time optimal control

We will first introduce the deterministic, *continuous-time* optimal control problem [50, 11], which will be our main focus throughout the thesis. Afterwards, we will discuss other typical problem formulations, including the discrete-time and the stochastic variant.

We consider the following control system. The state of the system at time t is denoted by $x(t)$ and lies in $\mathbb{R}^n$. The control variable is $u(t)$ and must be chosen in a closed subset $U \subseteq \mathbb{R}^m$. In the *finite-horizon* formulation, we study the evolution of the system up to a final time $T < \infty$, while $0 \leq t_0 < T$ will denote the initial time. Generally, the state is driven by a system of controlled ordinary differential equations

$$\begin{cases} \dot{x}(t) = f(x(t), u(t)) & t \in [t_0, T] \\ x(t_0) = x_0. \end{cases} \tag{2.1}$$

We refer to equation (2.1) as the *state equation*, whereas function f is called the *dynamics* of the system. We will make the following assumptions on the dynamics:

(A1) $f\colon \mathbb{R}^n \times U \to \mathbb{R}^n$ is continuous and bounded on $B(0,R) \times U$ for all $R > 0$;

(A2) f is Lipschitz continuous with respect to x: $\exists\, L_f > 0$ such that

$$|f(x,u) - f(y,u)| \leq L_f\,|x-y|\,, \qquad \forall\, x,y \in \mathbb{R}^n,\ \forall\, u \in U;$$

These conditions guarantee that the Cauchy problem (2.1) is well-posed. More precisely, the Carathéodory theorem (see e.g. [121]) implies that for every measurable *control function* $u\colon [t_0,T] \to U$ and initial condition $x(t_0) = x_0 \in \mathbb{R}^n$, there exists a unique solution of (2.1), which we will call the *trajectory* associated to $u(\cdot)$.

The goal of the problem is to minimize the *cost functional*

$$J_{t_0,x_0}[u] \coloneqq \int_{t_0}^{T} e^{-\lambda(t-t_0)}\ell(x(t),u(t))dt + e^{-\lambda(T-t_0)}h(x(T))\ , \tag{2.2}$$

over the class of the *admissible controls*

$$\mathcal{U}_{t_0} = \{u : [t_0,T] \to U,\ \text{Lebesgue measurable}\}, \tag{2.3}$$

where ℓ and h are respectively the *running cost* and the *terminal cost*. The term $e^{-\lambda(t-t_0)}$ in (2.2) has an economic interpretation of actualizing a future cost. It translates into mathematical terms the fact that having a gain now is preferable to having the same gain in a future instant. The coefficient $\lambda \geq 0$ is called the *discount factor*. When $\lambda > 0$, $J_{t_0,x_0}[\cdot]$ is called *discounted cost functional*, whereas the undiscounted case is recovered when $\lambda = 0$. We will make the following assumptions on the cost functions:

(A3) $\ell\colon \mathbb{R}^n \times U \to \mathbb{R}$ is continuous and bounded.

(A4) ℓ is Lipschitz continuous with respect to x: $\exists\, L_\ell > 0$ such that

$$|\ell(x,u) - \ell(y,u)| \leq L_\ell\,|x-y|\,, \qquad \forall\, x,y \in \mathbb{R}^n,\ \forall\, u \in U;$$

(A5) h is Lipschitz continuous: $\exists\, L_h$ such that

$$|h(x) - h(y)| \leq L_h\,|x-y|\,, \qquad \forall\, x,y \in \mathbb{R}^n.$$

A control $u^*(\cdot) \in \mathcal{U}_{t_0}$ is said *optimal* if

$$J_{t_0,x_0}[u^*] \leq J_{t_0,x_0}[u] \qquad \forall\, u \in \mathcal{U}_{t_0}. \tag{2.4}$$

The trajectory $x^*(\cdot)$ associated to an optimal control $u^*(\cdot)$ is called an *optimal trajectory* for the problem. The existence and the uniqueness of an optimal control (and the relative optimal trajectory) are non-trivial questions; in general, these properties are not guaranteed for a control system. These topics have been studied in depth, and we refer to the sources [50, 143] for a more complete discussion.

There are two general forms for an optimal control. An *open-loop optimal control* is expressed as a function of time only, $t \mapsto u^*(t)$, but it also depends on the initial state x_0. A *feedback optimal control* or *closed-loop optimal control* is instead expressed as a function of the current state and time, $u^*(t) = \mu(x(t),t)$.

The function $\mu\colon \mathbb{R}^n \times \mathbb{R}^+ \to \mathbb{R}^m$ is called a *policy*. A policy is said *optimal* if the corresponding feedback control is optimal for each initial state $x_0 \in \mathbb{R}^n$. In case of external disturbances or model errors, there is no way to correct an open-loop control. On the other hand, a feedback control can adjust itself in order to deal with disturbances. This makes it more stable and preferable for stochastic control systems. The Maximum Principle approach provides an open-loop optimal control, whereas dynamic programming finds an optimal control in feedback form.

Remark 2.1. There are more general formulations of the problem than this. For instance, one can consider *non-autonomous systems*, in which the dynamics f and the running cost ℓ depend explicitly on time:

$$f = f(t, x, u), \qquad \ell = \ell(t, x, u).$$

State constraints could also be considered, e.g. requiring that the state $x(t)$ always remains in a closed set X of $\mathbb{R}^n$. For the sake of simplicity, we have chosen to consider only the autonomous and unconstrained case.

Remark 2.2. Since each control $u(\cdot) \in \mathcal{U}_{t_0}$ corresponds to a single trajectory $x(\cdot)$, sometimes the optimal control problem is formulated in the following compact form, as an optimization problem in the pair $(x, u)(\cdot)$:

$$\begin{cases} \text{minimize } J_{t_0,x_0}[u] \\ \text{over } (x,u)(\cdot) \text{ such that } u \in \mathcal{U}_{t_0} \text{ and} \\ \dot{x}(t) = f(x(t), u(t)), \qquad t \in [t_0, T], \\ x(t_0) = x_0. \end{cases} \tag{2.5}$$

When the time horizon is infinite, i.e. $T = \infty$, we speak of an *infinite-horizon* optimal control problem. The system dynamics reads

$$\begin{cases} \dot{x}(t) = f(x(t), u(t)) & t \geq 0 \\ x(0) = x_0. \end{cases} \tag{2.6}$$

and the discounted cost functional to be minimized is

$$J_{x_0}[u] \coloneqq \int_0^\infty e^{-\lambda t} \ell(x(t), u(t)) dt \ . \tag{2.7}$$

If the discount factor is positive, i.e. $\lambda > 0$, the integral in (2.7) converges, so the functional is well-defined. Notice that since the state equation (2.6) and the running cost ℓ in (2.7) are time-independent, we can easily compare the costs starting at different times rescaling the costs at time 0:

$$\int_{t_0}^\infty e^{-\lambda t} \ell(x(t), u(t)) dt = e^{-\lambda s} \int_0^\infty e^{-\lambda \tau} \ell(x(s+\tau), u(s+\tau)) d\tau.$$

This is why we consider the cost functional (2.7) dependent on the initial state only and not also on the initial time as in the finite-horizon case.

Moreover, we can also consider a time-independent policy or *stationary policy* $\mu\colon \mathbb{R}^n \to \mathbb{R}^m$. Given a stationary policy μ, one can construct a feedback control $u(t) = \mu(x(t))$. In Section 2.2.4 we will see how dynamic programming methods can provide an optimal policy μ^* for the problem.

2.1.2 The Linear-Quadratic Regulator

The *Linear-Quadratic (LQ) optimal control problem* [81, 4, 50] is a classical problem in control theory, in which the system dynamics is described by a system of linear differential equations and the cost is a quadratic function. Thanks to the simplicity of the model, it is possible to express optimal control in closed form. The solution is provided by a feedback controller, the *Linear-Quadratic Regulator (LQR)*, which we will discuss shortly. Sometimes the problem and the solution are identified, so it is not uncommon for the LQ problem to be called the *LQR problem*. This model is also of great interest in nonlinear systems, where it is applied locally, through a linearization of the dynamics [107, 116].

In the finite-horizon, continuous-time case, the state of the system $x(t) \in \mathbb{R}^n$ evolves according to the following controlled dynamics

$$\begin{cases} \dot{x}(t) = Ax(t) + Bu(t), & t \in [t_0, T] \\ x(t_0) = x_0, \end{cases} \tag{2.8}$$

where $A \in \mathbb{R}^{n\times n}$ and $B \in \mathbb{R}^{n\times m}$. The control function $u(t) \in \mathbb{R}^m$ must be chosen among the admissible controls $\mathcal{U}_{t_0} := \{u : [t_0, T] \to \mathbb{R}^m, \text{ Lebesgue measurable}\}$ to minimize the quadratic cost functional

$$J_{t_0,x_0}[u] \coloneqq \frac{1}{2}\int_{t_0}^{T} \Big(x(t)^T Q x(t) + u(t)^T R u(t)\Big)\, dt + \frac{1}{2}x(T)^T Q_f x(T). \tag{2.9}$$

Some typical assumptions on the cost matrices are the following:

- $Q, Q_f \in \mathbb{R}^{n\times n}$ are symmetric and positive semi-definite;
- $R \in \mathbb{R}^{m\times m}$ is symmetric and positive definite.

Under these assumptions, the LQR provides the optimal control in feedback form [4] as

$$u^*(t) = -R^{-1}B^T P(t)x(t) \qquad \forall\, t \in [t_0, T], \tag{2.10}$$

where $P(t)$ is the unique symmetric solution of a Riccati matrix differential equation

$$\begin{cases} -\dot{P}(t) = A^T P(t) + P(t)A - P(t)BR^{-1}B^T P(t) + Q, & t \in [t_0, T] \\ P(T) = Q_f. \end{cases} \tag{2.11}$$

2.1.3 Discrete-time optimal control

Another common formulation of the optimal control problem is made in discrete time. Although continuous-time systems and differential equations are more appropriate to describe physical phenomena, discrete-time dynamical systems are widely used in several fields of application, such as computer science and economics. Furthermore, a discrete-time system is more suitable for numerical simulations. In reality, the two models are very closely related to each other, and it is easy to obtain a discrete-time system starting from a continuous model through a discretization process; we will provide an example in Section 2.2.4. Conversely, many discrete-time models can be seen as the discretization of a continuous-time system. A good reference for discrete-time optimal control is the monograph by Bertsekas [21].

In a discrete-time system, variables are measured at fixed instants of time $t_k = t_0 + k\Delta t$, where k is the *time index*. We denote the value of the state at time t_k with the subscript k, so that $x_k = x(t_k)$. Analogously, the control or *input* at time k is $u_k = u(t_k)$. The evolution of the dynamical system is described by a difference equation [75]. In *finite-horizon* problems, the system evolves over a finite number N of time steps:

$$\begin{cases} x_{k+1} = f(x_k, u_k) & k = 0, 1, \dots, N-1 \\ x_0 \text{ given }, \end{cases} \tag{2.12}$$

where $x_k \in X \subseteq \mathbb{R}^n$, $u_k \in U \subseteq \mathbb{R}^m$ and $f\colon X \times U \to X$. Depending on the problem, the *state space* X can be either continuous, e.g. all $\mathbb{R}^n$ or a subset of $\mathbb{R}^n$, or discrete, e.g. a set with a finite number of elements. If the system is a discretization of a continuous-time system, the state space will naturally be continuous. Instead, a problem where the state space is a finite set can be thought of as a deterministic version of a Markov chain (see Section 2.1.4).

The integral in the cost functional (2.2) is now a summation over the time index k

$$J_{0,x_0}[u] = \sum_{k=0}^{N-1} \gamma^k \ell(x_k, u_k) + \gamma^N h(x_N), \tag{2.13}$$

where $0 < \gamma \leq 1$ is a *discrete discount factor* that replaces the term $e^{-\lambda t}$ of the continuous-time case. The undiscounted case is recovered by choosing $\gamma = 1$. The cost functional (2.13) depends on the *control sequence* $u = \{u_0, \dots, u_{N-1}\} \in U^N$. Note that for a given initial state x_0 and a control sequence u, the discrete trajectory $\{x_1, \dots, x_N\}$ is uniquely determined by (2.12), so the cost (2.13) is well-defined.

The goal of the problem is to minimize the cost functional (2.13) over all possible control sequences $\{u_0, \dots, u_{N-1}\} \in U^N$. A control sequence u^* is called *optimal* if

$$J_{0,x_0}[u^*] \leq J_{0,x_0}[u] \qquad \forall\, u \in U^N. \tag{2.14}$$

Remark 2.3. More general formulations include the non-autonomous case, taking time-dependent dynamics and running cost,

$$f_k(x, u) = f(t_k, x, u) \;\text{ and }\; \ell_k(x, u) = \ell(t_k, x, u), \qquad \text{for } x \in X,\, u \in U.$$

As in the continuous-time case, we can define an *infinite-horizon* problem, where there is no time horizon. The *discounted cost functional* will be

$$J_{x_0}[u] = \sum_{k=0}^{\infty} \gamma^k\, \ell(x_k, u_k), \tag{2.15}$$

where $0 < \gamma < 1$ and $u = \{u_k\}_{k \geq 0}$ is an infinite control sequence. We look for an infinite control sequence that minimizes (2.15), this would be an open-loop optimal control. Alternatively, we look for a stationary policy $\mu\colon X \to U$ such that the corresponding feedback control $u_k = \mu(x_k)$ is optimal for each $x_0 \in \mathbb{R}^n$.

2.1.4 Markov Decision Processes

Markov Decision Processes (MDPs) [117, 136] are discrete-time *stochastic* control processes. Another name for them is *controlled Markov Chains* [80]. They were introduced in the 1950s by Bellman, who studied them as the stochastic version of discrete-time control systems [17, 16]. MDPs are still very popular nowadays in Reinforcement Learning, for which they constitute one of the most used frameworks.

The state variable of an MDP is driven by a probabilistic dynamics model. Following the same notation of the deterministic state equation (2.12), in the autonomous case we can write it as

$$\begin{cases} x_{k+1} = f(x_k, u_k, \omega_k) & k = 0, 1, \dots, N-1 \\ x_0 \text{ given}, \end{cases} \tag{2.16}$$

where ω_k is the random disturbance at time t_k. We assume that the *disturbance process* $\{\omega_k\}_k$ is a sequence of independent random vectors in $\mathbb{R}$, which incorporates all the stochasticity of the system.

When the state space X is a finite set, the system dynamics is conveniently specified in terms of the transition probabilities between the states. We can think, indeed, that the effect of ω_k in f describes the conditional distribution of the next state x_{k+1}, given the current state x_k and the control u_k and that there exists a function $p\colon X \times X \times U \to [0, 1]$ such that

$$\mathbb{P}\left(x_{k+1} = x' | x_k = x, u_k = u\right) = p(x'|x, u), \qquad \text{for } x, x' \in X,\ u \in U. \tag{2.17}$$

In this formulation the state process is *Markovian*, that is, its evolution does not depend on the whole history of the process, but only on the current state. We denote by

$$h_k \coloneqq (x_0, u_0, x_1, u_1, \dots, x_k, u_k) \tag{2.18}$$

the *history* of the process up to time k. In mathematical terms, the *Markov property* says that

$$p(x_{k+1}|h_k) = p(x_{k+1}|x_k, u_k), \tag{2.19}$$

where $p(x_{k+1}|h_k)$ indicates the conditional distribution of x_{k+1} given the whole process history up to time k. More details on the relationship between the two notations (2.16) and (2.17) can be found in [21, p. 6].

In the stochastic formulation, the cost functional is defined as the expected value of an additive random cost. When u is a deterministic control sequence $\{u_0, \dots, u_{N-1}\}$, we can write it as

$$J_{0,x_0}[u] = \mathbb{E}\left[\sum_{k=0}^{N-1} \gamma^k \ell(x_k, u_k, \omega_k) + \gamma^N h(x_N)\right]. \tag{2.20}$$

Here the expectation $\mathbb{E}[\cdot]$ is formally with respect to the disturbances $\omega_0, \dots \omega_N$, but notice that also the state x_k is random, since it depends on previous disturbances.

In Section 2.1.1 we discussed the two kinds of control: open-loop control and closed-loop control. In the stochastic case, it is convenient to find optimal closed-loop policies, since they can deal better with stochasticity. We can consider both deterministic and stochastic policies.

Definition 2.4. A *deterministic policy* is a function $\mu\colon X \times \mathbb{R}^+ \to U$. Given a policy μ, we can build a *feedback control* $u_k = \mu(x_k, t_k)$. We will denote the space of (time-dependent) deterministic policies by Π_t.

A *stochastic policy*, instead, is a function $\nu\colon X \times \mathbb{R}^+ \to \mathcal{P}(U)$ which takes values in the set $\mathcal{P}(U)$ of all probability distributions on U. When the controller follows a stochastic policy ν, the control at time u_k is random and its conditional distribution is given by

$$u_k | x_k, t_k \sim \nu(x_k, t_k).$$

For the moment, let us consider only deterministic policies. In fact, it can be shown that if there is a stochastic optimal policy, there is also a deterministic optimal policy. When the control is chosen according to a policy $\mu \in \Pi_t$, we will obtain a certain cost given by

$$J_{0,x_0}[\mu] = \mathbb{E}\left[\sum_{k=0}^{N-1} \gamma^k \ell(x_k, \mu(x_k, t_k), \omega_k) + \gamma^N h(x_N)\right]. \tag{2.21}$$

We call (2.21) the *cost of policy* μ when starting from x_0. A policy $\mu^* \in \Pi_t$ is said *optimal*, if it minimizes the cost functional (2.21) for each possible initial state $x_0 \in X$:

$$J_{0,x_0}[\mu^*] \leq J_{0,x_0}[\mu], \qquad \text{for each policy } \mu \in \Pi_t. \tag{2.22}$$

In the *infinite-horizon* problem, the system is generally autonomous, as in the deterministic case. This allows to look for stationary policies $\mu\colon X \to U$, such that the control depends only on the current state, $u_k = \mu(x_k)$. We will denote the space of stationary deterministic policies by Π. The discounted cost functional for a stationary policy $\mu \in \Pi$ takes the following form

$$J_{x_0}[\mu] = \mathbb{E}\left[\sum_{k=0}^{\infty} \gamma^k \, \ell(x_k, \mu(x_k), \omega_k)\right]. \tag{2.23}$$

2.2 Dynamic Programming

Dynamic programming (DP) [16, 21] is one of the two main approaches to solve optimal control problems, together with Pontryagin's approach based on the Maximum Principle [112, 50, 143]. DP was introduced by Bellman in the early 1950s [15, 16]. The basic idea of this method is to consider a whole family of optimal control problems with different initial times and states and look at the relationship between these problems. Bellman formalized this relationship in the so-called *principle of optimality*, which in the original version reads as follows: *"An optimal policy has the property that whatever the initial state and initial decisions are, the remaining decisions must constitute an optimal policy with regard to the state resulting from the first decisions."* [15].

DP is a general method and can be applied and adapted to different problems. However, there are some standard steps, which constitute the structure and main idea of the method. We list these steps below:

1. Define the *value function* of the problem as a function of the initial time and the initial state, in the finite-horizon case, or of the initial state only, in the infinite-horizon case.

2. State precisely the *principle of optimality* and derive the *Bellman equation*, a functional equation of which the value function is a solution.

3. Solve the Bellman equation and prove that the solution found is the value function.

4. Build an *optimal feedback control* from the value function.

DP has a major advantage over Pontryagin's approach, that it provides optimal feedback control. At the same time, this method requires a great deal of computational effort, and it is often impossible to solve the Bellman equation exactly. Bellman was aware of this and coined the expression *"curse of dimensionality"* [16], that is, the computational effort grows exponentially with the dimension of the problem. Numerical methods that solve the Bellman equation in an approximate way are called *approximate dynamic programming* [113, 18] or *Reinforcement Learning* [136, 20].

In this section, we will present the DP approach applied to different types of optimal control problems and discuss some algorithms that solve the Bellman equation exactly, when the state space and the control space are finite. In this case, we speak of *exact dynamic programming*. In particular, we will discuss the two main DP algorithms, *value iteration* and *policy iteration*, in different settings. We will begin by dealing with the discrete-time deterministic problem, the one that Bellman studied first [16]. Next, we will move on to the stochastic version, considering finite MDPs. Finally, we will discuss the generalization of these algorithms to the continuous-time problem.

2.2.1 Dynamic programming in discrete time: finite horizon

Let us consider the deterministic discrete-time problem, according to the formulation of Section 2.1.3. To keep things simple, we will assume that the state space X and the control space U have a finite number of elements. Also, from a numerical point of view, this allows to store all the computed values in lookup tables. When the state space is continuous, the principle of dynamic programming and the Bellman equation still hold, but the space must be discretized to solve them [21, 18]. We will see an example of space discretization in Section 2.2.4. It should be noted that the study of these algorithms for arbitrary state and control spaces is still an active research field [19]. In the following, we will analyze the finite-horizon case and the infinite-horizon case separately. A comprehensive reference for DP in discrete time is the monograph by Bertsekas [21].

We consider a family of problems with different initial times and states. Given an initial time $0 \leq n \leq N-1$ and an initial state $x_n \in X$, we rewrite the dynamics (2.12) as

$$\begin{cases} x_{k+1} = f(x_k, u_k) & k = n, \dots, N-1 \\ x_n \text{ given }, \end{cases} \tag{2.24}$$

and the cost functional (2.13) as

$$J_{n,x_n}[u] = \sum_{k=n}^{N-1} \gamma^{k-n} \ell(x_k, u_k) + \gamma^{N-n} h(x_N), \tag{2.25}$$

where u is a generic control sequence $\{u_n, \dots, u_{N-1}\} \in U^{N-n}$.

We define the *value function* in (n, x_n) as the optimal cost starting from x_n at time n:

$$V(n, x_n) = \min_{u \in U^{N-n}} J_{n,x_n}[u], \qquad \forall\, n = 0, \dots, N-1,\ \forall\, x_n \in X. \tag{2.26}$$

Furthermore, we set the value function at time N equal to the final cost h:

$$V(N, x_N) = h(x_N), \qquad \forall\, x_N \in X. \tag{2.27}$$

Remark 2.5. The term "value" refers to the fact that in equivalent formulations of the problem the goal is to maximize a payoff functional $P = -J$; in that context, the value of a state x represents how much I can earn at most starting from x. In our formulation, it would be more appropriate to call V the *cost function* or *optimal cost function*, but we choose to use the term that is more common in the literature.

The principle of optimality and the Bellman equation

As we will see below, the value function can provide optimal control in feedback form. Let us see how the value function can be computed. The optimization problem defined above satisfies the principle of optimality, which says essentially that the tail of an optimal trajectory is still optimal. We state the principle formally in the following theorem:

Theorem 2.6 (Principle of optimality, [20])**.** *Let $u^* = \{u_0^*, \dots, u_{N-1}^*\} \in U^{N-n}$ be an optimal control sequence, which together with x_0 determines the corresponding optimal trajectory $\{x_1^*, \dots, x_N^*\}$ via the system equation* (2.12)*. Consider the subproblem whereby we start at x_n^* at time n and wish to minimize the cost functional J_{n,x_n^*} in* (2.25) *from time n to time N, over $\{u_n, \dots, u_{N-1}\}$. Then the truncated optimal control sequence $u_n^*, \dots, u_{N-1}^*$ is optimal for this subproblem. In mathematical terms, the value function V satisfies the following equation*

$$V(0, x_0) = \min_{u_0, \dots, u_{k-1}} \left\{ \sum_{k=0}^{n-1} \gamma^k \ell(x_k, u_k) + \gamma^n V(n, x_n) \right\}, \tag{2.28}$$

for every $x_0 \in X$ and $n = 1, \dots, N-1$.

Proof. We prove equation (2.28) for arbitrary $x_0 \in X$ and $n = 1, \dots, N-1$. Let $u^* = \{u_0^*, \dots, u_{N-1}^*\}$ be an optimal control sequence and $\{x_0^*, \dots, x_N^*\}$ the corresponding optimal trajectory, where we defined $x_0^* \coloneqq x_0$. From the optimality of u^*, it holds

$$J_{0,x_0}[u^*] = \min_{u \in U^N} J_{0,x_0}[u] = V(0, x_0). \tag{2.29}$$

Substituting the definition (2.25) of $J_{0,x_0}[u^*]$ in (2.29), we get the formula

$$V(0,x_0) = \sum_{k=0}^{N-1} \gamma^k \ell(x_k^*, u_k^*) + \gamma^N h(x_N^*). \tag{2.30}$$

Now, since the control subsequence $u^*|_{n:N-1} \coloneqq \{u_n^*, \dots, u_{N-1}^*\} \in U^{N-n}$ is an admissible strategy starting from x_n^* at time n and thus competes for the minimization in the definition of the value function

$$V(n,x_n^*) = \min_{u_n,\dots,u_{N-1}} \left\{ \ell(x_n^*, u_n) + \sum_{k=n+1}^{N-1} \gamma^{k-n} \ell(x_k, u_k) + \gamma^{N-n} h(x_N) \right\}, \tag{2.31}$$

we can write, from (2.30),

$$\begin{aligned} V(0,x_0) &= \sum_{k=0}^{n-1} \gamma^k \ell(x_k^*, u_k^*) + \sum_{k=n}^{N-1} \gamma^k \ell(x_k^*, u_k^*) + \gamma^N h(x_N^*) \\ &= \sum_{k=0}^{n-1} \gamma^k \ell(x_k^*, u_k^*) + J_{n,x_n^*}\left[u^*|_{n:N-1}\right] \\ &\geq \sum_{k=0}^{n-1} \gamma^k \ell(x_k^*, u_k^*) + \gamma^n V(n,x_n^*) \\ &\geq \min_{u_0,\dots,u_{k-1}} \left\{ \sum_{k=0}^{n-1} \gamma^k \ell(x_k, u_k) + \gamma^n V(n,x_n) \right\}. \end{aligned} \tag{2.32}$$

This gives the first inequality of (2.28).

To prove the inverse inequality (the one with "$\leq$"), we assume that the opposite statement, that is

$$V(0,x_0) > \min_{u_0,\dots,u_{k-1}} \left\{ \sum_{k=0}^{n-1} \gamma^k \ell(x_k, u_k) + \gamma^n V(n,x_n) \right\}, \tag{2.33}$$

is true, and derive a contradiction. Let $\{\widetilde{u}_0, \dots, \widetilde{u}_{n-1}\}$ be the minimizing control subsequence in the right-hand side of (2.33) and $\{\widetilde{x}_0, \dots, \widetilde{x}_n\}$ be the corresponding trajectory up to time n (with $\widetilde{x}_0 \coloneqq x_0$), so that

$$\sum_{k=0}^{n-1} \gamma^k \ell(\widetilde{x}_k, \widetilde{u}_k) + \gamma^n V(n,\widetilde{x}_n) = \min_{u_0,\dots,u_{k-1}} \left\{ \sum_{k=0}^{n-1} \gamma^k \ell(x_k, u_k) + \gamma^n V(n,x_n) \right\} \tag{2.34}$$

holds true. Besides, let $\{\widetilde{u}_n, \dots, \widetilde{u}_{N-1}\}$ be the optimal control sequence in the time interval $[n, N]$, starting from $\widetilde{x}_n$, and let $\{\widetilde{x}_{n+1}, \dots, \widetilde{x}_N\}$ be the optimal trajectory. This means that

$$V(n,\widetilde{x}_n) = \sum_{k=n}^{N-1} \ell(\widetilde{x}_k, \widetilde{u}_k) + h(\widetilde{x}_N). \tag{2.35}$$

Then, from (2.33), (2.34) and (2.35) we get

$$\begin{aligned} V(0,x_0) &> \sum_{k=0}^{n-1} \gamma^k \ell(\widetilde{x}_k, \widetilde{u}_k) + \gamma^n V(n, \widetilde{x}_n) \\ &= \sum_{k=0}^{n-1} \gamma^k \ell(\widetilde{x}_k, \widetilde{u}_k) + \sum_{k=n}^{N-1} \gamma^k \ell(\widetilde{x}_k, \widetilde{u}_k) + h(\widetilde{x}_N) \\ &= \sum_{k=0}^{N-1} \gamma^k \ell(\widetilde{x}_k, \widetilde{u}_k) + h(\widetilde{x}_N) = J_{0,x_0}[\widetilde{u}], \end{aligned} \tag{2.36}$$

and this creates a contradiction, since the control sequence $\widetilde{u} \coloneqq \{\widetilde{u}_0, \dots, \widetilde{u}_{N-1}\}$ obtained concatenating the two subsequences would have a lower cost than the optimal one, which is impossible by definition of optimal control sequence.

In conclusion, the formula (2.28) is verified. □

Applying the principle of optimality in the interval $[k, k+1]$, it follows that the value function V solves a functional equation, called the *Bellman equation* or *dynamic programming equation*. A proof of the following theorem can be found in [21].

Theorem 2.7. *The value function V is the unique solution of the Bellman equation*

$$\begin{cases} V(k,x_k) = \min\limits_{u_k \in U} \left\{ \ell(x_k,u_k) + \gamma V(k+1, f(x_k,u_k)) \right\}, & \forall\, k = 0, \dots, N-1 \\ V(N,x_N) = h(x_N) \end{cases} \tag{2.37}$$

The DP algorithm

We call DP algorithms those methods that use the principle of optimality and the resulting Bellman equation to solve optimal control problems. When the state space X is finite (and relatively small), the Bellman equation can be solved exactly and the value function can be computed for each $x \in X$ and for each $n = 0, \dots, N$. In this case, we speak of *exact dynamic programming*. In many practical situations, however, the state space is continuous or the number of possible states is very large. For this reason, some approximate methods have been developed to work for general systems.

In the finite state space setting, the *DP algorithm* for the finite-horizon problem (2.24)-(2.25) sets the value function at time N equal to the final cost in each state $x_N \in X$; then proceeds backward in time using the Bellman equation (2.37). Notice that the value function must be computed for each possible state $x_k \in X$ at time k, before proceeding with the previous time step $k-1$. This is why the finiteness hypothesis is crucial in exact dynamic programming.

Once the value functions have been computed for all times and states, we can compute an optimal control sequence $\{u_0^*, \dots, u_{N-1}^*\}$ and the corresponding optimal trajectory $\{x_1^*, \dots, x_N^*\}$ starting from an initial state x_0. For each time step t_k, the optimal control is chosen by minimizing the right-hand side in (2.37):

$$u_k^* \in \arg\min_{u_k \in U} \left\{ \ell(x_k^*, u_k) + \gamma V(k+1, f(x_k^*, u_k)) \right\}. \tag{2.38}$$

If the values $u_k \in U$ where the minimum is attained are more than one, u_k^* can take any of these values. Hence, there may be more than one optimal control sequence.

Notice that the computation of the value function proceeds backward in time, while the second part of the algorithm starts from time 0 and proceeds forward in time. The whole method is summarized in Algorithm 1.

Algorithm 1 DP algorithm for the finite-horizon problem

Require: model (2.24)-(2.25), initial point $x_0 \in X$

Compute the value function solving the iterative Bellman equation (2.37)*:*

Initialize $V(N, x_N) \leftarrow h(x_N)$ for each $x_N \in X$
for k from N-1 to 0 **do**
  for all $x_k \in X$ **do**
    $V(k, x_k) \leftarrow \min_{u_k \in U} \{\ell(x_k, u_k) + \gamma V(k+1, f(x_k, u_k))\}$
  end
end

Construct the optimal control and trajectory from x_0*:*

for k from 0 to N-1 **do**
  $u_k^* \leftarrow \arg\min_{u_k \in U} \{\ell(x_k^*, u_k) + \gamma V(k+1, f(x_k^*, u_k))\}$
  $x_{k+1}^* = f(x_k^*, u_k^*)$
end

2.2.2 Dynamic programming in discrete time: infinite horizon

In the infinite-horizon problem (cf. Section 2.1.3) the system evolves for infinite time. The state dynamics is

$$\begin{cases} x_{k+1} = f(x_k, u_k) & k \geq 0 \\ x_0 \text{ given }, \end{cases} \tag{2.39}$$

and the cost functional for an infinite control sequence $u = \{u_k\}_{k \geq 0}$ is

$$J_{x_0}[u] = \sum_{k=0}^{\infty} \gamma^k \, \ell(x_k, u_k), \tag{2.40}$$

with $0 < \gamma < 1$. The big difference with the previous case is that the solution for the infinite-horizon problem is time-independent: the optimal strategy starting from x_0 at time t_0 is the same as the one starting from x_0 at time t_1. For this reason, we can look for a *stationary policy* $\mu \colon X \to U$. The corresponding *feedback control* will be $u_k = \mu(x_k)$. We can thus write the cost (2.40) as a function of μ:

$$J_{x_0}[\mu] = \sum_{k=0}^{\infty} \gamma^k \, \ell(x_k, \mu(x_k)). \tag{2.41}$$

Let Π be the set of all stationary policies, i.e. the set of all functions $\mu \colon X \to U$. We can formulate the optimal control problem as a minimization task on Π.

We define the *value function* at $x_0 \in X$ as the minimum cost starting from x_0, over all possible policies $\mu \in \Pi$:

$$V(x_0) = \min_{\mu \in \Pi} J_{x_0}[\mu]. \tag{2.42}$$

Notice that the value function in (2.42) depends only on the state and not on the time index, as it was in the finite-horizon case.

The principle of optimality and the Bellman equation

Also in this case the principle of optimality applies and one can derive a Bellman equation (2.44), which is satisfied by the value function V. Unfortunately, the Bellman equation (2.44) has multiple solutions, but we can still characterize V as the unique bounded solution of the equation. We summarize the results for the infinite-horizon case in the following theorem.

Theorem 2.8 ([21, vol. II])**.** *For each time index $n \geq 1$ and for each initial state $x_0 \in X$, the following principle of optimality holds:*

$$V(x_0) = \min_{u_0,\dots,u_{n-1}} \left\{ \sum_{k=0}^{n-1} \gamma^k \ell(x_k, u_k) + \gamma^n V(x_n) \right\}. \tag{2.43}$$

Furthermore, if the running cost ℓ is bounded, the value function V defined in (2.42) *is the unique bounded solution of the Bellman equation*

$$V(x) = \min_{u \in U} \{\ell(x, u) + \gamma V(f(x, u))\} \qquad \forall\, x \in X. \tag{2.44}$$

From Theorem 2.8 we can build the two main DP algorithms for the infinite-horizon problem: *value iteration* and *policy iteration*.

Value iteration

The *value iteration* algorithm [21, vol. II] is essentially the adaptation of the DP Algorithm 1 to the infinite-horizon case. It computes an approximate solution of the Bellman equation 2.44 through a fixed-point method (see Algorithm 2). The algorithm generates a sequence V^k of successive approximations of the value function V. Under rather general hypotheses (e.g. if the running cost ℓ is bounded), the operator on the right-hand side of (2.44), known as the *Bellman operator*, is a contraction and thus the sequence converges to the true value function [21, vol. II]. The speed of convergence depends on the coefficient γ.

Once the value function has been approximated, an optimal policy can be constructed by minimizing the Bellman operator:

$$\mu^*(x) = \arg\min_{u} \{\ell(x, u) + \gamma V(f(x, u))\} \qquad \forall\, x \in X. \tag{2.45}$$

As in the finite-horizon problem, when there is more than one minimizer, $\mu^*(x)$ can take any of these values. As a result, there may be several optimal policies.

Policy iteration

The *policy iteration* algorithm was introduced by Howard in 1960 [68]. The idea is to build a sequence of policies $\mu^k \in \Pi$, each with reduced cost over the preceding one. The sequence is proven to converge to an optimal policy μ^*. The algorithm is

Algorithm 2 Value iteration for the infinite-horizon problem

Require: model (2.39)-(2.40), initial guess V^0, tolerance ε

Approximate the value function solving (2.44)*:*

while $||V^k - V^{k-1}|| \geq \varepsilon$ **do**
  for all $x \in X$ **do**
    $V^{k+1}(x) \leftarrow \min_{u \in U} \left\{ \ell(x,u) + \gamma V^k(f(x,u)) \right\}$
  end
  $k \leftarrow k+1$
end

Construct an optimal policy:

for all $x \in X$ **do**
  $\mu^*(x) \leftarrow \arg\min_{u \in U} \left\{ \ell(x,u) + \gamma V^k(f(x,u)) \right\}$
end

based on two building blocks: *policy evaluation* and *policy improvement*. These two steps are alternated, until convergence is reached.

In the *policy evaluation* algorithm, we freeze a policy μ and compute the cost of the policy $J_{x_0}[\mu]$ defined in (2.41) starting from all possible $x_0 \in X$. The cost of the policy is sometimes called the *value of the policy* μ (cf. Remark 2.5) and is denoted by $V_\mu(x) \coloneqq J_x[\mu]$. Since we have set a policy, the minimum operator in the Bellman equation (2.44) disappears, so the value V_μ satisfies a linear version of (2.44)

$$V_\mu(x) = \ell(x, \mu(x)) + \gamma V_\mu(f(x, \mu(x))), \qquad \forall\, x \in X, \tag{2.46}$$

known as the *Bellman equation for the value* V_μ [136]. In the finite state space case, equation (2.46) is a linear system and can be solved either with direct or iterative methods. The algorithm known as *policy evaluation* uses an iterative method and approximates V_μ through a sequence of functions V_μ^k. The resulting method, outlined in Algorithm 3, turns out to be similar to value iteration (Algorithm 2), but without the minimization procedure.

Algorithm 3 Policy evaluation for the infinite-horizon problem

Require: model (2.39)-(2.40), policy μ, initial guess V_μ^0, tolerance ε

while $||V_\mu^k - V_\mu^{k-1}|| \geq \varepsilon$ **do**
  for all $x \in X$ **do**
    $V_\mu^{k+1}(x) \leftarrow \ell(x, \mu(x)) + \gamma V_\mu^k(f(x, \mu(x)))$
  end
  $k \leftarrow k+1$
end

The *policy improvement* step is based on next Theorem 2.9. In particular, one applies formula (2.47) to obtain a better policy than the current one. We refer to [21, vol. II] for a proof of the theorem.

Theorem 2.9 (Policy improvement theorem)**.** *Let* $\mu \in \Pi$ *be a stationary policy and*

V_μ its value. We define the improved policy μ' as

$$\mu'(x) = \arg\min_{u \in U} \{\ell(x, \mu(x)) + \gamma V_\mu(f(x, \mu(x)))\}, \qquad \forall\, x \in X. \tag{2.47}$$

Then, the policy μ' is an equal or better policy than μ, that is

$$V_{\mu'}(x) \leq V_\mu(x), \qquad \forall\, x \in X. \tag{2.48}$$

Furthermore, the value of the policies V_μ and $V_{\mu'}$ are equal if and only if the two policies are both optimal, and

$$V_\mu(x) \equiv V_{\mu'}(x) \equiv V(x), \qquad \forall\, x \in X. \tag{2.49}$$

The *policy iteration* algorithm starts with an arbitrary policy μ_0, computes V_{μ_0} using policy evaluation and then applies policy improvement, which yields a better policy μ_1 (see Theorem 2.9). The evaluation and improvement steps are repeated alternately, generating a sequence of policies. The following explanatory scheme is taken from Sutton and Barto's book [136]:

$$\mu_0 \xrightarrow{\text{E}} V_{\mu_0} \xrightarrow{\text{I}} \mu_1 \xrightarrow{\text{E}} V_{\mu_1} \xrightarrow{\text{I}} \mu_2 \xrightarrow{\text{E}} \cdots \xrightarrow{\text{I}} \mu^* \xrightarrow{\text{E}} v_*$$

Theorem 2.9 guarantees that each policy is an improvement of the previous one. Furthermore, it gives a stopping criterion: when the values of two subsequent policies V_{μ_k} and $V_{\mu_{k+1}}$ are equal, it means that the value function V has been reached, and the last policy found is optimal. The overall policy iteration algorithm is described in Algorithm 4. In Algorithm 4, we denoted by $V^k \equiv V_{\mu_k}$ the (approximated) value of policy μ_k. In the policy evaluation step, we denoted by $\{V^{k,m}\}_{m \geq 0}$ the sequence which is converging to V^k. Notice that, in the policy evaluation step, the value of the new policy is initialized to the previous one. This speeds up convergence, because the values of two successive policies are usually close to each other.

2.2.3 Dynamic programming for finite MDPs

Optimal control problems for MDPs can be solved by dynamic programming, when a complete description of the system behavior is available. In particular, classical dynamic programming methods require to know the dynamics f and the running cost ℓ or, equivalently, the transition probabilities of the Markov process and the distribution of costs or *rewards*.

It must be said that MDPs are also very popular in Reinforcement Learning (RL), where they constitute one of the most used frameworks. The reason is that they provide a mathematical framework able to capture all features of a sequential decision-making problem, while remaining simple. As we will explain better in Section 2.4, a large portion of RL algorithms is devoted to "optimal control of *incompletely-known* Markov decision processes" (cf. [136, p. 2]), that is, RL methods do not necessarily require knowledge of the system.

In this section, we will consider *finite MDPs*, where both the state space X and the control space U have a finite number of elements. For this class of problems, we will define the notions of value function, state the principle of optimality and present

Algorithm 4 Policy iteration for the infinite-horizon problem

Require: model (2.39)-(2.40), initial guess μ_0, initial guess $V^{0,0}$, tolerance ε

while $||V^k - V^{k-1}|| \geq \varepsilon$ **do**
  Policy evaluation:
  while $||V^{k,m} - V^{k,m-1}|| \geq \varepsilon$ **do**
    for all $x \in X$ **do**
      $V^{k,m+1}(x) \leftarrow \ell(x, \mu_k(x)) + \gamma V^{k,m}(f(x, \mu_k(x)))$
    end
    $m \leftarrow m + 1$
  end
  $V^k \leftarrow V^{k,m}$
  Policy improvement:
  for all $x \in X$ **do**
    $\mu_{k+1}(x) \leftarrow \arg\min_{u \in U} \left\{ \ell(x, u) + \gamma V^k(f(x, u)) \right\}$
  end
  $V^{k+1,0} \leftarrow V^k$
  $k \leftarrow k + 1$
end

the two main algorithms, which are again value iteration and policy iteration, as in the deterministic case. For MDP with continuous state space or control space, function approximators are used [22, 136].

We will consider only the infinite-horizon case, as it is time-independent and the notations are simpler. A comprehensive treatment of the subject can be found in Puterman's monograph [117]. Other references for the subject are [21] and [136].

The Bellman equation for a finite MDP

Recall the stochastic dynamics of an MDP (cf. Section 2.1.4)

$$\begin{cases} x_{k+1} = f(x_k, u_k, \omega_k) & k \geq 0 \\ x_0 \in X \text{ given}. \end{cases} \tag{2.50}$$

We assume that the MDP is *finite*, which means that the state space X and the control space U are both finite. Given a stationary policy $\mu \colon X \to U$, the cost functional is defined as

$$J_{x_0}[\mu] = \mathbb{E}\left[\sum_{k=0}^{\infty} \gamma^k \, \ell(x_k, \mu(x_k), \omega_k) \right], \tag{2.51}$$

with $0 < \gamma < 1$.

Given a policy $\mu \in \Pi$, the *value of the policy* in $x_0 \in X$ is the cost of the policy starting from x_0,

$$V_\mu(x_0) \coloneqq J_{x_0}[\mu].$$

The *value function* or *optimal value function* of the MDP is defined as the optimal cost over all possible policies,

$$V(x_0) \coloneqq \min_{\mu \in \Pi} J_{x_0}[\mu] = \min_{\mu \in \Pi} V_\mu(x_0), \qquad x_0 \in X. \tag{2.52}$$

A policy μ^* is said *optimal*, if $V_{\mu^*} \equiv V$.

The following theorem guarantees that there exists at least one optimal policy and gives the Bellman equation for a finite MDP. We omit the proof, which can be found in [117].

Theorem 2.10. *The following results hold for a finite MDP:*

1. *There exists at least one optimal policy μ^*. If there are more optimal policies, all share the same value $V_{\mu^*} \equiv V$.*
2. *The value function V is the unique solution of the following Bellman equation, which is independent of μ^*:*

$$V(x) = \min_{u \in U} \mathbb{E}\left[\ell(x, u, \omega) + \gamma V(f(x, u, \omega))\right] \tag{2.53}$$

The two techniques that we have illustrated in the deterministic case, value iteration and policy iteration, can also be used to solve finite MDPs, by modifying the formula of the Bellman equation. The adapted versions are reported in Algorithm 5 and Algorithm 6. Similarly, the DP algorithm for the finite-horizon, Algorithm 1, may be readily tweaked and utilized for a finite-horizon MDP.

Algorithm 5 Value iteration for a finite MDP

Require: model (2.50)-(2.51), initial guess V^0, tolerance ε

Approximate the value function solving (2.53)*:*

while $||V^k - V^{k-1}|| \geq \varepsilon$ **do**
  for all $x \in X$ **do**
    $V^{k+1}(x) \leftarrow \min_{u \in U} \mathbb{E}\left[\ell(x, u, \omega) + \gamma V^k(f(x, u, \omega))\right]$
  end
  $k \leftarrow k + 1$
end

Construct an optimal policy:

for all $x \in X$ **do**
  $\mu^*(x) \leftarrow \arg\min_{u \in U} \mathbb{E}\left[\ell(x, u, \omega) + V^k(f(x, u, \omega))\right]$
end

2.2.4 Dynamic programming in continuous time and the Hamilton-Jacobi-Bellman equation

We conclude the presentation of dynamic programming dealing with the continuous-time problem. To begin with, we will state the optimality principle and write the resulting *Hamilton-Jacobi-Bellman (HJB)* equation, a first-order, nonlinear partial differential equation, which is satisfied by the value function. Afterwards, we will recall the definition of viscosity solution and give the characterization of the value function as the unique viscosity solution of the HJB equation. Finally, we will present two numerical methods for continuous-time optimal control problems, based on dynamic programming. In particular, the first method will use a semi-lagrangian scheme to solve the Bellman equation, and can be seen as a value iteration algorithm.

Algorithm 6 Policy iteration for a finite MDP

Require: model (2.50)-(2.51), initial guess μ_0, initial guess $V^{0,0}$, tolerance ε

while $||V^k - V^{k-1}|| \geq \varepsilon$ **do**
  Policy evaluation:
  while $||V^{k,m} - V^{k,m-1}|| \geq \varepsilon$ **do**
    for all $x \in X$ **do**
      $V^{k,m+1}(x) \leftarrow \mathbb{E}\left[\ell(x, \mu_k(x), \omega) + \gamma V^{k,m}(f(x, \mu_k(x), \omega))\right]$
    end
    $m \leftarrow m + 1$
  end
  $V^k \leftarrow V^{k,m}$
  Policy improvement:
  for all $x \in X$ **do**
    $\mu_{k+1}(x) \leftarrow \arg\min_{u \in U} \mathbb{E}\left[\ell(x, u, \omega) + \gamma V^k(f(x, u, \omega))\right]$
  end
  $V^{k+1,0} \leftarrow V^k$
  $k \leftarrow k + 1$
end

The second one will be an adaptation of the policy iteration algorithm in continuous time.

We will limit ourselves again to the infinite-horizon case, but similar steps can be made for the finite-horizon problem. For a comprehensive discussion of the finite-horizon case, we refer to the books [11, 50, 149].

The problem setting has been described in Section 2.1.1. Recall the state equation

$$\begin{cases} \dot{x}(t) = f(x(t), u(t)), & t \geq 0 \\ x(0) = x_0. \end{cases} \tag{2.54}$$

and the cost functional corresponding to the infinite-horizon problem,

$$J_{x_0}[u] \coloneqq \int_0^\infty e^{-\lambda t} \ell(x(t), u(t)) dt \ , \tag{2.55}$$

for $u \in \mathcal{U} = \{u : [0, \infty) \to U, \text{ measurable}\}$.

For each $x_0 \in \mathbb{R}^n$, we define the *value function* as the infimum of the cost functional

$$V(x_0) \coloneqq \inf_{u \in \mathcal{U}} J_{x_0}[u]. \tag{2.56}$$

If a minimizer u^* exists such that

$$J_{x_0}[u^*] \leq J_{x_0}[u] \qquad \forall\, u \in \mathcal{U}, \tag{2.57}$$

we say that u^* is an *optimal control* associated to the optimal control problem (2.54)-(2.55).

The Hamilton-Jacobi-Bellman equation

Bellman's principle of optimality (see Theorem 2.6) also applies in continuous time, in the form of the so-called *Dynamic Programming Principle (DPP)*:

Theorem 2.11 (Dynamic Programming Principle [11, Prop. III.2.5])**.** *Assume hypotheses* $(A_1) - (A_4)$*. Then for all* $x_0 \in \mathbb{R}^n$ *and* $\tau > 0$*, the value function* V *satisfies the following equation:*

$$V(x_0) = \inf_{u \in \mathcal{U}} \left\{ \int_0^\tau e^{-\lambda t} \ell(x(t), u(t)) dt + e^{-\lambda \tau} V(x(\tau)) \right\}, \tag{2.58}$$

where $x(t)$ *is the trajectory starting from* x_0 *and corresponding to an arbitrary control* $u \in \mathcal{U}$*.*

From the DPP, it is possible to derive the Bellman equation, which in the continuous-time case is known as the *Hamilton-Jacobi-Bellman (HJB) equation*. We define the *Hamiltonian* $H \colon \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}$ as

$$H(x, p) \coloneqq \sup_{u \in U} \{-f(x, u) \cdot p - \ell(x, u)\}. \tag{2.59}$$

The HJB equation for the infinite-horizon problem is

$$\lambda V + H(x, \nabla V) = 0 \qquad \text{in } \mathbb{R}^n. \tag{2.60}$$

Equation (2.60) is a first-order, nonlinear partial differential equation; as such, it does not always admit regular solutions, and therefore weak solutions are usually sought. Of particular interest is the so-called *viscosity solution*, according to the definition given by Crandall and Lions in 1983 [34]. Before stating the main result of the section, we recall the definition of a continuous viscosity solution for a general first-order Hamilton-Jacobi equation

$$F(x, u(x), \nabla u(x)) = 0 \qquad x \in \Omega, \tag{2.61}$$

where Ω is an open domain of $\mathbb{R}^n$ and $F \colon \Omega \times \mathbb{R} \times \mathbb{R}^n \to \mathbb{R}$ is a continuous function.

Definition 2.12 (Viscosity solution [11])**.** A function $u \in C(\Omega)$ is a *viscosity subsolution* of (2.61) if, for any $\varphi \in C^1(\Omega)$,

$$F(x_0, u(x_0), \nabla \varphi(x_0)) \leq 0$$

at any local maximum point $x_0 \in \Omega$ of $u - \varphi$. Similarly, $u \in C(\Omega)$ is a *viscosity supersolution* of (2.61) if, for any $\varphi \in C^1(\Omega)$,

$$F(x_1, u(x_1), \nabla \varphi(x_1)) \geq 0$$

at any local minimum point $x_1 \in \Omega$ of $u - \varphi$. Finally, u is a *viscosity solution* of (2.61) if it is simultaneously a viscosity sub- and supersolution.

The following result characterizes the value function among the set $BUC(\mathbb{R}^n)$ of *bounded and uniformly continuous functions.*

Theorem 2.13 ([11])**.** *Assume* $(A_1)-(A_4)$*. Then the value function* V *is the unique viscosiy solution of* (2.60) *in* $BUC(\mathbb{R}^n)$*.*

Remark 2.14 (Evolutive HJB equation for the finite-horizon problem). For completeness, we also provide the finite-horizon version of the HJB equation. In this case, the value function is time-dependent and is defined as

$$V(t_0, x_0) \coloneqq \inf_{u \in \mathcal{U}_{t_0}} J_{t_0,x_0}[u], \tag{2.62}$$

where J_{t_0,x_0} and $\mathcal{U}_{t_0}$ have been defined respectively in (2.2) and (2.3).

The value function (2.62) satisfies the *evolutive HJB equation*

$$\begin{cases} V_t(t,x) + H(x, \nabla V) = 0, & t \in [0,T],\ x \in \mathbb{R}^n, \\ V(T,x) = h(x), & x \in \mathbb{R}^n. \end{cases} \tag{2.63}$$

For more details, we refer to [11, 50].

Remark 2.15 (DP for the LQ problem). For the LQ problem (see Section 2.1.2), the Riccati equation (2.11) can be derived from the HJB equation (2.63) and the two equations are indeed equivalent. The LQR solution (2.10), therefore, is nothing other than the one provided by dynamic programming, when applied to the LQ problem.

Furthermore, one can express the value function and its gradient via the Riccati matrix $P(t)$ (see, e.g. [81], Theorem 3.4)

$$V(t,x) = x^T P(t) x \ \text{ and } \nabla_x V(t,x) = 2P(t)x, \tag{2.64}$$

for all $t \in [t_0, T]$ and $x \in \mathbb{R}^n$.

We will now present two numerical methods for solving the continuous-time optimal control problem.

Value iteration via a semi-Lagrangian scheme

Taking a cue from what happens in the discrete-time case with the value iteration algorithm (see Algorithm 2), we would like to solve the HJB equation (2.60) with an iterative method. A big difference with the discrete-time case is that here the HJB equation can have infinitely many (weak) solutions. As a result, a valid numerical method is required to converge to the unique viscosity solution of the equation.

Several approximation schemes on a fixed grid have been proposed for solving (2.60). Here we will present a value iteration type algorithm based on a semi-Lagrangian approximation, proposed by Falcone in 1987 [44, 45]. For an extensive discussion of semi-Lagrangian schemes, we recommend the monograph by Falcone and Ferretti [46]. The basic idea is to start the construction of the algorithm by a semi-discretization of the problem, to obtain a discrete-time optimal control problem. Fix a timestep $\Delta t > 0$. The discrete-time dynamics is

$$\begin{cases} x_{k+1} = x_k + \Delta t\, f(x_k, u_k) & k \geq 0 \\ x_0 \text{ given }, \end{cases} \tag{2.65}$$

and the cost functional is approximated by

$$J^h_{x_0}[u] = \sum_{k=0}^{\infty} (1-\lambda\Delta t)^k \, \Delta t \, \ell(x_k, u_k) \approx \int_0^{\infty} \ell(x(t), u(t)) \, e^{-\lambda t} dt. \tag{2.66}$$

The time step $0 < \Delta t < 1/\lambda$ is an arbitrary small number and γ has been chosen equal to $1 - \lambda\Delta t$, which is the first-order Taylor expansion for $e^{-\lambda\Delta t}$. The Bellman equation (2.44) for the discretized problem (2.65)-(2.66) reads

$$v_{\Delta t}(x) = \inf_{u \in U} \left\{ \Delta t \, \ell(x,a) + (1-\lambda\Delta t) v_{\Delta t}(x + \Delta t \, f(x,a)) \right\}, \qquad x \in \mathbb{R}^n. \tag{2.67}$$

The value function $v_{\Delta t}$ of the discretized problem has the property to converge, for $\Delta t \to 0^+$, to the value function V of the continuous problem.

Theorem 2.16 ([11, Theorem VI.1.1])**.** *Let the control space $U \subset \mathbb{R}^m$ be a compact set. Then for every compact set $K \subset \mathbb{R}^n$,*

$$\sup_{x \in K} |v_{\Delta t}(x) - V(x)| \to 0 \qquad \text{as } \Delta t \to 0^+.$$

The second step in building the algorithm consists in a space discretization with a spatial step $\Delta x > 0$, to obtain a finite-dimensional problem that can be finally solved by a computer. More precisely, in the simpler case we assume that all the discrete controlled trajectories stay in a bounded domain $\Omega \subset \mathbb{R}^n$ and construct a regular triangulation $\mathcal{G} = \{x_i \colon i \in \mathcal{J}\}$ of Ω with diameter Δx. The numerical method consists in a fixed point method that approximates the solution of (2.67) on the grid nodes x_i, $i \in \mathcal{J}$. The resulting method (see Algorithm 7) is a version of value iteration in continuous time. Here V_i^k represents the values of the numerical value function at a node x_i at the k-th iteration. The point $y_i \coloneqq x_i + \Delta t \, f(x_i, u)$ may not be a grid node, so we use an interpolation operator $\mathcal{I}$ acting on the values of the grid to reconstruct the value of V^k at y_i. The minimization is performed over a discretization $\widetilde{U}$ of the control space U.

When the value function has been approximated at the grid nodes, we can reconstruct an optimal feedback control and the resulting optimal trajectory starting from an initial state $x_0 \in \Omega$, up to a finite time step $N \in \mathbb{N}$. The optimal feedback control can be approximated at *each point* of the domain Ω and not only on the grid nodes; in fact, we can use again the interpolation operator $\mathcal{I}$. The procedure is described in the second part of Algorithm 7.

Policy iteration in continuous time

The other algorithm we present here is a *policy iteration* in continuous time [30]. The original policy iteration algorithm was proposed by Howard [68] and has been discussed in 2.2.2 (see Algorithm 4). In order to apply it to the continuous-time problem, we first discretize in time to obtain the discrete-time Bellman equation (2.67). Then, we solve the discretized problem using a variant of Algorithm 4 on a discrete grid $\mathcal{G} = \{x_i \colon i \in \mathcal{J}\}$.

In the *policy evaluation* step, we fix a policy μ_k. As a result, no search for the minimum in the control space is needed in equation (2.67). The equation becomes

Algorithm 7 Value iteration with a SL scheme for the continuous-time problem

Require: model (2.54)-(2.55), mesh $\mathcal{G}$, Δt, initial guess V^0, tolerance ε

Approximate the value function solving (2.67)*:*

while $||V^k - V^{k-1}|| \geq \varepsilon$ **do**
  for all $x_i \in \mathcal{G}$ **do**
    $V_i^{k+1} \leftarrow \min_{u \in \widetilde{U}} \left\{ \Delta t\, \ell(x_i, u) + (1 - \lambda \Delta t)\, \mathcal{I}[V^k](x_i + \Delta t\, f(x_i, u)) \right\}$
  end
  $k \leftarrow k + 1$
end
$V \leftarrow V^k$

Construct the optimal control and trajectory from x_0*:*

for k from 0 to N-1 **do**
  $u_k^* \leftarrow \arg\min_{u_k \in \widetilde{U}} \{\Delta t\, \ell(x_k^*, u_k) + (1 - \lambda \Delta t)\, \mathcal{I}[V](x_k^* + \Delta t\, f(x_k^*, u_k))\}$
  $x_{k+1}^* = x_k^* + \Delta t\, f(x_k^*, u_k^*)$
end

linear and can be solved as an advection equation. As in the value iteration algorithm, we use an interpolation operator $\mathcal{I}$ to reconstruct the value of V^k outside the grid nodes. In the *policy improvement* step, a new policy is computed, choosing for each state the best control among a finite number of values $\widetilde{U}$.

The constructed sequence of value functions V^k turns out to be monotone decreasing at every node of the grid [3]. The convergence of the method is guaranteed locally. The algorithm is outlined in Algorithm 8.

Finally, let us observe that the DP approach is very flexible and can be applied to other types of problems. For instance, a similar characterization of the value function can be obtained for stochastic optimal control problems, in which case the HJB equation will be of the second order [51]. Another example is that of differential games [69, 53]. The corresponding equation, known as the *Hamilton-Jacobi-Isaacs equation*, is still of first order, but the convexity with respect to ∇V in H is lost [69].

2.3 Optimal control of uncertain systems

Classical optimal control methods, such as those presented in the previous section, assume that the system behavior is described by a well known function f, according to the ODE system

$$\dot{x}(t) = f(x(t), u(t)). \tag{2.68}$$

Unfortunately, there are several applications and practical situations where a precise mathematical description of the system is not available or there is only limited information about it. Or, the system is subject to random disturbances, which make its behavior stochastic. More generally, we can think that the dynamics of the system is described by the following expression

$$\dot{x}(t) = f(x(t), u(t)) + \varepsilon(t), \tag{2.69}$$

where $\varepsilon(t)$ is a random variable that represents the uncertainty of the system.

Algorithm 8 Policy iteration for the continuous-time problem

Require: model (2.54)-(2.55), mesh $\mathcal{G}$, Δt, initial guesses μ_0 and $V^{0,0}$, tolerance ε

while $||V^k - V^{k-1}|| \geq \varepsilon$ **do**
  Policy evaluation:
  while $||V^{k,m} - V^{k,m-1}|| \geq \varepsilon$ **do**
    for all $x_i \in \mathcal{G}$ **do**
      $V_i^{k,m+1} \leftarrow \Delta t\, \ell(x_i, \mu_k(x_i)) + (1 - \lambda \Delta t)\, \mathcal{I}[V^{k,m}](x_i + \Delta t\, f(x_i, \mu_k(x_i)))$
    end
    $m \leftarrow m + 1$
  end
  $V^k \leftarrow V^{k,m}$
  Policy improvement:
  for all $x_i \in \mathcal{G}$ **do**
    $\mu_{k+1}(x_i) \leftarrow \arg\min_{u \in \widetilde{U}} \left\{ \Delta t\, \ell(x_i, u) + (1 - \lambda \Delta t)\, \mathcal{I}[V^k](x_i + \Delta t\, f(x_i, u)) \right\}$
  end
  $V^{k+1,0} \leftarrow V^k$
  $k \leftarrow k + 1$
end

In the context of uncertainty quantification, we speak of two types of uncertainty [39]:

- *Aleatory uncertainty*, also known as statistical uncertainty, is essentially randomness. It is generated by aleatory events, where the values of certain variables differ each time we run the same experiment. This is the case of random phenomena described by stochastic dynamical systems, such as Brownian Motion. It also includes measurement noise on data. This kind of uncertainty is generally inherent in the system and cannot be eliminated.

- *Epistemic uncertainty* is also called systematic uncertainty and is essentially ignorance. It is due to a lack of knowledge of the considered system. It arises when the model neglects certain effects of the phenomenon or when there are few data available. It can be reduced by new experiments and new observations. The concept of epistemic uncertainty is close to the subjective view of Bayesian statistics, where the concept of probability is interpreted as a rational measure of the belief that the agent has on the system [111].

Over time, various techniques have been proposed to deal with model uncertainty. In this section, we will report some of the most relevant ones. Let us mention that each of the methods presented below is more suited to one or the other type of uncertainty, but we avoid further details and from now on we will consider a generic system uncertainty.

2.3.1 System identification

A research area related to uncertain systems and often linked to control is that of system identification. Given a dynamical system, *system identification* [6, 90, 74]

studies how to approximate the system behavior with a mathematical model like in (2.68) from observed time series and prior knowledge of the system. We stress that a mathematical model is always an *approximation* of the real system, which is usually very complex and includes disturbances. Therefore, a differential equation as in (2.68) describes the physical phenomenon only in an approximate way, and can be considered correct only up to a certain level of accuracy. A representation like the one in equation (2.69) is more realistic.

The system identification procedure

System identification is a complex procedure which includes several steps: experiment design, modeling, parameter estimation and model validation.

- The *experiment design* is a fundamental step. It consists in generating data through direct interactions with the system. An optimal design of experiments is necessary for efficiently generating informative data able to capture the system behavior.
- In the *modeling* step, we use the observed data and prior knowledge of the system to build a *model structure*, a family of models which may still contain some unknown coefficients, the model parameters. We say that the model structure is *parametric* when the number of parameters is finite; otherwise, the model structure will be called *non-parametric*.
- The model structure is then *fitted* in the *parameter estimation* step. In other words, we use the observed data to find some numerical values for the model parameters. It usually includes an optimization problem in the parameter space.
- Finally, in the *model validation step*, the model is tested to check if it truly reproduces the system behavior.

We would like to stress that the procedure is strongly influenced by the ultimate goal. System identification and control are often carried out together (see e.g. [79, 140, 47] and Section 2.3.3 on Adaptive Control). However, control is only one of the many applications of system identification. For instance, such a model can be used to obtain insight in physical phenomena and investigate some properties of the system like stability. Other primary applications are simulation and forecasting. In machine learning, the concepts of *supervised learning*, *model learning*, *function approximation* and *regression* are close to system identification. Furthermore, much theory of system identification has been revived by machine learning, identifying old models like *NARX (nonlinear autoregressive exogenous)* with new ones like neural networks [91].

Parametric and non-parametric models

Among the possible model structures, i.e. families of models, we distinguish between parametric and non-parametric models. Parametric approaches specify a set of models completely characterized by a finite number of parameters. Among them,

we recall the vast class of transfer function models [70, 2], which include the finite impulse response (FIR) model, the output-error model, the autoregressive exogenous (ARX) model, the autoregressive moving-average exogenous (ARMAX) model and the Box-Jenkins (BJ) structure. In machine learning, popular choices are linear models, e.g. linear and nonlinear regression [42], as well as deep neural networks themselves [84, 59]. During the parameter estimation step, the parameters are chosen as to minimize a certain error or loss function, for instance the least-squares error. Other classical parameter estimation methods are the Prediction Error Methods (PEM) [90].

On the other hand, in non-parametric methods the family of models is rather general, for instance the set of functions C^n whose first n derivatives are continuous. Among this broad class of functions, a model is built based on the observed data. The complexity of the model and, in some sense, the number of "parameters", grow with increasing observations. For more details on classical non-parametric methods, we refer to [90, Chap. 6]. Popular models in machine learning are Radial Basis Functions (RBF, although there is a parametric variant) [28], Support Vector Regression Machines (SVRM) [43] and decision trees [27]. Some probabilistic non-parametric models based on the Bayesian approach managed to quantify the model uncertainty. These include Bayesian linear regression [26, 123] and Gaussian process regression [119]. Recently, a probabilistic approach to neural networks and deep learning has been proposed through the so-called Bayesian neural networks (BNN) [100, 55]. They have proved to be very efficient, especially with the joint use of network ensembles [64, 10].

2.3.2 Bayesian Linear Regression

Before going on to describe optimal control methods for uncertain systems, in this section we present *Bayesian linear regression (BLR)*, a probabilistic method for solving the classical *linear regression (LR)* problem, which is very popular in machine learning. BLR uses Bayesian formulas to update the parameters estimate at each iteration. For more details on the method, we refer to the many available references [26, 102, 57, 123].

In a regression problem, we consider two variables $x \in \mathbb{R}^n$ and $y \in \mathbb{R}$, that we know, suspect or assume to be somehow related by a functional dependency, that is

$$y = f(x) + \varepsilon, \tag{2.70}$$

where ε is a random noise. The available *data* are observations of the variables, gathered as couples $\mathcal{D} = \{(x_i, y_i)\}_{i=1,\dots,N}$.

Linear regression

The *linear regression* model estimate the relation between x and y using linear functions

$$f_\theta(x) = \theta^T x,$$

depending on a parameter vector $\theta \in \mathbb{R}^n$. We look for a parameter $\theta \in \mathbb{R}^n$ that *"best fits"* the data $\mathcal{D}$, such that

$$y_i \approx \theta^T x_i \qquad \forall\, i = 1, \dots, N.$$

For instance, when x is a 1-dimensional variable, this corresponds to estimate the points (x_i, y_i) through a line on the $\mathbb{R}^2$ plane.

Depending on what is meant by the expression "best fits", the parameter θ can be chosen following different criteria. To give examples, the least squares method [83], Lasso regression [138] and Ridge regression [66] are all different ways to solve the linear regression problems. We will limit ourselves to recalling the least squares method, which is currently the most used and the oldest among them. In fact, its invention dates back to the early 1800s and is credited to Legendre [86] and Gauss [132].

The *(ordinary) least squares (LS) approach* chooses the parameter vector θ by minimizing the sum of squared residuals

$$E(\theta) = \sum_{i=1}^{N} \left| y_i - \theta^T x_i \right|^2 . \tag{2.71}$$

The minimization is equivalent to solving the (rectangular) linear system

$$Y = X^T \theta \tag{2.72}$$

in the *least squares sense*, where we collected all the observed inputs in a matrix $X \in \mathbb{R}^{n \times N}$ and all the observed outputs in a vector $Y \in \mathbb{R}^N$:

$$X = \begin{pmatrix} | & | & & | \\ x_1 & x_2 & \cdots & x_N \\ | & | & & | \end{pmatrix} \quad Y = \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_N \end{pmatrix}. \tag{2.73}$$

The *LS solution* of (2.72) is given by

$$\theta_{LS} = (XX^T)^{-1} XY. \tag{2.74}$$

We point out that the LS approach is rather general and can be applied to any parametric model $y \approx f_\theta(x)$ [83].

Remark 2.17. There are some immediate generalizations of the model. For instance, one can add a constant intercept to the model simply adding a component to the variable x and set it constantly equal to 1 for all observations. Moreover, the model must be linear in the parameters, but can be nonlinear in the single components of x. For example, if $n = 2$ and we write $x_i = (x_{i1}, x_{i2})^T$, we could consider x_{i1}^2 and x_{i2}^2 as further variables, generating an augmented input variable $\widetilde{x}_i = (1, x_{i1}, x_{i2}, x_{i1}^2, x_{i2}^2)^T$. In this case, the model would still be linear with respect to the 5 parameters

$$f_\theta(\widetilde{x}_i) = \theta_0 + \theta_1 x_{i1} + \theta_2 x_{i2} + \theta_3 x_{i1}^2 + \theta_4 x_{i2}^2, \tag{2.75}$$

and we could apply the same formula (2.74) as before.

Bayesian linear regression

Bayesian linear regression (BLR) uses a probabilistic model to explain the relation in (2.70), also taking into account the uncertainty and randomness due to noise. The tool it uses to do this is Bayesian inference.

The available data are again observations $\mathcal{D} = \{(x_i, y_i)\}_{i=1,\dots,N}$. In BLR, we assume that the deviation of the data from the linear model can be described by a Gaussian noise $\varepsilon_i \sim \mathcal{N}(0, \sigma^2)$:

$$y_i = \theta^T x_i + \varepsilon_i, \tag{2.76}$$

where θ is an unknown parameter which must be determined. We will assume that the value of σ is known, though more general formulations apply Bayesian inference on σ as well. The noises ε_i are assumed to be independent and identically distributed, for $i = 1, \dots, N$. Equation (2.76) corresponds to fix a conditional distribution of the random variable y given the value of x and θ,

$$p(y|x, \theta) \sim \mathcal{N}(\theta^T x, \sigma^2). \tag{2.77}$$

This is what in Bayesian inference is called the *likelihood function*. If we assume that the N observations are independent, the global likelihood function can be written as

$$p(Y|X, \theta) = \prod_{i=1}^{N} p(y_i|x_i, \theta) \sim \mathcal{N}(X^T\theta, \sigma^2 I_N), \tag{2.78}$$

where X and Y have been defined in (2.73), and I_N denotes the N-dimensional identity matrix.

The available information on the parameter θ is included in the model through the definition of a *prior distribution*, which we assume to be Gaussian with initial mean $m_0 \in \mathbb{R}^n$ and covariance matrix $\Sigma_0 \in \mathbb{R}^{n\times n}$:

$$\theta \sim \mathcal{N}(m_0, \Sigma_0). \tag{2.79}$$

Bayesian formulas allow to compute the *posterior distribution* of the parameter θ, which is again a Gaussian distribution [123, 26]

$$p(\theta|Y, X) = \frac{p(\theta)p(Y|X, \theta)}{\int_{\mathbb{R}^n} p(\theta')p(Y|X, \theta')d\theta'} \sim \mathcal{N}(m, \Sigma), \tag{2.80}$$

where

$$\Sigma^{-1} = \frac{1}{\sigma^2} XX^T + \Sigma_0^{-1} \quad \text{and} \quad m = \Sigma\left(\frac{1}{\sigma^2} XY + \Sigma_0^{-1} m_0\right). \tag{2.81}$$

From the posterior distribution one can extract a point estimate of the parameter θ, that is the posterior mean

$$\bar{\theta}_{BLR} = \Sigma\left(\frac{1}{\sigma^2} XY + \Sigma_0^{-1} m_0\right) = \left(\frac{1}{\sigma^2} XX^T + \Sigma_0^{-1}\right)^{-1}\left(\frac{1}{\sigma^2} XY + \Sigma_0^{-1} m_0\right). \tag{2.82}$$

However, the advantage of BLR is that it provides a quantification of the uncertainty of this estimate. Finally, we remark that the estimate $\bar{\theta}_{BLR}$ in (2.82) converges to the LS solution θ_{LS} in (2.74), when the noise variance σ goes to 0. In an ideal situation, where there were no noise in the data, BLR would give a result equivalent to the LS method.

2.3.3 Adaptive and dual control

In the following subsections, we will deal with optimal control methods for uncertain systems. Adaptive control [7, 82] is the study of controllers which must *adapt*, or adjust themselves, to handle unknown model uncertainties. In the typical setting, the system dynamics depends on parameters that are initially unknown or time-varying. The first adaptive control problems date back to the early 1950s, when they wanted to design an autonomous aircraft. The dynamical behavior of an aircraft depends on some parameters with strong variations that can occur during a flight. Similarly, rockets undergo significant mass change because a large portion of their mass is made up of fuel, which is ejected during flight to provide propulsive power. Another example is given by robot manipulators, that must deal with large objects with unknown inertial parameters.

The main goal is thus to keep the system performing consistently in the face of uncertainty or unknown variation in its parameters. To achieve its purpose, an adaptive controller integrates optimal control methods with parameter estimation techniques. The plant's or the controller's parameters are updated in real-time so that the control adapts to the changing system. Adaptive control methods are usually divided into two classes: *direct* and *indirect methods*. In the former class, the adjustment rules tell directly how the controller parameters should be updated. Indirect methods, instead, use parameter estimation techniques on the plant and then update the controller according to the new plant model. Hybrid methods are somewhere between direct and indirect methods.

Remark 2.18 (Warning on the terms "direct" and "indirect"). In optimal control a distinction is made between direct and indirect methods, but in that context we refer to different notions. In fact, direct methods are those optimal control methods in which the problem is fully discretized, thus becoming a nonlinear programming problem; on the other hand, indirect methods consider the necessary conditions provided by Pontryagin's maximum principle and solve these equations using a numerical method [118].

In the following, we will briefly discuss the main adaptive control methods.

MRAC: Model-Reference Adaptive Controllers

The Model-Reference Adaptive Controller (MRAC), also referred to as Model-Reference Adaptive System (MRAS), was designed to solve a problem where a reference model is available [7]. The model tells how the process output should respond to the command signal and the goal is to track this reference trajectory as well as possible. The performance of the controller is in fact measured in terms of the reference trajectory. The MRAC method updates directly the controller parameters and is thus considered a direct method.

If we denote by y the output of the controlled system and by y_m the reference output, the controller will try to minimize the model error $e = y - y_m$. The *MIT rule*, which takes its name from the institution where it was first developed in 1960, the Massachusetts Institute of Technology (MIT), tells how to update the controller

parameter θ to reduce the model error:

$$\frac{d\theta}{dt} = -\gamma e \frac{\partial e}{\partial \theta}.$$

The quantity $\frac{\partial e}{\partial \theta}$ is the sensitivity derivative of the error with respect to parameter θ and γ is an adaptation rate. The MIT rule can be regarded as a gradient descent or steepest descent algorithm to minimize the squared error e^2, with step length γ [101].

MIAC: Model Identification Adaptive Controllers

The Model Identification Adaptive Controller (MIAC), in some contexts known as Self-Tuning Regulator (STR), is a type of indirect adaptive controller [7]. In fact, in the MIAC the process parameters are estimated in real-time by linear regression or other parameter estimation methods. Then the controller is designed using the estimated model. When the model parameters are known and the state is fully observed, one can use dynamic programming or other methods to obtain the optimal controller, yet in MIAC only an estimate of the system parameters are available.

Many algorithms don't consider the uncertainties of the estimates and use the estimated parameters as if they were equal to the true parameters. This is called the *certainty equivalence* (CE) principle [129, 9, 94]. CE adaptive control schemes have been studied by many authors, achieving asymptotic convergence results [60]. However, it was soon realized that the CE principle does not always provide enough information to properly estimate the parameters and the computed parameters can converge to incorrect values with positive probability [14]. As a result, the performance might be suboptimal.

Other algorithms take into account also the accuracy of the model predictions to design the controller. These kinds of controllers are called *adaptive cautious controllers* since they usually generate lower magnitude inputs [147].

Dual control

Dual control or adaptive dual control [48, 148, 49] is an important class of adaptive algorithms, to the point that a common classification divides adaptive controllers into *dual controllers* and *non-dual controllers* [148]. It was introduced by Feldbaum in 1960 [48]. A drawback of the previous existing schemes, including MRAC and CE approaches, was that they may not probe the system sufficiently to obtain a good estimate of the parameters. Feldbaum affirmed that an optimal controller should have dual goals. First, it should cautiously control the process as well as possible. Second, it should excite the plant sufficiently for accelerating the parameter estimation process. More excitation leads to better estimation, but also to worse control performance in most systems. Hence, the need to find a compromise between the two features. This conflict somehow anticipated what in Reinforcement Learning is known as the exploration/exploitation trade-off (see Section 2.4).

In practice, Feldbaum [48] proposed dual controllers where the parameter uncertainty was one of the system variables in an augmented state. The formal solution to the optimal dual control problem can be obtained through the use of dynamic

programming, but the equations can neither be solved analytically nor numerically, due to the growing dimension of the underlying space, except with very simple problems (see e.g. [131]). These difficulties in finding the optimal solution led to the formulation of approximate dual controllers, which achieved suboptimal performance, but at least maintained the main dual features. Currently, dual adaptive methods are divided into *optimal dual methods* and *suboptimal dual methods* [147].

2.3.4 Robust control

Robust control [151, 2] is another control approach to deal with system uncertainty. While adaptive controllers adjust themselves when new information is gained on the system behavior, in robust control one assumes to have a priori estimates on the model uncertainty. In particular, robust controllers are designed to achieve good performance and stability across a wide range of models, to handle bounded modeling errors. These errors may be due to the system dynamics being unknown or to some random perturbations in the system. Naturally, the demand for robustness penalizes performance.

The modern robust control theory began in the 1970s, although Horowitz had previously worked in this direction, without using the word "robust" [67]. Many ways to construct a robust controller have been proposed since then. One of the most important techniques is H^∞ *control* [150, 41, 13], which handles the worst-case scenario among the possible models. The name H^∞ stands for the Hardy space of all complex-valued functions of a complex variable, which are analytic and bounded in the open right-half complex plane. In H^∞ methods, the control is chosen by solving a complex optimization problem indeed in the Hardy space. For a comprehensive discussion on H^∞, we recommend the monograph by Zhou [151].

Among H^∞ methods, we recall the so-called *minimax* approach [142, 130, 13], which is based on dynamic game theory. The first player corresponds to the controller, while the second player plays the role of a random perturbation. Not knowing the opponent's future moves, the minimax control considers all possible scenarios and acts by assuming that the opponent will make his best move, which corresponds, from the point of view of the first player, to the worst-case scenario. This formulation has been widely studied both using a dynamic programming approach [130, 13] and using the Pontryagin maximum principle approach [25, 145, 105].

2.3.5 Average cost control

Another problem that has been gaining increasing interest in recent years is the *average cost optimal control problem* [122, 23, 98]. In this framework, the system uncertainty is described by a probability distribution on the set of possible models. The cost functional is defined as an average over all possible trajectories.

In designing a robust control (see Section 2.3.4) one aims for a good performance whatever the model is and this comes at the expense of performance. Here, instead, the goal is to perform well "on average", giving less importance to the unlikely scenarios. Using the same logic, Zuazua [153] studied the controllability of a system in an average way instead of a *simultaneous controllability*. Bettiol and Khalil [23] established some necessary conditions for the optimal control. Interestingly, this

formulation is not far from the concept of cautious control [147].

Recently, Palladino and Murray [98, 99] used an average cost control framework to describe some probabilistic Reinforcement Learning algorithms. In the wake of their work, we provided some rigorous theory to the framework, strengthening the connection between optimal control and Reinforcement Learning [109, 108].

We will describe the average cost problem in detail in Chapter 3 and some related results for the linear-quadratic case in Chapter 4.

2.4 Reinforcement Learning

In the following, we will introduce Reinforcement Learning (Section 2.4.1) and compare it with optimal control theory, stressing the main similarities and differences (Section 2.4.2). Finally, in Section 2.4.3 we will describe a special class of probabilistic model-based RL methods. In Chapter 3 and Chapter 4 we will provide a rigorous mathematical framework able to describe this class of methods.

2.4.1 An introduction to Reinforcement Learning

Reinforcement Learning (RL) [136, 22] is an important branch of Machine Learning aiming to provide satisfactory policies that an agent can easily implement in an uncertain environment [136]. The agent acquires knowledge (*learns*) on the environment from its past experience, which is usually represented by a series of statistical data, but can also learn while interacting with the system.

Markov Decision Processes (MDPs), which have already been introduced in Section 2.1.4, are the ideal framework for RL algorithms, as demonstrated by the fact that most RL methods are set in MDPs. The reason is that they are a simple mathematical formalization of a sequential decision-making problem, easy to work with, but at the same time able to describe many practical control problems. The monographs by Sutton and Barto [136] and by Bertsekas and Tsitsiklis [22], which constitute the two main references for RL, focus on methods for MDPs.

DP methods are a powerful tool and are suitable for a large variety of problems. However, they suffer from the *"curse of dimensionality"* [16] and they are computationally intractable when the state space is large. For this reason, some *approximate dynamic programming* algorithms have been developed, that can solve the Bellman equation in an approximate way, with a lower computation cost. Furthermore, DP methods require to fully know the transition probabilities of the MDP, and this is not always the case. The RL methods were therefore designed to be able to solve an MDP even without knowing precisely the behavior of the environment. In summary, we can say that the methods of RL come to the aid of DP in three cases:

- when a complete model of the system is available, but the state space S is too large;
- when a complete model of the system is not available, but a simulator of the system exists;
- when neither a complete model of the system nor a simulator is available.

RL algorithms are generally classified in two categories: *model-based* methods, which build a predictive model of the environment and use it to construct a controller, and *model-free* methods, which directly learn a policy or a value function by interacting with the environment. Model-free RL includes standard approximate dynamic programming methods such as Monte Carlo methods [136], Temporal-Difference (TD) Learning [134, 125] and Q-Learning [146]. More recent model-free RL algorithms have shown great performances [95, 126, 89, 62], although they are generally quite expensive to train, especially in terms of sample complexity; this often limits their applications to simulated environments. On the contrary, model-based techniques show a higher sample efficiency [37, 144], which results in faster learning. Furthermore, recent algorithms have managed to limit the model-bias phenomenon by using *probabilistic models*, which capture the uncertainty of the learned model [37, 56, 73]. In Bayesian Reinforcement Learning (BRL), the dynamics model is updated when new data are available [58]. Finally, the recently used probabilistic model ensembles [32, 71] allowed model-based methods to achieve the same asymptotic performance as state-of-the-art model-free methods, with higher sample efficiency. Thanks to these features, model-based methods seem to be the most suitable for solving complex real-world problems.

A crucial point in reinforcement learning algorithms is the trade-off between exploration and exploitation [61]. Exploitation consists in acting in an optimal way with respect to the data collected up to that moment. This type of action is safer because it avoids bad decisions, but at the same time it does not allow discovering potential better actions. Exploration, on the other hand, consists in choosing actions that have never been performed or about which there is little information. These actions typically generate lower returns, but may also discover better strategies than the current one. Both exploration and exploitation are important for achieving optimal policy. A good algorithm must therefore balance exploratory and exploitation actions. This trade-off was already present in dual control (see Section 2.3.3).

When the state space is large or even continuous, classical DP methods cannot be used, due to the curse of dimensionality. RL algorithms such as Monte Carlo [136], TD-Learning [134, 125] or Q-Learning [146] also suffer from large dimensions. To solve this issue, we resort to function approximators, i.e. approximate parametric representations of the value function. This way we can describe the value function with relatively few numbers. This idea, known as *"approximation in value space"* or simply *"value function approximation"*, goes back to Shannon's work on computer chess [127]. Some popular choices of function approximators are neural networks [95, 89], decision trees [152], coarse coding [135], Fourier/wavelet bases [78, 139], radial basis functions [136]. Once a parametric representation has been chosen, Monte Carlo, TD-Learning or Q-Learning methods can be applied directly to the low-dimensional parameter. For more details we refer to the books by Bertsekas and Tsitsiklis [22] and by Sutton and Barto [136].

2.4.2 Comparison with optimal control

Optimal control and RL solve almost the same problem, but using different techniques. The primary goal is to control a system to achieve a certain purpose. In this section we will discuss the main overlaps and differences between the two approaches.

First of all, both OC and RL consider a great variety of problems, including continuous-time or discrete-time problems, deterministic or stochastic systems. However, while optimal control has dealt in depth with each of these cases over the years, RL has focused mainly on the MDP framework. As an example, the study of stochastic optimal control resulted in an advanced theory on controlled Markov diffusion processes [51], which, on the other hand, have not been studied much in reinforcement learning.

A strong link between the two approaches is the use of dynamic programming. In fact, we have seen how DP is one of the main methods for solving optimal control problems. The techniques of RL, on the other hand, are strongly rooted in DP, to the point that another name for RL is approximate dynamic programming [113, 18].

One of the main differences between the two fields of study is that optimal control generally assumes complete knowledge of the system or environment, whereas RL methods do not require it. Sutton and Barto wrote about 20 years ago: *"We formalize the problem of reinforcement learning using ideas from dynamical systems theory, specifically, as the optimal control of incompletely-known Markov decision processes"* [136]. The concept of RL has now expanded, but the fact that knowing the model is not necessary remains one of the main advantages. OC methods generally start from the model to construct an optimization problem. This problem is then solved to produce the optimal control. RL, on the contrary, is data-driven and is based on *experience*, that is, direct interaction with the system. The strength of RL is that, even in the presence of a known model, it can be more efficient than the classic optimal control methods. For example, think of Alpha Go, an RL architecture that defeated the world champion of GO [128], a very complex game, whose number of possible board position is of the order of 10^{172}.

An important step in comparing optimal control and RL was taken in 1992 by Sutton, Barto and Williams, who wrote a paper entitled *"Reinforcement Learning is Direct Adaptive Optimal Control"* [137]. In the paper, the authors explain how adaptive control (including MRAC and MIAC algorithms, see Section 2.3.3) generally deals with tracking a reference trajectory, while adaptive methods for optimal control problems had received little attention. Furthermore, almost all adaptive optimal control methods at that moment were *indirect adaptive methods* (see Section 2.3.3 for the definition of direct/indirect adaptive control methods and Remark 2.18 for the difference with optimal control terminology). Reinforcement learning methods, then, could be identified as the *direct* version of adaptive optimal control methods. Today we can still identify, with a certain margin of error, model-free RL with direct adaptive optimal control methods and model-based RL with the indirect methods. It should be noted that over the years, RL has been applied to tracking adaptive problems too [63]. Among RL algorithms, we can also find references to dual control (see Section 2.3.3). For example, Klenske and Hennig compared the Bayesian RL to dual control [76].

In Chapters 3 and 4 we will strengthen this link between OC and RL, showing that some probabilistic model-based RL methods can be described by an optimal control problem with average cost. For a more in-depth comparison of the two fields, we refer to works [20] and [120].

Terminology

If it is true that optimal control and reinforcement learning solve almost the same problem, it must also be said that they use very different notations and terminology. This has caused RL scientists to "re-discover" some optimal control algorithms years after their discovery. For this reason, different names are still used today to indicate the same thing. A non-exhaustive list of examples is given in Table 2.1 (cf. [20, §1.4]).

Table 2.1. Comparison of reinforcement learning and optimal control terminology.

Reinforcement Learning	**Optimal Control**
Agent	Controller or decision maker
Environment	Controlled system or plant
State	State
Action	Control
Reward	(opposite of) Cost
Maximizing the return	Minimizing the cost functional
Model learning	System identification
Prediction	Policy evaluation
Planning	Solving a DP problem with a known mathematical model
Reinforcement Learning	Approximate DP
Deep reinforcement learning	Approximate DP using value and/or policy approximation with deep neural networks
Model-free method	Direct adaptive optimal control
Model-based method	Indirect adaptive optimal control

2.4.3 Probabilistic model-based RL methods

We have already outlined the distinction between model-free methods, which solve the control problem directly interacting with the environment, and model-based methods, which use experience to estimate a model and then use the model to build an optimal control.

A model-based method generally consists of two steps:

1. **Model learning:** learn, estimate a model from experience;
2. **Planning or control:** compute the optimal value function and/or build an optimal policy from the approximate model.

One of the major flaws of model-based methods is the fact that even a small error in the model can result in suboptimal controls. This phenomenon is known as *model bias* and is the reason why for a long time model-free methods have been preferred

to model-based ones [8, 1]. This problem has been reduced with the introduction of probabilistic models to estimate the dynamics, which we can describe as probability distributions on spaces of functions. The use of a probabilistic model made it possible to incorporate model uncertainty into long-term planning, thus reducing model bias, but at the same time achieving outstanding sample efficiency. These features have caused the class of probabilistic model-based RL (*probabilistic MBRL*) methods to become increasingly popular over the past 10 years [77, 144]. Many of them fall within *Bayesian Reinforcement Learning (BRL)*, as probabilistic models are updated over time, using Bayesian statistic formulas. Recently, Moerland et al. provided an exhaustive survey on the topic [96].

Now we will briefly describe some of the most famous methods of this type; for more details, we refer to the original papers. Notice how a method is determined by the choice of an algorithm for each of the two steps, model learning and planning. For the model learning part, this is equivalent to choosing a class of probabilistic model, e.g. Gaussian Processes [37, 73] or deep Bayesian Neural Networks [56, 32, 71]. For the planning procedure, on the other hand, a standard method is to consider a family of policies described by a parameter and apply an optimization method in the policy space, this is known as *policy search* [37, 56, 71]. An alternative is to use *Model Predictive Control (MPC)* [29, 73, 32].

PILCO (2011)

In 2011 Deisenroth and Rasmussen proposed PILCO, a *"data-efficient model-based policy search method"* [38, 37]. The novelty of PILCO compared to previous MBRL algorithms was the possibility to quantify the uncertainty of the model (dealing mainly with deterministic physical systems, we are talking about an *epistemic* uncertainty, that is due to lack of enough data, cf. Section 2.3) and to incorporate it in the planning process. This allowed to reduce the model bias, one of the main problems of MBRL. Furthermore, PILCO showed incredible data-efficiency, meaning that it could learn a task in less time and with less experience than other algorithms.

From a practical point of view, PILCO learns the dynamics using *Gaussian Processes (GP)* as function approximators. As the agent acquires new data through experience, the model is updated with closed Bayesian formulas. For more details on GP regression we refer to the monograph by Rasmussen and Williams [119]. The planning task is solved through an optimization in the policy space. As a policy space, PILCO considers a parameterized family of policies. Two examples are a family of linear policies or a nonlinear network of Radial Basis Functions (RBF) [103]. Taking up the typical structure of a policy iteration, planning is divided into two steps: policy evaluation and policy improvement (cf. Section 2.2.2). In the policy evaluation step, the cost of the policy is calculated taking into account the uncertainty of the model. In particular, the expectation of the cost with respect to the probabilistic model is considered. The policy improvement step, on the other hand, consists of a steepest descent step in the policy parameter space, with a classic gradient descent method. The algorithm is explained in detail in Deisenroth's PhD Thesis [36]. We outline the procedure in Algorithm 9.

Algorithm 9 PILCO [36]

Require: parametrized family of policies
Set policy parameters to random ▷ policy initialization
repeat
    Execute system with policy ▷ interaction
    Record collected experience
    Learn probabilistic (GP) dynamics model ▷ model learning
    repeat ▷ planning
        Simulate system with policy
        Compute expected cost of the policy ▷ policy evaluation
        Improve policy ▷ policy improvement
    until policy parameters converge
until task learned

DeepPILCO (2016)

PILCO worked well with simple and low-dimensional problems, but relied on GPs, whose update requires a great computational cost. This prohibited its application to problems that require a large number of trials to be solved. Some years after its introduction, Gal et al. [56] proposed an improvement of the algorithm, called Deep PILCO, by replacing Gaussian processes with *Bayesian neural networks (BNNs)* [93], the probabilistic equivalent of neural networks (NNs). We point out that one of the authors, Rasmussen, was also an author of PILCO. Deep PILCO required lower time complexity and obtained lower computational costs than PILCO on some standard benchmarks.

GP-MPC (2018)

Subsequently, Kamthe and Deisenroth [73] (again, one of them was among the authors of PILCO) proposed a new algorithm, called GP-MPC, that coupled the use of Gaussian Processes with *Model Predictive Control (MPC)* [29], a popular method for solving optimal control problems. Their algorithm therefore still uses GPs for model learning, whereas it adopts a probabilistic version of MPC for the planning task. GP-MPC was able to deal with state and control constraints and turned out to be extremely data-efficient.

PETS (2018)

A limitation of the algorithms presented so far was that, despite being very efficient in terms of data, they did not achieve the same asymptotic performance of the state-of-the-art model-free RL algorithms. In 2018 Chua et al. [32] proposed a new MBRL algorithm called *probabilistic ensembles with trajectory sampling (PETS)*, which narrowed this performance gap. It achieved indeed results that are comparable with those of state-of-the-art model-free methods in terms of asymptotic performance, while requiring significantly fewer trials.

To learn the dynamics model, PETS use not a single BNN, but rather an ensemble of BNNs. The authors explain in the paper that the use of a "probabilistic network"

can capture aleatoric uncertainty, whereas the use of "ensembles" captures epistemic uncertainty. The planning task is performed using MPC together with a special *trajectory sampling* propagation technique.

Table 2.2. Comparison of probabilistic MBRL metods.

Algorithm	Model Learning	Planning
PILCO [38]	GP	policy search
Deep PILCO [56]	BNN	policy search
GP-MPC [73]	GP	MPC
PETS [32]	ensemble of BNNs	MPC
MBPO [71]	ensemble of BNNs	SAC [62]

MBPO (2019)

Finally, in 2019 Janner et al. [71] proposed *model-based policy optimization (MBPO)*, a new efficient MBRL algorithm. They use an ensemble of BNNs as probabilistic model, as in PETS. To find the control, they perform an optimization in the policy space using *soft-actor critic (SAC)* [62], a popular model-free method. MBPO matches the asymptotic performance of the best model-free algorithms, surpassing the sample efficiency of prior model-based methods.

In Table 2.2 we summarize the tools used respectively for model learning and for planning by the methods presented. It should be noted that PETS [32] and MBPO [71] are currently the state of the art among model-based RL methods. In the next chapters, we will try to capture the behavior of these methods through a rigorous mathematical description. This will allow us to make general considerations on the convergence of the methods (cf. Section 3.2).

Chapter 3

An average cost optimal control problem for modeling model-based Reinforcement Learning

In Section 2.4.3 we have introduced a class of model-based RL algorithms which use experience to build a probabilistic model of the environment and then use the model to construct a control. In this chapter, we present a mathematical framework that allows us to better describe what these algorithms do and, consequently, to study their properties. We consider a nonlinear control system in which the dynamics is only partially known, and we assume that the belief on the dynamics that an agent has is represented by a probability distribution π on a space of functions. The task is thus written as an optimal control problem with averaged cost. The cost functional to be minimized is indeed calculated averaging the costs of the trajectories generated by all the possible dynamics.

Average cost control (cf. Section 2.3.5), together with averaged controllability [153], is a problem that has been studied by several researchers in recent years [92, 23, 98]. Palladino and Murray [98, 99] were the first to link this framework to a class of RL algorithms. We have followed their work by establishing strong results, that will be presented in this and the next chapter. A major difference with respect to their works is that they considered a time-dependent distribution $\pi(t)$, which corresponds to an online model learning, whereas we are working with a stationary π, meaning that the learning is offline. However, we allow the policy to be updated after each episode and this generates a sequence of policies $\{\pi^N\}_N$.

A characterizing aspect of our framework, compared for example to stochastic systems, is that the true dynamic is only one and is deterministic, but it is unknown to the agent. The probability distribution describes the agent's belief in the dynamics, following a Bayesian approach to probability [35]. In other words, the uncertainty that the agent has on the system is epistemic rather than aleatory (cf. Section 1.3). This aspect had already been highlighted in a work by Zuazua [153].

The main result of this chapter concerns a *convergence result of the value function* V_π of an averaged (with respect to a probability measure π) optimal control problem

to the "true value function" V. Here, by true value function we mean the one defined by the optimal control governed by the *true*, underlying dynamics. Roughly speaking, the main result can be stated as follows: *the value function V_π is close to V as soon as π provides an accurate representation of the true, underlying dynamics.* Here we will focus our attention on a general, nonlinear optimal control problem over a finite horizon, under globally Lipschitz assumptions on the costs and the controlled dynamics. Stronger results will be discussed in next chapters under further assumptions. The results presented here have been published in [109].

In Section 3.1, we will introduce the average cost control framework, whereas in Section 3.2 we will explain in detail the connection with model-based RL. Later, in Section 3.3 we will state and prove the main result. Finally, Section 3.4 is devoted to some numerical tests.

3.1 Problem formulation

In this section, we will recall the optimal control problem in the continuous-time setting and introduce an optimal control problem with an average cost.

3.1.1 Problem A: a classical optimal control problem

Let us consider a classical finite horizon optimal control problem (cf. Section 2.1.1 and Section 2.2.4), which we will call *Problem A*. For $0 \leq t_0 < T$, let us consider the controlled dynamics

$$\begin{cases} \dot{x}(t) = \widehat{f}(x(t), u(t)) & t \in [t_0, T] \\ x(t_0) = x_0, \end{cases} \tag{3.1}$$

where the nonlinear dynamics $\widehat{f} : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^n$ is continuous in the pair (x, u) and Lipschitz continuous with respect to x, uniformly with respect to u. Those conditions guarantee that the Cauchy problem (3.1) is well-posed, in the sense that for every measurable control u and initial condition $x(t_0) = x_0 \in \mathbb{R}^n$, there exists a unique solution of (3.1).

The goal is to minimize the cost functional

$$J_{t_0,x_0}[u] \coloneqq \int_{t_0}^{t} \ell(x(t), u(t))dt + h(x(T)) \ , \tag{3.2}$$

over the class of the *admissible controls* $\mathcal{U}_{t_0} = \{u \colon [t_0, T] \to U, \text{ Lebesgue measurable}\}$, where U is a closed subset of $\mathbb{R}^m$ and ℓ and h are respectively the running cost and the terminal cost, which we require to be Lipschitz continuous with respect to x. Notice that we are considering the non-discounted case, i.e. the discount factor λ in (2.2) is equal to 0. This choice makes sense for the finite horizon case if T is not very large, since the effect of the discount factor is not relevant in that case and the integral converges anyway. For each $(t_0, x_0) \in [0, T] \times \mathbb{R}^n$, the *value function* and the corresponding *optimal control* associated to the optimal control problem (3.1)-(3.2) are respectively defined as

$$V(t_0, x_0) \coloneqq \inf_{u \in \mathcal{U}_{t_0}} J_{t_0,x_0}[u] \quad \text{and} \quad u^*(t_0, x_0) \coloneqq \arg\min_{u \in \mathcal{U}_{t_0}} J_{t_0,x_0}[u] \, .$$

3.1.2 Problem B: an optimal control problem with uncertain dynamics

Let us now introduce another control problem in which the real dynamics $\widehat{f}$ is *unknown*, meaning that one has merely a partial knowledge on $\widehat{f}$. Such a model uncertainty is captured by a probability distribution on a space of functions $\mathcal{X}$ (which $\widehat{f}$ belongs to). More precisely, $\mathcal{X}$ is a compact subset of $C(\mathbb{R}^n \times U; \mathbb{R}^n)$ with respect to the $||\cdot||_\infty$ norm (the Arzelà-Ascoli Theorem provides necessary and sufficient conditions for the set $\mathcal{X}$ being compact [124]). For $0 \leq t_0 < T$ and every $g \in \mathcal{X}$, let us define the dynamical system

$$\begin{cases} \dot{x}_g(t) = g(x_g(t), u(t)) & t \in [t_0, T] \\ x_g(t_0) = x_0. \end{cases} \tag{3.3}$$

Given a probability distribution $\pi \in \mathcal{P}(\mathcal{X})$ over $\mathcal{X}$, one can define a cost functional for Problem B:

$$\begin{aligned} J_{\pi,t_0,x_0}[u] \coloneqq & \mathbb{E}_\pi \left[\int_{t_0}^{T} \ell(x_g(t), u(t)) dt + h(x(T)) \right] \\ = & \int_{\mathcal{X}} \left[\int_{t_0}^{T} \ell(x_g(t), u(t))\, dt + h(x(T)) \right] d\pi(g). \end{aligned} \tag{3.4}$$

Note that, even if there is a different trajectory $x_g(t)$ for each $g \in \mathcal{X}$, the task concerns to look for a single control u to be applied to every system dynamics $g \in \mathcal{X}$. The notions of value function and optimal control are analogous to those given for Problem A:

$$V_\pi(t_0, x_0) \coloneqq \inf_{u \in \mathcal{U}_{t_0}} J_{\pi,t_0,x_0}[u] \quad \text{and} \quad u^*_\pi(t_0, x_0) \coloneqq \arg\min_{u \in \mathcal{U}_{t_0}} J_{\pi,t_0,x_0}[u]\,. \tag{3.5}$$

3.2 Connection to model-based RL

There are several model-based RL (MBRL) methods that rely on the design of a probabilistic model of the environment. We introduced this class of methods in Section 2.4.3.

In the following, we focus on a generic algorithm of this kind, which consists of two steps:

Step 1. Model learning: learn, estimate a *probabilistic* model from experience;

Step 2. Planning or control: compute a control which is optimal *with respect to the approximate model*.

These two steps are generally repeated until the method produces a control with satisfactory performance (cf. Section 2.4.3). The goodness of the method is then measured merely on the basis of the obtained results and on heuristic reasoning, not on the basis of proven theoretical properties of convergence. For this reason, we tried to analyze the method from the point of view of optimal control theory, in order to have a solid foundation on which to build a theory of convergence.

In Step 1, we can identify a probabilistic model of the environment with a probability distribution on the space of possible dynamics. For instance, PILCO algorithm [37] builds the model as a Gaussian Process, which is a probability distribution on the space of continuous functions $C(\mathbb{R}^n \times U; \mathbb{R}^n)$ (cf. [36, p. 34]). Thus, in our framework, the probability distribution $\pi \in \mathcal{P}(\mathcal{X})$ represents the dynamics model that the agent builds upon the data collected while exploring the partially unknown environment, where $\mathcal{X}$ is a general functional space. This distribution is updated as new experience is gained, and this generally happens after each episode. In our analysis, the probability distribution $\pi^N \in \mathcal{P}(\mathcal{X})$ will indicate the one updated after the N-th episode. It is reasonable to expect that an increase of the experience of the environment (here represented by $\widehat{f}$) will produce a more accurate distribution over the space of possible dynamics. This fact can be translated into the assumption that the probability distribution π^N get "closer" (in a precise sense that will be specified in the sequel) to the Dirac delta $\delta_{\widehat{f}} \in \mathcal{P}(\mathcal{X})$ as N increases.

Remark 3.1. It is worth pointing out that the theory developed here works both for non-parametric models (e.g. Gaussian processes [37, 73]) and for parametric models (e.g. deep neural networks ([56, 32, 71])). In the latter case, we are assuming that the support of π is a family of functions $\{f_\lambda\}_{\lambda \in \mathbb{R}^d}$ described by a parameter $\lambda \in \mathbb{R}^d$, with the dimension d arbitrarily large. π can thus be seen as a probability distribution on the parameter space $\mathbb{R}^d$ and is then easier to work with (see, for instance, the numerical tests in Section 3.4).

In Step 2 the algorithm solves a control problem with respect to the approximate model. An important feature in probabilistic MBRL is that *"model uncertainty needs to be incorporated into planning and policy evaluation"* [37]. This translates into the fact that the algorithm does not solve a classical Problem A, but a problem in which the cost functional is averaged over all the possible dynamics, i.e. a problem of type B (see Section 3.1). The policy evaluation step (cf. Section 2.2) of PILCO, as an example, computes an averaged cost functional as in (3.4).

We would like to stress that our goal is not to propose a new algorithm to find an optimal policy, but rather to consider a class of existing algorithms and motivate their good performances. In particular, we aim to provide an insight into the convergence of Bayesian-like RL algorithms in which a recursive construction of probability measures is carried out. Here, by "convergence", we mean convergence of the optimal policy obtained by estimating the underlying dynamics using data from the real system, i.e. the optimal control for Problem B u^*_π, towards the optimal policy u^* obtained by solving problem A.

In our analysis, we make two assumptions on the algorithm:

(i) We assume that the probability distribution π^N converges to $\delta_{\widehat{f}}$ as N goes to infinity.

(ii) We assume that in Step 2 the algorithm in question is able to find the *optimal* control, meaning here that the control is optimal *with respect to the averaged cost functional*, i.e. it can find $u^*_\pi(t_0, x_0)$ in (3.5).

Clearly, as the accuracy of π increases, one should expect that the value function of Problem B is close (in a sense that will be made precise) to the value function of Problem A. In particular, we will investigate the following questions:

(A) Is the value function of Problem B V_π close to the value function of Problem A V, when π is close to $\delta_{\widehat{f}}$ w.r.t. the Wasserstein distance (see (3.6) for the formal definition)?

(B) If $\pi^N \to \delta_{\widehat{f}}$, then is it true that $V_{\pi^N} \to V$?

(C) If $\pi^N \to \delta_{\widehat{f}}$, then is it true that $u^*_{\pi^N} \to u^*$?

In this chapter, we will consider a general, nonlinear control system and provide positive answers to question (A) and (B). In the next chapters we will deal with more specific settings, in which we are able to prove also the convergence of the optimal control, i.e. question (C).

3.3 Convergence of the value function

In this section, we will state and prove a convergence result that is valid for a general family of probability distributions. First, we recall the definition of the Wasserstein distance (cf. [141]):

Definition 3.2. Let $\pi, \pi' \in \mathcal{P}(X)$ be two probability distributions on a compact metric space (X, d). The *1-Wasserstein distance* between them is defined as

$$W_1(\pi, \pi') \coloneqq \inf_{\gamma \in \Gamma(\pi, \pi')} \int_{X \times X} d(x, y)\, d\gamma(x, y) \ , \tag{3.6}$$

where $\Gamma(\pi, \pi')$ is the collection of all probability measures on $X \times X$ having π and π' as marginals, x and y are generic elements in X and the symbol $d\gamma$ indicates that the integral is with respect to the measure γ.

Now we can state the result:

Theorem 3.3 (Lipschitz estimate for the value function w.r.t. π)**.** *Let us consider two Problems of type B as described in Section 3.1.2, one with distribution $\pi^N \in \mathcal{P}(\mathcal{X})$ and the other with distribution $\pi^\infty \in \mathcal{P}(\mathcal{X})$. We make the further assumptions:*

($H1$) *There exists a constant $L_f > 0$ such that*

$$|f(x, u) - f(y, u)| \leq L_f |x - y|$$

for each $f \in supp(\pi^\infty)$, $x, y \in \mathbb{R}^n$, $u \in U$;

($H2$) *The two cost functions ℓ and h are Lipschitz continuous in the first argument with constants respectively L_ℓ and L_h.*

Then, the following estimate holds:

$$||V_{\pi^N} - V_{\pi^\infty}||_\infty \leq C(L_f, L_\ell, L_h, T)\, W_1(\pi^N, \pi^\infty) \ , \tag{3.7}$$

where $C(L_f, L_\ell, L_h, T)$ is a constant depending only on the Lipschitz constants and on T, and $W_1(\pi^N, \pi^\infty)$ is the 1-Wasserstein distance (3.6) *constructed on the space $(\mathcal{X}, \|\cdot\|_\infty)$.*

Proof. We divide the proof in three steps.

STEP 1: Distance between two single trajectories.
Fix two dynamics $g \in \mathcal{X}$ and $f \in supp(\pi^\infty)$, an initial condition $x(t_0) = x_0$ with $s \in [0, T]$ and $x_0 \in \mathbb{R}^n$ and a control $u \in \mathcal{U}_{t_0}$. We estimate how far is $x_g(t)$ from $x_f(t)$, using Gronwall's Lemma, for each $t \in [t_0, T]$.
Recall that $x_f(t)$ and $x_g(t)$ are solutions of the dynamical systems (3.3):

$$x_f(t) = x_0 + \int_{t_0}^{t} f(x_f(\tau), u(\tau))\, d\tau$$
$$x_g(t) = x_0 + \int_{t_0}^{t} g(x_g(\tau), u(\tau))\, d\tau$$

We have the following estimate:

$$\begin{aligned}
|x_g(t) - x_f(t)| &\leq \int_{t_0}^{t} |g(x_g(\tau), u(\tau)) - f(x_f(\tau), u(\tau))|\, d\tau \\
&\leq \int_{t_0}^{t} |g(x_g(\tau), u(\tau)) - f(x_g(\tau), u(\tau))|\, d\tau \\
&\quad + \int_{t_0}^{t} |f(x_g(\tau), u(\tau)) - f(x_f(\tau), u(\tau))|\, d\tau \\
&\leq (t - s)\ \|f - g\|_\infty + L_f \int_{t_0}^{t} |x_g(\tau) - x_f(\tau)|\, d\tau\,.
\end{aligned}$$

By Gronwall's Lemma,

$$|x_g(t) - x_f(t)| \leq (t - s)\ \|f - g\|_\infty\ e^{L_f(t-s)} \leq t\ \|f - g\|_\infty\ e^{L_f t}. \tag{3.8}$$

STEP 2: Cost difference between two probability distributions.
Fix an initial condition $x(t_0) = x_0$ with $s \in [0, T]$ and $x_0 \in \mathbb{R}^n$ and a control $u \in \mathcal{U}_{t_0}$. We estimate how far is $J_{\pi^N, t_0, x_0}[u]$ from $J_{\pi^\infty, t_0, x_0}[u]$. To simplify the notation, we will write

$$J_{\pi^N}[u] = J_{\pi^N, t_0, x_0}[u] \quad \text{and} \quad J_{\pi^\infty}[u] = J_{\pi^\infty, t_0, x_0}[u].$$

For each $g \in \mathcal{X}$ and $f \in supp(\pi^\infty)$ it holds

$$\begin{aligned}
|\ell(x_g(t), u(t)) - \ell(x_f(t), u(t))| &\leq L_\ell\, |x_g(t) - x_f(t)| \\
&\leq L_\ell\, t\ \|f - g\|_\infty\ e^{L_f t}
\end{aligned} \tag{3.9}$$

and

$$|h(x_g(T)) - h(x(T))| \leq L_h\, T\ \|f - g\|_\infty\ e^{L_f T}. \tag{3.10}$$

As a property of the Wasserstein distance W_1, there exists (see Theorem 4.1 on [141]) a distribution γ^* on $\mathcal{X} \times \mathcal{X}$ with marginal distributions π^N and π^∞ such that

$$W_1(\pi^N, \pi^\infty) = \int_{\mathcal{X}\times\mathcal{X}} \|g - f\|_\infty\, d\gamma^*(g, f). \tag{3.11}$$

Let $(g(x), f(x))_{x\in\mathbb{R}^n}$ be a continuous process which has γ^* as distribution. For each event $\omega \in \Omega$, we consider the realization of the process (g^ω, f^ω).

Let us sum up over the $g \in supp(\pi^N)$ and the $f \in supp(\pi^\infty)$:

$$\begin{aligned} J_{\pi^N}[u] - J_{\pi^\infty}[u] &= \int_{\mathcal{X}} \Big[\int_{t_0}^{t} \ell(x_g(t), u(t))\, dt + h(x_g(T)) \Big] d\pi^N(g) \\ &\qquad - \int_{\mathcal{X}} \Big[\int_{t_0}^{t} \ell(x_f(t), u(t)) dt + h(x_f(T)) \Big] d\pi^\infty(f) \\ &= \int_{\mathcal{X}\times\mathcal{X}} \Big[\int_{t_0}^{t} (\ell(x_g(t), u(t)) - \ell(x_f(t), u(t)))\, dt \\ &\qquad + \big(h(x_g(T)) - h(x_f(T))\big) \Big] d\gamma^*(g, f). \end{aligned}$$

Passing to the absolute value and using the bounds in (3.9) and (3.10), the expression of W_1 (3.11) comes up:

$$\begin{aligned} \Big| J_{\pi^N}[u] - J_{\pi^\infty}[u] \Big| &\leq \left[L_\ell \int_0^T t\, e^{L_f t} dt + L_h\, e^{L_f T} \right] \int_{\mathcal{X}\times\mathcal{X}} \|g - f\|_\infty \, d\gamma(g, f) \\ &= \left(L_\ell \, \frac{e^{L_f T}(L_f T - 1) + 1}{{L_f}^2} + L_h e^{L_f T} \right) W_1(\pi^N, \pi^\infty) \\ &= C(L_f, L_\ell, L_h, T)\, W_1(\pi^N, \pi^\infty). \end{aligned} \tag{3.12}$$

Note that this estimate does not depend on x_0 or u.

STEP 3: Lipschitz estimate for the value function.
We now prove the estimate (3.7) using (3.12). Fix an initial condition $x(t_0) = x_0$ and some $\varepsilon > 0$. By the definition of V_{π^∞}, there exists a control $u \in \mathcal{U}_{t_0}$ such that

$$J_{\pi^\infty}[u_\varepsilon] \leq V_{\pi^\infty}(t_0, x_0) + \varepsilon\,.$$

So we have

$$\begin{aligned} V_{\pi^N}(t_0, x_0) - V_{\pi^\infty}(t_0, x_0) &= \inf_{u\in\mathcal{U}_{t_0}} J_{\pi^N}[u] - \inf_{u\in\mathcal{U}_{t_0}} J_{\pi^\infty}[u] \\ &< \inf_{u\in\mathcal{U}_{t_0}} J_{\pi^N}[u] - J_{\pi^\infty}[u_\varepsilon] + \varepsilon \\ &\leq J_{\pi^N}[u_\varepsilon] - J_{\pi^\infty}[u_\varepsilon] + \varepsilon \\ &\leq \sup_{u\in\mathcal{U}_{t_0}} |J_{\pi^N} - J_{\pi^\infty}| + \varepsilon. \end{aligned}$$

In the same way, we get

$$V_{\pi^\infty}(t_0, x_0) - V_{\pi^N}(t_0, x_0) < \sup_{u\in\mathcal{U}_{t_0}} |J_{\pi^N} - J_{\pi^\infty}| + \varepsilon\,,$$

and since ε is arbitrary, we conclude

$$|V_{\pi^N}(t_0, x_0) - V_{\pi^\infty}(t_0, x_0)| \leq \sup_{u\in\mathcal{U}} |J_{\pi^N} - J_{\pi^\infty}|\ .$$

Finally, noting that the estimate is independent of x_0, we get our result:

$$||V_{\pi^N} - V_{\pi^\infty}||_\infty \leq \sup_{u\in\mathcal{U}} |J_{\pi^N} - J_{\pi^\infty}| \leq C(L_f, L_\ell, L_h, T)\, W_1(\pi^N, \pi^\infty).$$

□

In the particular case where $\pi^\infty \equiv \delta_{\widehat{f}}$, $\widehat{f}$ being the true underlying dynamics, then one can express Theorem 3.3 in a more expressive way as follows:

Corollary 3.4 (Convergence of the value functions)**.** *Suppose that $\widehat{f} : \mathbb{R}^n \times U \to \mathbb{R}^n$ is a Lipschitz continuous function and the assumption $(H2)$ on ℓ and h is satisfied. Let $\{\pi^N\}_{N\in\mathbb{N}} \subset \mathcal{P}(\mathcal{X})$ be a sequence of probability distributions on $\mathcal{X} \subset C(\mathbb{R}^n \times U; \mathbb{R}^n)$ compact set, with $\pi^N \xrightarrow{W_1} \delta_{\widehat{f}}$. Then the value function V_{π^N} of Problem B converges uniformly on $[0,T] \times \mathbb{R}^n$ to the value function V of Problem A.*

The previous Corollary provides a positive answer to the questions (A)-(B) posed in the introduction. More precisely, it tells us that, whenever it is possible to construct a sufficiently close (with respect to the Wasserstein distance) probability distribution π to the real dynamics f, then the value function V_π is a good approximation of the value function V. Note that Theorem 3.3 is useful even when the distribution π^N does not exactly reach $\delta_{\widehat{f}}$, in that it provides an error estimate for the algorithm.

3.3.1 On finite support measures converging to $\delta_{\widehat{f}}$

Let us consider the special case in which the distribution $\pi^N \in \mathcal{P}(\mathcal{X})$ is a linear combination of a finite number of Dirac deltas defined on a family of equi-bounded and equi-Lipschitz continuous functions $\mathcal{X} := \{f_1, \dots, f_M\}$:

$$\pi^N := \sum_{i=1}^{M} \alpha_i^N \, \delta_{f_i}, \tag{3.13}$$

where $\alpha_i^N \geq 0$, $\sum_{i=1}^M \alpha_i^N = 1$ for every $i = 1, \dots, M$ and $N \in \mathbb{N}$. In this case, the cost functional (3.4) can be written as

$$J_{\pi^N,s,x0}[u] := \sum_{i=1}^{M} \alpha_i^N \left[\int_{t_0}^{t} \ell(x_{f_i}(t), u(t))\, dt + h(x_{f_i}(T)) \right].$$

Without loss of generality, let us also assume that $f \equiv f_1$ is the real underlying dynamics. From Theorem 3.3 we obtain:

Corollary 3.5. *Let us consider a sequence of probability distributions $\{\pi^N\}_{N\in\mathbb{N}} \subset \mathcal{P}(\mathcal{X})$ defined as in (3.13). Assume that $f_1 : \mathbb{R}^n \times U \to \mathbb{R}^n$ is Lipschitz continuous and assume the assumption $(H2)$ on the functions ℓ and h. Then*

1. *$||V_{\pi^N} - V||_\infty \leq C(L_f, L_\ell, L_h, T)\, \mathbb{E}_{\pi^N}[\|f - g\|_\infty] \;\forall\, N \in \mathbb{N}$, where V is the value function of Problem A relative to the true dynamics f_1.*

2. *If $\{\pi^N\}_{N\in\mathbb{N}} \subset \mathcal{P}(\mathcal{X})$ is a sequence of probability distributions with the same support $\{f_1, \dots, f_M\}$ and π^N is converging to δ_{f_1}, then V_{π^N} converges to V.*

Proof. The thesis follows directly from Theorem 3.3, noting that the quantity

$$\mathbb{E}_{\pi^N}[\|f - g\|_\infty]$$

coincides with the 1-Wasserstein Distance between π^N and $\delta_{\widehat{f}}$:

$$W_1(\pi^N, \delta_{\widehat{f}}) := \inf_{\gamma \in \Gamma(\pi^N, \delta_{\widehat{f}})} \int_{\mathcal{X}\times\mathcal{X}} \|g_1 - g_2\|_\infty \, d\gamma(g_1, g_2) \ ,$$

where $\Gamma(\pi^N, \delta_{\widehat{f}})$ denotes the set of all joint probability distributions on $\mathcal{X} \times \mathcal{X}$, having π^N and $\delta_{\widehat{f}}$ as marginals. In this case, indeed, the set $\Gamma(\pi^N, \delta_{\widehat{f}})$ consists of a single element

$$\Gamma(\pi^N, \delta_{\widehat{f}}) = \left\{ \sum_{i=1}^{M} \alpha_i \, \delta_{(f_i, f)} \right\} \ ,$$

and we get

$$W_1(\pi^N, \delta_{\widehat{f}}) = \sum_{i=1}^{M} \alpha_i \, d_\infty(f_i, f) = \mathbb{E}_{\pi^N}[\|f - g\|_\infty].$$

□

3.4 Numerical tests

In this section, we will present three numerical tests. They all deal with extremely simplified situations: a parametric model, where the parameter can take only a finite number of values (cf. Section 3.3.1). We remark that these tests are intended for *illustrative purposes* only and their main goal is to verify that Theorem 3.3 and Corollary 3.4 hold in a particular case. Indeed, the theory developed here includes far more general cases than this, including parametric models with a large number of parameters and infinite possible values for each parameter (e.g. deep neural networks [56, 32, 71]) or even non-parametric models (e.g. Gaussian processes [37, 73]).

3.4.1 Test 1

We consider a dynamical system governed by the differential equation

$$\begin{cases} \dot{x}(t) = \lambda x(t) + sin(x(t)) + u(t) & t \in [t_0, T] \\ x(t_0) = x_0, \end{cases}$$

with $u(t) \in U = [-1, 1]$ for $t \in [t_0, T]$. The agent, who doesn't know the parameter λ, has a probability distribution on a set of 5 possible values for λ:

$$\Lambda = \{\lambda_1 = 0, \ \lambda_2 = 1, \ \lambda_3 = -1, \ \lambda_4 = 0.5, \ \lambda_5 = -0.5\} \, .$$

For each $N \in \mathbb{N}$ the probability distribution $\pi^N \in \mathcal{P}(\Lambda)$ can be written, similarly to (3.13), as

$$\pi^N = \sum_{i=1}^{5} \alpha_i^N \delta_{\lambda_i}.$$

Note that a probability distribution on a set of parameters is equivalent to a probability distribution on a family of functions $\mathcal{X} = \{f_i\}_i$ (see also Remark 3.1), where

$$f_i(x, u) = \lambda_i x + sin(x) + u.$$

We set $s \equiv 0$. The agent wants to minimize an averaged cost

$$\begin{aligned} J_{\pi^N,0,x_0}[u] &= \sum_{i=1}^{5} \alpha_i^N \left[\int_0^T u(t)^2 dt - x_i(T) \right] \\ &= \int_0^T u(t)^2 dt + \sum_{i=1}^{5} \alpha_i^N \left[-x_i(T) \right] \end{aligned} \tag{3.14}$$

over the set of measurable controls $\mathcal{U}_0$. Problem B associated to a fixed π^N can thus be seen as an optimal control problem in dimension 5, where the state variable include the five trajectories $x_1, \dots, x_5$.

We solved this problem numerically when the probabilities α_i^N were defined according to the following rule:

$$\alpha_1^N = 1 - \frac{1}{2^N}, \quad \alpha_i^N = \frac{1}{4}\frac{1}{2^N} \text{ for } i = 2, \dots, 5 .$$

It is clear that the sequence is converging to δ_{λ_1}, which we assume to be the parameter λ of the *true dynamics*, following the setting of subsection 3.3.1. An example of optimal (multi-)trajectory of Problem B relative to π^1 is plotted in Fig. 3.1. In order to minimize the cost functional (3.14), the agent tries to steer all the trajectories towards the positive values of the real axis, using a control close to $+1$; at the same time, the optimal control cannot be constantly $+1$, since the cost functional penalizes larger values of the control (Fig. 3.2).

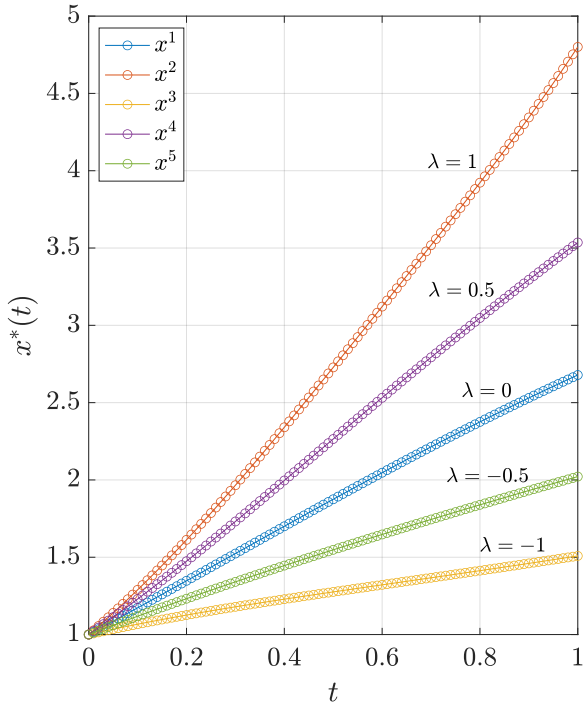


Figure 3.1. Test 1: Optimal (multi-)trajectory starting from $x_0 = 1$.

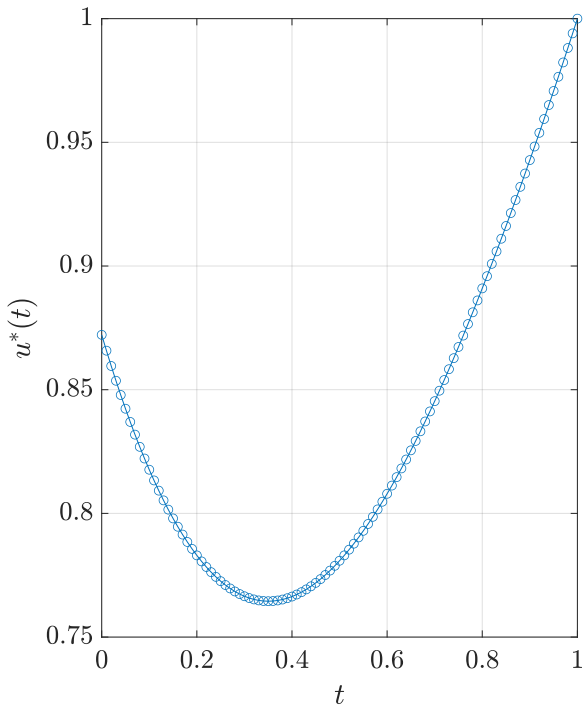


Figure 3.2. Test 1: Optimal control starting from $x_0 = 1$.

For each $N = 1, \dots, 8$ we computed the value function of Problem B solving the equations given by the Pontryagin Maximum Principle (see e.g. [50]), for a grid of initial points $x_0 \in [-1, 1]$. Then, we compared them to the value function of Problem A relative to the true dynamics f_1, computing the sup norm of the difference $V_{\pi^N} - V$ over the interval $[-1, 1]$; the results are reported in Table 3.1. Note that at each iteration the Wasserstein distance W_1 between π^N and δ_{f_1} is halved and so is the error; this means that the numerical convergence order is 1, which agrees to the estimate given by Corollary 3.5.

3.4.2 Test 2

We consider a perturbed logistic model. The true dynamics follows a classical logistic differential equation:

$$\dot{x_1}(t) = u(t)x_1(t)(1 - x_1(t)),$$

while the other dynamics have an additional goniometric term:

$$f_i(x, u) = (u + g_i(x))\, x(1 - x), \quad \text{for } i = 2, \dots, 5,$$

Table 3.1. Test 1: Errors for value functions related to π^N for $N = 1, \dots, 8$ with respect to the true value function of Problem A.

N	α_1^N	$\|\|V_{\pi^N} - V\|\|_{\infty,[-1,1]}$	*order*
1	0.5	1.57e-1	-
2	0.75	7.87e-2	1.00
3	0.875	3.94e-2	1.00
4	0.9375	1.97e-2	1.00
5	0.9687	9.84e-3	1.00
6	0.9844	4.92e-3	1.00
7	0.9922	2.46e-3	1.00
8	0.9961	1.23e-3	1.00

where $g_2(x) = sin(x)$, $g_3(x) = -sin(x)$, $g_4(x) = cos(x)$, $g_5(x) = sin(x)$. Thus, there are 5 possible dynamics:

$$\begin{cases} \dot{x_1}(t) = u(t)\, x_1(t)\, (1 - x_1(t)) \\ \dot{x_2}(t) = (u(t) + sin(x(t)))\; x_2(t)\, (1 - x_2(t)) \\ \dot{x_3}(t) = (u(t) - sin(x(t)))\; x_3(t)\, (1 - x_3(t)) \\ \dot{x_4}(t) = (u(t) + cos(x(t)))\; x_4(t)\, (1 - x_4(t)) \\ \dot{x_5}(t) = (u(t) - cos(x(t)))\; x_5(t)\, (1 - x_5(t)) \\ x_i(0) = x_0 \qquad i = 1, \dots, 5\ . \end{cases} \tag{3.15}$$

The agent doesn't know which of the previous ones is the true dynamics, he only knows a probability distribution on the space of functions $\mathcal{X} = \{f_1, \dots, f_5\}$. We use the same notation of the previous example:

$$\pi^N = \sum_{i=1}^{5} \alpha_i^N \delta_{f_i},$$

with $\alpha_1^N = 1 - \frac{1}{2^N}$ and $\alpha_i^N = \frac{1}{4}\frac{1}{2^N}$ for $i = 2, \dots, 5$.

The cost functional to be minimized is

$$J[u] = \int_0^T u(t)^2 dt + \sum_{i=1}^{5} \alpha_i \, |1 - x_i(T)| \tag{3.16}$$

Fig. 3.3 and Fig. 3.4 show respectively an example of optimal (multi-)trajectory and of an optimal control for Problem B with π^1. In Table 3.2 we reported the error of the value functions of Problem B with respect to that of Problem A, with the relative estimated convergence order. Note that the order of convergence is still 1, even if in this case the dynamics is quadratic and thus not globally Lipschitz continuous. This make us think that maybe the hypothesis of Theorem 3.5 could be relaxed, asking only a local Lipschitz continuity.

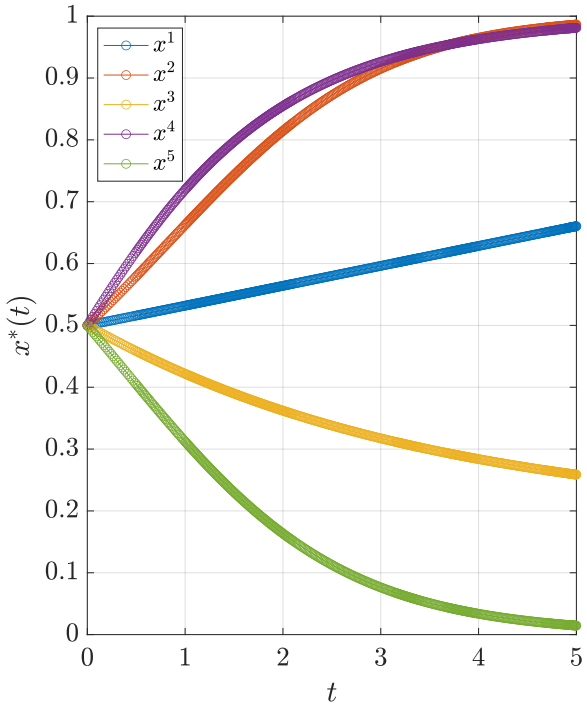


Figure 3.3. Test 2: Optimal (multi-)trajectory for π^1 starting from $x_0 = 0.5$.

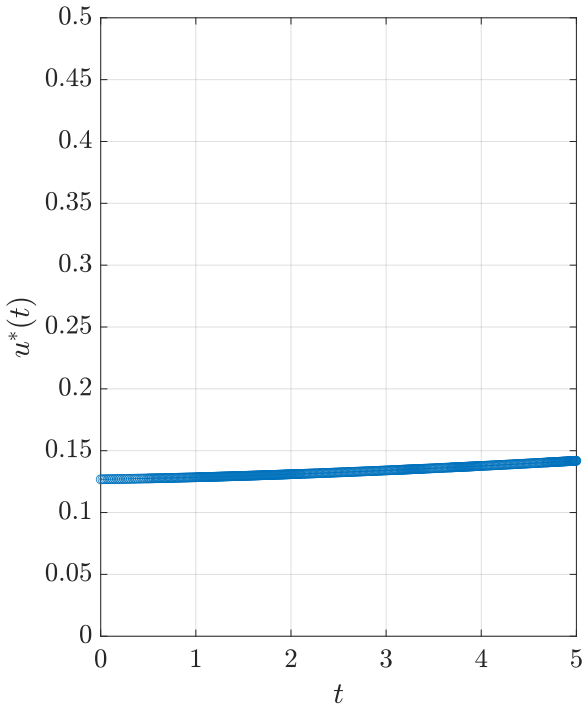


Figure 3.4. Test 2: Optimal control for π^1 starting from $x_0 = 0.5$.

Table 3.2. Test 2: Errors for value functions related to π^N with $N = 1, \dots, 8$ with respect to the true value function of Problem A.

N	α_1^N	$\|\|V_{\pi^N} - V\|\|_{\infty,[-1,1]}$	*order*
1	0.5	1.43e-1	-
2	0.75	7.14e-2	0.99
3	0.875	3.57e-2	1.00
4	0.9375	1.78e-2	1.00
5	0.9687	8.91e-3	1.00
6	0.9844	4.46e-3	1.00
7	0.9922	2.23e-3	1.00
8	0.9961	1.11e-3	1.00

3.4.3 Test 3

The convergence results presented in this chapter hold under the assumption that the cost functions ℓ and h are both globally Lipschitz continuous. With the following numerical test, we show that there are other examples of practical interest, where we can observe similar convergent behavior in the error $\|V_{\pi^N} - V\|_\infty$, even when this hypothesis is not verified.

In this third example, the state of the system is 2-dimensional, $x = (x^1, x^2)^T$. The three possible dynamics are all linear in the space variable and they differ only by the system matrix $A_i \in \mathbb{R}^{2\times 2}$:

$$\begin{pmatrix} \dot{x}_i^1 \\ \dot{x}_i^2 \end{pmatrix} = A_i \begin{pmatrix} x_i^1 \\ x_i^2 \end{pmatrix} + \begin{pmatrix} cos(u) \\ sin(u) \end{pmatrix},$$

where u is a 1-dimensional control which lies in $[0, 2\pi]$. The three possible matrices are

$$A_1 = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}, \; A_2 = \begin{pmatrix} 0.5 & 0 \\ 0 & 2 \end{pmatrix} \text{ and } A_3 = \begin{pmatrix} 0.5 & -0.5 \\ 0.5 & 0.5 \end{pmatrix}.$$

Without loss of generality, we assume that the true dynamics corresponds to the first matrix. The agent only knows a probability distribution π^N on the set of the three matrices, defined as in the previous example:

$$\pi^N = \sum_{i=1}^{3} \alpha_i^N \delta_{A_i},$$

where the weights are defined according to the rule

$$\alpha_1^N = 1 - \frac{1}{2^N}, \quad \alpha_i^N = \frac{1}{2^{N+1}} \;\text{ for } i = 2, 3.$$

We set again $s \equiv 0$. The cost functional to be minimized is

$$J_{\pi^N,0,x_0}[u] = \frac{1}{2} \sum_{i=1}^{3} \alpha_i^N \left[\int_0^T \|x_i(t)\|^2 \, dt + \|x_i(T)\|^2 \right], \tag{3.17}$$

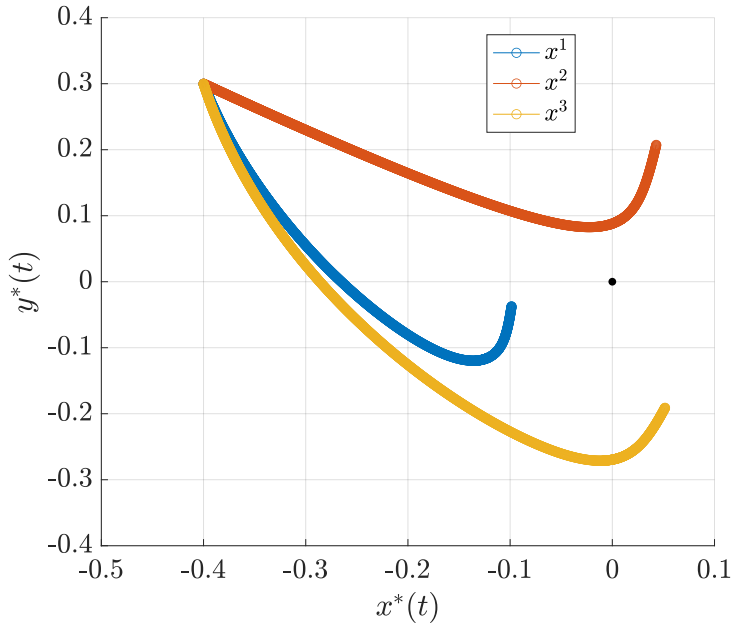


Figure 3.5. Test 3: Optimal (multi-)trajectory for $\pi = \left\{\frac{1}{3}, \frac{1}{3}, \frac{1}{3}\right\}$ starting from $x_0 = (-0.4, 0.3)$.

where $\|\cdot\|$ indicates the euclidean norm in $\mathbb{R}^2$. In Fig. 3.5 we can see an example of optimal (multi-)trajectory for this problem, when the distribution π has been chosen to be $\left\{\frac{1}{3}, \frac{1}{3}, \frac{1}{3}\right\}$. In this case, the agent has to minimize the average squared distance of the trajectories from the origin, thus he looks for a single control to steer all three trajectories towards the origin at the same time. We can observe that none of the trajectories reaches exactly the origin, since a control which is optimal for one of the three dynamics may not be optimal for the other two. In Table 3.3 we

Table 3.3. Test 3: Errors for value functions related to π^N with $N = 1, \dots, 6$ with respect to the true value function of Problem A.

N	α_1^N	$\|\|V_{\pi^N} - V\|\|_{\infty,[-1,1]^2}$	*order*
1	0.5	2.52e-0	-
2	0.75	1.32e-0	0.94
3	0.875	6.81e-1	0.95
4	0.9375	3.47e-1	0.97
5	0.9687	1.75e-1	0.98
6	0.9844	8.81e-2	0.99

report the distance between the value function of Problem B and the true value function of Problem A, for different distributions π^N. Even in this case, although the hypothesis of Corollary 3.5 are not satisfied, we can see a clear convergence with order 1.

Chapter 4

Convergence results for an average cost LQR problem

In this chapter, we deal with a LQR problem (see Section 2.1.2) where the dynamics is partially unknown. Recall that the underlying system dynamics is linear:

$$\dot{x}(t) = \widehat{A}x(t) + Bu(t).$$

We assume that the true state matrix $\widehat{A}$ is unknown, whereas the control matrix B is known. Our knowledge of the matrix $\widehat{A}$ is described by a probability distribution on a compact space of matrices $\mathcal{A}$ to which $\widehat{A}$ belongs. In a higher perspective, we can identify $\mathcal{A}$ with a class of *linear* dynamical systems, namely

$$\mathcal{A} \hookrightarrow \{(x,u) \mapsto Ax + Bu \mid A \in \mathcal{A}\} =: \widetilde{\mathcal{A}},$$

and see π as a probability distribution on a space of functions $\widetilde{\mathcal{A}}$ as well. Thus, the problem considered here can be seen as a special case of the framework presented in Chapter 3. In this setting, we were able to prove *the convergence of the value function and of the optimal control*. The results of this chapter have been published in a recent work [108].

The chapter is organized as follows. In Section 4.1, we state the problem formulation and study the basic properties of the average cost LQR problem. Then, in Section 4.2 we also derive a Pontryagin's Maximum Principle for the problem, refining some results in [23]. In Section 4.3 we state and prove the main convergence results. In Section 4.4, we strengthen the results presented in Section 5 4.3 in the case in which one is dealing with a discrete probability measure π. That result is further stressed in Section 4.5, where we present and analyze a numerical example.

4.1 Problem Statements and Preliminary Results

We begin our discussion considering two *Linear Quadratic Regulator (LQR)* optimal control problems. We will see in the sequel how the two problems are connected.

Problem A: the LQR problem. Let us recall the finite-horizon LQR problem presented in Section 2.1.2, which we will refer to as *Problem A*:

$$\begin{cases} \text{minimize } J_{t_0,x_0}[u] \\ \text{over } (x,u)(\cdot) \text{ such that } u \in \mathcal{U}_{t_0} \text{ and} \\ \dot{x}(t) = \widehat{A}x(t) + Bu(t), \quad t \in [t_0, T], \\ x(t_0) = x_0 \end{cases} \tag{4.1}$$

where $0 \leq t_0 < T$, $x_0 \in \mathbb{R}^n$, $\mathcal{U}_{t_0} \coloneqq \{u : [t_0, T] \to \mathbb{R}^m, \text{ Lebesgue measurable}\}$ and

$$J_{t_0,x_0}[u] \coloneqq \frac{1}{2}\int_{t_0}^{T} \Big(x(t)^T Q x(t) + u(t)^T R u(t)\Big)\, dt + \frac{1}{2}x(T)^T Q_f x(T)\ . \tag{4.2}$$

Throughout the whole chapter, the following *Standing Hypothesis* will be imposed both for Problem A and for Problem B:

(SH): $Q, Q_f \in \mathbb{R}^{n\times n}$ are symmetric and semipositive definite and $R \in \mathbb{R}^{m\times m}$ is symmetric and positive definite.

The pair $(x,u)(\cdot)$ such that $u \in \mathcal{U}_{t_0}$ and $x(\cdot)$ is the solution of the Cauchy problem

$$\begin{cases} \dot{x}(t) = \widehat{A}x(t) + Bu(t) & t \in [t_0, T] \\ x(t_0) = x_0\ . \end{cases} \tag{4.3}$$

is called *admissible process for Problem A*.
Let us define the *value function* $V : [0,T] \times \mathbb{R}^n \to \mathbb{R}$ for Problem A as

$$V(t_0, x_0) \coloneqq \inf_{u \in \mathcal{U}_{t_0}} J_{t_0,x_0}[u]\,. \tag{4.4}$$

We shall say that $(\bar{x}, \bar{u})(\cdot)$ is an *optimal process for Problem A* if

$$J_{t_0,x_0}[\bar{u}] \leq J_{t_0,x_0}[u] \tag{4.5}$$

for any other admissible process $(x,u)(\cdot)$ of Problem A. In this case, $\bar{u}$ will be denoted as *optimal control* for Problem A.

Problem B: an LQR problem with unknown dynamics. Let us now introduce an optimal control that does not require the exact knowledge of the matrix $\widehat{A}$, but merely a probability distribution defined on a compact space of matrices $\mathcal{A}$ containing $\widehat{A}$. For each $0 \leq t_0 < T$, $x_0 \in \mathbb{R}^n$ and $\pi \in \mathcal{P}(\mathcal{A})$, consider the following optimal control problem, which we will refer to as *Problem B*:

$$\begin{cases} \text{minimize } J_{\pi,t_0,x_0}[u] \\ \text{over } \{(x_A,u)(\cdot) :\ A \in \mathcal{A}\} \text{ such that } u \in \mathcal{U}_{t_0} \text{ and} \\ \dot{x}_A(t) = Ax_A(t) + Bu(t), \quad A \in \mathcal{A}, \quad t \in [t_0, T], \\ x_A(t_0) = x_0, \quad A \in \mathcal{A}, \end{cases} \tag{4.6}$$

where $\mathcal{U}_{t_0} \coloneqq \{u : [t_0, T] \to \mathbb{R}^m \text{ Lebesgue measurable}\}$ and

$$J_{\pi,t_0,x_0}[u] \coloneqq \mathbb{E}_\pi \left[\frac{1}{2} \int_{t_0}^{T} \Big(x_A(t)^T Q x_A(t) + u(t)^T R u(t) \Big)\, dt + \frac{1}{2} x_A(T)^T Q_f x_A(T) \right]$$
$$= \int_{\mathcal{A}} \left[\frac{1}{2} \int_{t_0}^{T} \Big(x_A(t)^T Q x_A(t) + u(t)^T R u(t) \Big)\, dt + \frac{1}{2} x_A(T)^T Q_f x_A(T) \right] d\pi(A)\ . \tag{4.7}$$

Remark 4.1. Sometimes we will denote this problem as Problem B_π, to stress its dependency on the probability distribution π.

The definition of the value function $V_\pi : [0, T] \times \mathbb{R}^n \to \mathbb{R}$ for Problem B is

$$V_\pi(t_0, x_0) \coloneqq \inf_{u \in \mathcal{U}_{t_0}} J_{\pi,t_0,x_0}[u]\,. \tag{4.8}$$

The collection $\{(x_A, u)(\cdot) :\ A \in \mathcal{A}\}$ such that $u \in \mathcal{U}_{t_0}$ and, for each $A \in \mathcal{A}$, $x_A(\cdot)$ is the solution of the Cauchy problem

$$\begin{cases} \dot{x}_A(t) = A x_A(t) + B u(t) & t \in [t_0, T] \\ x_A(t_0) = x_0 \end{cases} \tag{4.9}$$

is called *admissible process for Problem B*. Note that the initial condition is the same for every A. The admissible process $\{(\bar{x}_A, \bar{u})(\cdot) :\ A \in \mathcal{A}\}$ is *optimal for Problem B* if

$$J_{\pi,t_0,x_0}[\bar{u}] \le J_{\pi,t_0,x_0}[u] \tag{4.10}$$

for any other admissible process $\{(x_A, u)(\cdot) :\ A \in \mathcal{A}\}$ of Problem B. In this case, $\bar{u}$ will be denoted as *optimal control for Problem B*.

Remark 4.2. As we will show in Lemma 4.5, Problem B admits a unique optimal process when the assumption **(SH)** holds true. Furthermore, it is interesting to observe that Problem A can be regarded as a particular case of Problem B when one chooses $\pi = \delta_{\widehat{A}}$, namely when π is a Dirac delta concentrated at $\widehat{A}$:

$$\text{Problem } \mathrm{B}_{\delta_{\widehat{A}}} \hookrightarrow \text{Problem A}$$

Remark 4.3. In our framework, we consider a probability distribution merely on the matrix A of the dynamics and not on the matrix B. It is not hard to check that the arguments proposed and the results achieved in the next sections hold as well in an extended framework where the matrix B is possibly unknown. However, we preferred to consider a simpler case to keep the overall presentation as clear as possible. Moreover, it seems reasonable to assume that the agent doesn't know how the environment works (matrix A), whereas being aware of how the control affects the system (matrix B).

4.1.1 Preliminary results for Problem B

Let us start showing a series of basic results on the existence and the regularity of trajectories and optimal controls for the system we are considering.

In this section we assume that π is a probability distribution on a compact set of matrices $\mathcal{A} \subset \mathbb{R}^{n\times n}$. Since $\mathcal{A}$ is bounded, there exists a constant C_A such that $\|A\|_2 \leq C_A,\ \forall\, A \in \mathcal{A}$.
For a given matrix $A \in \mathcal{A}$ and an admissible control $u \in \mathcal{U}_{t_0}$, the notation $x_A(t;u)$ will denote the solution of (4.9) relative to A and u. However, when it is not ambiguous, we will omit the dependency on the control u and write simply $x_A(t)$.

Lemma 4.4 (Boundedness and continuity of trajectories)**.** *Let us consider the dynamical system in* (4.9)*. The following results hold:*

(i) For each $u \in L^1([t_0,T];\mathbb{R}^m)$, the trajectory $x_A(\cdot;u)$ is uniformly bounded for all $A \in \mathcal{A}$:

$$|x_A(t;u)| \leq C_x^u \qquad \forall\, t \in [t_0,T],\, \forall\, A \in \mathcal{A}\,,$$

where C_x^u is a constant which depends on u and on x_0.

(ii) For each $u \in L^1([t_0,T];\mathbb{R}^m)$, the map $A \mapsto x_A(\cdot;u)$ is continuous;

(iii) For each $A \in \mathcal{A}$, the map $u \mapsto x_A(\cdot;u)$ is continuous for $u \in L^1([t_0,T];\mathbb{R}^m)$.

Proof. (i) Recall that $x_A(t)$ satisfies the relation

$$x_A(t) = x_0 + \int_{t_0}^{t} Ax_A(\tau) + Bu(\tau)\,d\tau, \qquad \forall\, t \in [t_0,T].$$

for each $A \in \mathcal{A}$. Then

$$|x_A(t)| \leq |x_0| + \int_{t_0}^{t} \|A\|_2\,|x_A(\tau)|\,d\tau + \|B\|_2 \int_{t_0}^{t} |u(\tau)|\,d\tau \qquad \forall\, t \in [t_0,T],$$

so by the Grönwall Lemma (see e.g. Lemma 2.4.4 on [143]) we get

$$\begin{aligned} |x_A(t)| &\leq \left(|x_0| + \|B\|_2 \int_{t_0}^{t} |u(\tau)|\,d\tau\right) e^{\|A\|_2(t-t_0)} \\ &\leq \left(|x_0| + \|B\|_2 \int_{t_0}^{t} |u(\tau)|\,d\tau\right) e^{C_A T} =: C_x^u \end{aligned} \tag{4.11}$$

for every $A \in \mathcal{A}$ and $t \in [t_0,T]$. This shows condition (i).

(ii) Fix a control $u \in L^1([t_0,T];\mathbb{R}^m)$ and consider the trajectories solutions of (4.9) relative to two different matrices A and A'. If we define

$$z(t) \coloneqq x_A(t) - x_{A'}(t)\,,$$

then z solves the following differential system:

$$\begin{cases} \dot{z}(t) = Ax_A(t) - A'x_{A'}(t) & t \in [t_0,T] \\ z(t_0) = 0\ . \end{cases}$$

Notice that we can rewrite the right hand side as

$$Ax_A(t) - A'x_{A'}(t) = Az(t) + (A - A')x_{A'}(t)\,,$$

and thus we can give the estimate

$$|\dot{z}(t)| \leq C_A|z(t)| + \|A - A'\|_2 C_x^u .$$

Applying again the Grönwall Lemma on z, we get

$$|z(t)| \leq \|A - A'\|_2 C_x^u \int_{t_0}^t e^{\int_\tau^t C_A d\sigma} d\tau \leq TC_x^u e^{C_A T} \|A - A'\|_2 \quad \forall\, t \in [t_0, T] ,$$

which implies the continuity of the map $A \mapsto x_A(\cdot)$.

(iii) Fix a matrix $A \in \mathcal{A}$ and consider the trajectories relative to two different controls $u, u' \in \mathcal{U}_{t_0}$. In a similar way as in (ii) we define

$$z(t) \coloneqq x_A(t; u) - x_A(t; u') ,$$

which solves the ODE system

$$\begin{cases} \dot{z}(t) = Az(t) + B\big(u(t) - u'(t)\big) & t \in [t_0, T] \\ z(t_0) = 0 , \end{cases}$$

and, by Grönwall's Lemma, we get

$$|z(t)| \leq \|B\|_2 \int_{t_0}^t e^{\int_\tau^t C_A d\sigma} |u(\tau) - u'(\tau)| d\tau \leq e^{C_A T} \|B\|_2 \|u - u'\|_1$$

for all $t \in [t_0, T]$. The last inequality gives the continuity with respect to $u \in L^1([t_0, T]; \mathbb{R}^m)$. □

The following result guarantees that the minimization problem (4.6) is well posed.

Lemma 4.5 (Existence, uniqueness and upper bounds of the optimal control)**.** *Let the assumption **(SH)** hold true. Given $\pi \in \mathcal{P}(\mathcal{A})$, Problem B* (4.6) *admits a unique minimizer $\{(\bar{x}_A, \bar{u})(\cdot) : A \in \mathcal{A}\}$, satisfying the following upper bound:*

$$\left(\int_{t_0}^T |\bar{u}(t)|^2 \, dt \right)^{1/2} \leq \overline{C}_u , \tag{4.12}$$

where $\overline{C}_u$ does not depend on π and is defined as

$$\overline{C}_u \coloneqq \sqrt{\frac{1}{r_1} \left(T \|Q\|_2 + \|Q_f\|_2 \right) |x_0|^2 e^{2C_A T}} ,$$

where r_1 is the smallest eigenvalue of the matrix R.

Proof. Consider a minimizing sequence $u^k \in \mathcal{U}_{t_0}$ of the cost functional J_{π,t_0,x_0} defined in (4.7), satisfying $J_{\pi,t_0,x_0}[u^k] \to \inf_{u \in \mathcal{U}_{t_0}} J_{\pi,t_0,x_0}[u]$, $\inf_{u \in \mathcal{U}_{t_0}} J_{\pi,t_0,x_0}[u] \leq J_{\pi,t_0,x_0}[u^k]$, and the related minimizing process $\{(x_A^k, u^k)(\cdot) \colon A \in \mathcal{A}\}$. Set

$$\varepsilon_k \coloneqq J_{\pi,t_0,x_0}[u^k] - \inf_{u \in \mathcal{U}_{t_0}} J_{\pi,t_0,x_0}[u] \geq 0.$$

It is not restrictive to assume that $\varepsilon_k < 1$ for all $k \in \mathbb{N}$.

Let us consider the system (4.9) when the control is $u^0 \equiv 0$. The process $\{(x_A^0, u^0)(\cdot) \colon A \in \mathcal{A}\})$ is solution of

$$\begin{cases} \dot{x}_A^0(t) = Ax_A^0(t) & t \in [t_0, T] \\ x_A^0(t_0) = x_0 \end{cases}$$

for every $A \in \mathcal{A}$. Hence

$$|x_A^0(t)| = |e^{At}\, x_0| \leq |x_0| e^{\|A\|_2 T} \qquad \forall\, t \in [0,T], \quad \forall A \in \mathcal{A}\ .$$

The cost achieved by the control u^0 can be estimated as follows:

$$\begin{aligned} J_{\pi,t_0,x_0}[u^0] &= \mathbb{E}_\pi \left[\frac{1}{2} \int_{t_0}^T x_A^0(t)^T Q x_A^0(t)\, dt + \frac{1}{2} x_A^0(T)^T Q_f x_A^0(T) \right] \\ &\leq \mathbb{E}_\pi \left[\frac{1}{2} T \left\|Q\right\|_2 |x_0|^2 e^{2\|A\|_2 T} + \frac{1}{2} \left\|Q_f\right\|_2 |x_0|^2 e^{2\|A\|_2 T} \right] \\ &\leq \frac{1}{2} \left(T \left\|Q\right\|_2 + \left\|Q_f\right\|_2 \right) |x_0|^2 e^{2C_A T}\ . \end{aligned}$$

From the construction of the minimizing sequence we get

$$J_{\pi,t_0,x_0}[u^k] \leq J_{\pi,t_0,x_0}[u^0] + \varepsilon_k \leq \frac{1}{2} \left(T \left\|Q\right\|_2 + \left\|Q_f\right\|_2 \right) |x_0|^2 e^{2C_A T} + \varepsilon_k\ .$$

Since the matrix $R > 0$, one has

$$\begin{aligned} J_{\pi,t_0,x_0}[u^k] &= \mathbb{E}_\pi \left[\frac{1}{2} \int_{t_0}^T x_A^k(t)^T Q x_A^k(t) + u^k(t)^T R u^k(t)\, dt + \frac{1}{2} x_A^k(T)^T Q_f x_A^k(T) \right] \\ &\geq \frac{1}{2} \int_{t_0}^T u^k(t)^T R u^k(t)\, dt \geq \frac{1}{2} \int_{t_0}^T r_1 |u^k(t)|^2\, dt\,, \end{aligned}$$

where r_1 is the smallest eigenvalue of the matrix R.
Hence one obtains the bound on the minimizing sequence

$$\int_{t_0}^T |u^k(t)|^2\, dt \leq \frac{1}{r_1} \left(T \left\|Q\right\|_2 + \left\|Q_f\right\|_2 \right) |x_0|^2 e^{2C_A T} + \frac{2\varepsilon_k}{r_1}\ , \tag{4.13}$$

which results in a uniformly bounded norm:

$$\left\|u^k\right\|_2 \leq \sqrt{\frac{1}{r_1} \left(T \left\|Q\right\|_2 + \left\|Q_f\right\|_2 \right) |x_0|^2 e^{2C_A T} + \frac{2}{r_1}} =: C_u \quad \forall k \in \mathbb{N}\ . \tag{4.14}$$

In view of the previous relation, it follows from standard compactness arguments that $u^k \rightharpoonup \bar{u}$ weakly in $L^2([t_0,T];\mathbb{R}^m)$. Since u^k is uniformly bounded in L^2, then using in turn the relation (4.11), the Hölder inequality and the relation (4.14), one obtains that there exists a constant $C_x > 0$ such that

$$|x_A^k(t)| \leq C_x^{u^k} \leq \left(|x_0| + \sqrt{T} \left\|B\right\|_2 C_u \right) e^{C_A T} =: C_x \tag{4.15}$$

holds for every $k \in \mathbb{N}$, $A \in \mathcal{A}$ and $t \in [t_0, T]$. Furthermore, for each $k \in \mathbb{N}$, $A \in \mathcal{A}$ and $t \in [t_0, T]$, one has

$$\begin{aligned}\int_{t_0}^{t} \left|\dot{x}_A^k(\tau)\right| d\tau &\leq \int_{t_0}^{t} \|A\|_2 \, |x_A^k(\tau)| d\tau + \int_{t_0}^{t} \|B\|_2 \, |u^k(\tau)| d\tau \\ &\leq T C_A C_x + \sqrt{T} \, \|B\|_2 \, C_u \,,\end{aligned}$$

which implies that, for each $A \in \mathcal{A}$, $\dot{x}_A^k \rightharpoonup \dot{\bar{x}}_A$ weakly in $L^1([t_0, T]; \mathbb{R}^m)$, $x_A^k \to \bar{x}_A$ uniformly in $[t_0, T]$ and, in view of the linearity of the control system, the process $\{(\bar{x}_A, \bar{u})(\cdot) : \ A \in \mathcal{A}\}$ is the solution of the linear system

$$\begin{cases} \dot{\bar{x}}_A(t) = A\bar{x}_A(t) + B\bar{u}(t), & t \in [t_0, T], \\ \bar{x}_A(t_0) = x_0, \end{cases}$$

for each $A \in \mathcal{A}$. So the process $\{(\bar{x}_A, \bar{u})(\cdot) : \ A \in \mathcal{A}\}$ is a minimizer for Problem B and, in view of (4.13), $\bar{u}$ satisfies the bound (4.12) with the stricter constant

$$\overline{C}_u \coloneqq \sqrt{\frac{1}{r_1} \left(T \, \|Q\|_2 + \|Q_f\|_2\right) |x_0|^2 e^{2C_A T}} \,.$$

Since the functional $u \mapsto J_{\pi, t_0, x_0}[u]$ is strictly convex, the uniqueness of the minimizer follows by standard arguments. This completes the proof. □

Remark 4.6. In view of the previous results, if $\{(\bar{x}_A, \bar{u})(\cdot) : \ A \in \mathcal{A}\}$ is an optimal process for problem B, then the constant

$$\overline{C}_x \coloneqq \left(|x_0| + \sqrt{T} \, \|B\|_2 \, \overline{C}_u\right) e^{C_A T}$$

is such that

$$|\bar{x}_A(t)| \leq \overline{C}_x, \quad \forall A \in \mathcal{A}, \quad \forall t \in [t_0, T]$$

and does not depend on π.

4.2 Optimality conditions

Let us consider the optimal control problem

$$\begin{cases} \text{minimize } J_{\pi, t_0, x_0}[u] \\ \text{over } u : [t_0, T] \to \mathbb{R}^m \text{ measurable such that} \\ u(t) \in U(t) \ \text{ a.e. } t \in [t_0, T], \\ \dot{x}_A(t) = A x_A(t) + B u(t), \qquad A \in \mathcal{A}, \qquad t \in [t_0, T] \\ x_A(t_0) = x_0, \end{cases} \tag{4.16}$$

where $U : [t_0, T] \rightsquigarrow \mathbb{R}^m$ is a $\mathscr{L} \times \mathscr{B}_{\mathbb{R}^m}$-measurable multifunction taking values compact sets and J_{π, t_0, x_0} is the cost functional defined in (4.7) for a given, fixed $\pi \in \mathcal{P}(\mathcal{A})$. For a given reference process $\{(\bar{x}_A, \bar{u})(\cdot) : \ A \in \mathcal{A}\}$, we assume that the following condition holds true:

(TH): There exist $\delta > 0$ and a function $c \in L^2([t_0, T]; \mathbb{R})$ such that

$$|Ax + Bu| \leq c(t),$$

for all $x \in \mathbb{B}_n(\bar{x}_A(t), \delta)$, $u \in U(t)$, $A \in \mathcal{A}$, a.e. $t \in [t_0, T]$.

It follows from standard ODE theory that, when condition **(TH)** holds, for every admissible process $\{(\bar{x}_A, \bar{u})(\cdot) : \ A \in \mathcal{A}\}$ one has that $\bar{x}_A(\cdot)$ is in $W^{1,1}([t_0, T]; \mathbb{R}^n)$ for all $A \in \mathcal{A}$.

Definition 4.7. For a given $\delta > 0$, a process $\{(\bar{x}_A, \bar{u})(\cdot) : \ A \in \mathcal{A}\}$ is said to be a $W^{1,1}$-local minimizer for problem (4.16) if

$$J_{\pi,t_0,x_0}[\bar{u}] \leq J_{\pi,t_0,x_0}[u]$$

for every process $\{(x_A, u)(\cdot) : \ A \in \mathcal{A}\}$ such that

$$\sup_{A \in \mathcal{A}} \|\bar{x}_A(\cdot) - x_A(\cdot)\|_{W^{1,1}} \leq \delta.$$

We recall the following result due to Bettiol and Khalil, that is a special case of Theorem 3.3 in [23].

Theorem 4.8 (Bettiol-Khalil, 2019)**.** *Let $\{(\bar{x}_A, \bar{u})(\cdot) : \ A \in \mathcal{A}\}$ be a $W^{1,1}$-local minimizer for the optimal control problem* (4.16)*. Let the assumption **(TH)** be satisfied. Then, there exists a $\mathscr{L} \times \mathscr{B}_{\mathcal{A}}$ measurable function $p\colon [t_0, T] \times \mathcal{A} \to \mathbb{R}^n$, $p(t, A) \equiv p_A(t)$, such that*

$$(i) \qquad p_A(\cdot) \in W^{1,1}([t_0, T]; \mathbb{R}^n) \qquad \forall\, A \in \mathit{supp}(\pi)\,;$$

$$(ii) \qquad \int_{\mathcal{A}} p_A(t) B\bar{u}(t)\, d\pi(A) - \frac{1}{2}\bar{u}(t)^T R\bar{u}(t) = \max_{u \in U(t)} \left\{ \int_{\mathcal{A}} p_A(t) Bu\, d\pi(A) - \frac{1}{2} u^T R u \right\} \quad \text{a.e. } t \in [t_0, T]$$

$$(iii) \qquad -\dot{p}_A(t) = A^T p_A(t) - Q\bar{x}_A(t) \qquad \textit{a.e. } t \in [t_0, T],\ \forall A \in \mathit{supp}(\pi)\,;$$

$$(iv) \qquad -p_A(T) = Q_f \bar{x}_A(T) \qquad \forall A \in \mathit{supp}(\pi)\,.$$

Remark 4.9. Let us notice that Theorem 3.3 in [23] is derived under the stronger assumption:

(TH'): there exist $\delta > 0$ and $c > 0$ such that

$$|Ax + Bu| \leq c\,,$$

for all $x \in \mathbb{B}(\bar{x}_A(t), \delta)$, $u \in U(t)$, $A \in \mathcal{A}$, a.e. $t \in [t_0, T]$.

However, a scrutiny of the proof given there reveals that the result still holds true under the relaxed condition **(TH)**. Furthermore, Theorem 3.3 in [23] is derived for a Mayer optimal control problem, i.e. with only a final cost, but an analogous theorem for Bolza optimal control problems can be easily obtained by using a standard state augmentation argument.

We are now ready to prove the necessary optimality conditions for Problem B:

Theorem 4.10. *Let **(SH)** hold true. Then the following optimality condition is satisfied by the minimizer* $\{(\bar{x}_A, \bar{u})(\cdot) : A \in \mathcal{A}\}$ *for Problem B* (4.6)*. There exists a* $\mathscr{L} \times \mathscr{B}_{\mathcal{A}}$ *measurable function* $p\colon [t_0, T] \times \mathcal{A} \to \mathbb{R}^n$*,* $p(t, A) \equiv p_A(t)$*, such that*

$$(i) \qquad p_A(\cdot) \in W^{1,1}([t_0, T]; \mathbb{R}^n) \qquad \forall\, A \in \mathit{supp}(\pi)\,;$$

$$(ii) \qquad \bar{u}(t) = +R^{-1}B^T \int_{\mathcal{A}} p_A(t)\, d\pi(A) \qquad t \in [t_0, T]\ ;$$

$$(iii) \qquad -\dot{p}_A(t) = A^T p_A(t) - Q\bar{x}_A(t) \qquad \text{a.e. } t \in [t_0, T],\ \forall A \in \mathit{supp}(\pi)\,;$$

$$(iv) \qquad -p_A(T) = Q_f \bar{x}_A(T) \qquad \forall A \in \mathit{supp}(\pi)\,.$$

Proof. Let us first observe that the optimal process $\{(\bar{x}_A, \bar{u})(\cdot) : A \in \mathcal{A}\}$ exists and is unique, due to Lemma 4.5. Consider now the optimal control problem

$$\begin{cases} \text{minimize } J_{\pi,t_0,x_0}[u] \\ \text{over } u : [t_0, T] \to \mathbb{R}^m \text{ measurable such that} \\ u(t) \in \mathbb{B}_m(\bar{u}(t), 1) \ \text{ a.e. } t \in [t_0, T], \\ \dot{x}_A(t) = Ax_A(t) + Bu(t), \qquad\qquad A \in \mathcal{A},\ t \in [t_0, T] \\ x_A(t_0) = x_0\ . \end{cases} \tag{4.17}$$

Such an optimal control problem is a special case of (4.16) with the choice of $U(t) = \mathbb{B}_m(\bar{u}(t), 1)$. Clearly, since $\{(\bar{x}_A, \bar{u})(\cdot) : A \in \mathcal{A}\}$ is a minimizer for Problem B, then it is also a minimizer for problem (4.17). Furthermore, since any element of $u \in U(t)$ can be written as $u = \bar{u}(t) + v$, for some $v \in \mathbb{B}_m(0, 1)$ and in view of Remark 4.6, then one can easily find $\delta > 0$ and a function $c \in L^2([t_0, T], \mathbb{R})$ such that the hypothesis **(TH)** is satisfied. Indeed,

$$|Ax + Bu| = |Ax + B(\bar{u}(t) + v)| \le \|A\|_2\,(\overline{C}_x + \delta) + \|B\|_2\,(|\bar{u}(t)| + 1) =: c(t) \tag{4.18}$$

for all $x \in \mathbb{B}_n(\bar{x}_A(t), \delta)$, $\forall A \in \mathcal{A}$, for all $u \in \mathbb{B}_m(\bar{u}(t), 1)$, a.e. $t \in [t_0, T]$, where $\overline{C}_x$ is the constant appearing in Remark 4.6. So the process $\{(\bar{x}_A, \bar{u})(\cdot) : A \in \mathcal{A}\}$ is a $W^{1,1}$-minimizer (see Definition 4.7) for the optimal control problem (4.17) and the hypothesis **(SH)** are satisfied. Then one can invoke Theorem 4.8, which provides conditions (i)-(iii)-(iv) of the statement. In order to obtain condition (ii), it is enough to observe that

$$\bar{u}(t) = \arg\max_{u \in U(t)} \left\{ \int_{\mathcal{A}} p_A(t) Bu\, d\pi(A) - \frac{1}{2} u^T R u \right\}, \qquad \text{a.e. } t \in [t_0, T]$$

and that $\bar{u}(t)$ is clearly an interior point of $U(t)$ for a.e. $t \in [t_0, T]$. Hence, $\bar{u}(t)$ has to satisfy also condition (ii) of Theorem 4.10. This completes the proof. □

Remark 4.11. Theorem 4.10 provides the existence of a multiplier $p_A(\cdot)$ for all $A \in supp(\pi)$. In general, we can extend its definition to all $A \in \mathcal{A}$, considering $p_A(\cdot)$ as the unique solution of

$$\begin{cases} -\dot{p}_A(t) = A^T p_A(t) - Q\bar{x}_A(t) & t \in [t_0, T] \\ -p_A(T) = Q_f \bar{x}_A(T) \ , \end{cases} \tag{4.19}$$

where $\{(\bar{x}_A, \bar{u})(\cdot) : \ A \in \mathcal{A}\}$ is the unique minimizer of Problem B.

The following result guarantees that this extension defined in Remark 4.11 is continuous with respect to A:

Lemma 4.12 (Boundedness and continuity of multipliers)**.** *Let us consider the multipliers defined in* (4.19) *for each $A \in \mathcal{A}$. They have the following properties:*

(i) There exists a positive constant $\overline{C}_p$, independent from π, which bounds uniformly all multipliers, i.e.

$$|p_A(t)| \leq \overline{C}_p \qquad \forall\, t \in [t_0, T], \forall\, A \in \mathcal{A} \ ,$$

(ii) The map $A \mapsto p_A(\cdot)$ is continuous.

Proof. The proof is similar to that of Lemma 4.4, with the only difference that here we need to apply the Grönwall Lemma backward instead of forward. Notice that the final condition

$$-p_A(T) = Q_f \bar{x}_A(T)$$

will not be the same for all $A \in \mathcal{A}$ whereas the initial condition was the same in Lemma 4.4. However, it is still continuous with respect to A by Lemma 4.4, so the same arguments can be applied to prove (ii). □

4.3 Main convergence results

In this section we will present our main results, regarding respectively the *Convergence of the value function* (Corollary 4.14) and the *Convergence of the optimal control* (Theorem 4.15). Given a sequence of probability distributions $\{\pi^N\} \subset \mathcal{P}(\mathcal{A})$, for each $N \in \mathbb{N}$ we consider *Problem* B_{π^N}, namely problem (4.6) relative to the distribution π^N. Recalling the definition of the value function in (4.8), we define

$$V^N(t_0, x_0) \coloneqq V_{\pi^N}(t_0, x_0) = \inf_{u \in \mathcal{U}_{t_0}} J_{\pi^N, t_0, x_0}[u] \, . \tag{4.20}$$

If $\{\pi^N\} \subset \mathcal{P}(\mathcal{A})$ is such that $W_1(\pi^N, \pi^\infty) \to 0$ for $N \to \infty$, where W_1 is the 1-Wasserstein distance defined in (3.6), what can be said about the convergence of the value functions V^N to V^∞ and of the optimal controls u^N to u^∞ for $N \to \infty$? In this section we give an answer to those questions.

4.3.1 Convergence of the value function

To begin with, we show that the value function V_π is Lipschitz continuous with respect to the probability distribution π. As an application, we will have the convergence of the value functions V^N to V^∞ when $W_1(\pi^N, \pi^\infty) \to 0$.

Theorem 4.13 (Lipschitz estimate for the value function w.r.t. π)**.** *Let the assumption **(SH)** be satisfied. Given π, $\pi' \in \mathcal{P}(\mathcal{A})$ and $(t_0, x_0) \in [0,T] \times K$ with $K \subset \mathbb{R}^n$ compact set, let us consider the two value functions V_π and $V_{\pi'}$ as defined in* (4.8)*. Then the distance between V and V' can be bounded uniformly for $(t_0, x_0) \in [0,T] \times K$, that is:*

$$\|V_\pi - V_{\pi'}\|_{\infty,[0,T]\times K} \le C_K\, W_1(\pi, \pi') \ , \tag{4.21}$$

where $C_K = C_K(T, C_A, \|Q\|_2, \|Q_f\|_2, r_1, K)$ is a constant which does not depend on the distributions π and π', but merely on the compact set $\mathcal{A}$.

Proof. The proof is similar to that of Theorem 3.3. We divide it into three steps.

STEP 1: Cost difference between two single trajectories.
Fix two matrices $A, A' \in \mathcal{A}$, any point $(t_0, x_0) \in [0,T] \times \mathbb{R}^n$ and a control $u \in \mathcal{U}_{t_0}$. Using Grönwall's Lemma as we did for point (ii) of Lemma 4.4, we get the following estimate:

$$\begin{aligned} |x_A(t) - x_{A'}(t)| \le & C_x^u\,(t-t_0)\, e^{\|A\|_2 (t-t_0)}\ \|A - A'\|_2 \\ \le & C_x^u\, t\, e^{\|A\|_2 t}\ \|A - A'\|_2 \qquad\qquad \forall\, t \in [t_0, T] \ , \end{aligned}$$

with C_x^u given by Lemma 4.4.
Let us denote by ℓ the running cost and by h the final cost:

$$\ell(x,u) \coloneqq \frac{1}{2}\left(x^T Q x + u^T R u\right) \quad \text{and} \quad h(x) \coloneqq \frac{1}{2} x^T Q_f x \ ,$$

so we can write

$$J_{\pi,t_0,x_0}[u] = \int_{\mathcal{A}} \left[\int_{t_0}^T \ell(x_A(t), u(t))\, dt + h(x_A(T)) \right] d\pi(A) \ .$$

Since both ℓ and h are locally Lipschitz continuous, one has

$$\begin{aligned} |\ell(x_{A'}(t), u(t)) - \ell(x_A(t), u(t))| &= \frac{1}{2} |{x_{A'}}^T(t) Q x_{A'}(t) - x_A^T(t) Q x_A(t)| \\ &\le \frac{1}{2} |{x_{A'}}^T Q (x_{A'}(t) - x_A(t))| + \frac{1}{2} |(x_{A'}(t) - x_A(t))^T Q x_A(t)| \\ &\le \|Q\|_2\, C_x |(x_{A'}(t) - x(t))| \le L_\ell^u\, C_x^u\, t\, e^{\|A\|_2 t}\ \|A - A'\|_2 \ , \end{aligned}$$

and, similarly,

$$|h(x_{A'}(T)) - h(x_A(T))| \le L_h^u\, C_x^u\, T\, e^{\|A\|_2 T}\ \|A - A'\|_2 \ ,$$

where we defined the Lipschitz constants

$$L_\ell^u \coloneqq \|Q\|_2\, C_x^u \quad \text{and} \quad L_h^u \coloneqq \|Q_f\|_2\, C_x^u \, ;$$

these two constants inherits from C_x^u the dependency on x_0 and u.
Finally, the cost difference between two single trajectories can be easily bounded by

$$\begin{aligned}\int_{t_0}^{T} |\ell(x_{A'}(t),u(t)) - \ell(x_A(t),u(t))|dt + |h(x_{A'}(T)) - h(x_A(T))| \\ \leq (TL_\ell^u + L_h^u)\, C_x^u T\, e^{C_A T}\ \|A - A'\|_2\ .\end{aligned} \tag{4.22}$$

STEP 2: Cost difference between two probability distributions.
Fix an initial condition $x(t_0) = x_0 \in \mathbb{R}^n$ with $0 \leq t_0 < T$ and a control $u \in \mathcal{U}_{t_0}$. We want to prove a bound for the distance between $J_{\pi,t_0,x_0}[u]$ and $J_{\pi',t_0,x_0}[u]$.
As a property of W_1, there exists (see Theorem 4.1 on [141]) a probability distribution $\gamma^* \in \Gamma(\pi,\pi')$ on $\mathcal{A}\times\mathcal{A}$ with marginal distributions π and π' such that

$$W_1(\pi,\pi') = \int_{\mathcal{A}\times\mathcal{A}} d_2(A,A')\, d\gamma^*(A,A')\ , \tag{4.23}$$

where $d_2(A,A') = \|A - A'\|_2$ is the distance between two matrices $A, A' \in \mathbb{R}^{n\times n}$ induces by the 2-norm. We can thus write

$$\begin{aligned}|J_{\pi,t_0,x_0}[u] - J_{\pi',t_0,x_0}[u]| &= \left| \int_{\mathcal{A}}\int_{t_0}^{T} \ell(x_A(t),u(t))\, dtd\pi(A) - \int_{\mathcal{A}}\int_{t_0}^{T} \ell(x_{A'}(t),u(t))dtd\pi'(A') \right. \\ &\quad \left. + \int_{\mathcal{A}} h(x_A(T))d\pi(A) - \int_{\mathcal{A}} h(x_{A'}(T))d\pi'(A') \right| \\ &= \left| \int_{\mathcal{A}\times\mathcal{A}} \left[\int_{t_0}^{T} (\ell(x_A(t),u(t)) - \ell(x_{A'}(t),u(t)))\ dt \right.\right. \\ &\quad \left.\left. + h(x_A(T)) - h(x_{A'}(T)) \right] d\gamma^*(A,A') \right|,\end{aligned}$$

where we have used that

$$\int_{\mathcal{A}\times\mathcal{A}} (\varphi(A) + \psi(A'))\, d\gamma^*(A,A') = \int_{\mathcal{A}} \varphi(A)d\pi(A) + \int_{\mathcal{A}} \psi(A')d\pi'(A')$$

for all measurable functions φ, ψ on $\mathcal{A}$, since γ^* admits π and π' as marginals.

Using the bound (4.22) from STEP 1 and formula (4.23), we get

$$\begin{aligned}|J_{\pi,t_0,x_0}[u] - J_{\pi',t_0,x_0}[u]| &\leq \int_{\mathcal{A}\times\mathcal{A}} (TL_\ell^u + L_h^u)\ C_x^u\, T\, e^{C_A T}\ \|A - A'\|_2\ d\gamma^*(A,A') \\ &= (TL_\ell^u + L_h^u)\, C_x^u\, T\, e^{C_A T}\, W_1(\pi,\pi')\ .\end{aligned} \tag{4.24}$$

Note that the constant C_x^u which appears here depends merely on x_0 and u.

STEP 3: Lipschitz estimate for the value function.
We will now show that an estimate similar to (4.24) holds true even for the value functions V_π and $V_{\pi'}$.
Fix any point $(t_0,x_0) \in [0,T]\times K$. In view of Lemma 4.5, there exist controls $\bar{u}, \bar{u}' \in \mathcal{U}_{t_0}$ such that

$$J_{\pi,t_0,x_0}[\bar{u}] = V_\pi(t_0,x_0), \qquad J_{\pi',t_0,x_0}[\bar{u}'] = V_{\pi'}(t_0,x_0)\,.$$

Then one has

$$V_{\pi'}(t_0,x_0) - V_\pi(t_0,x_0) = \inf_{u\in\mathcal{U}_{t_0}} J_{\pi',t_0,x_0}[u] - J_{\pi,t_0,x_0}[\bar{u}] \le J_{\pi',t_0,x_0}[\bar{u}] - J_{\pi,t_0,x_0}[\bar{u}]$$

and, in the same way,

$$V_\pi(t_0,x_0) - V_{\pi'}(t_0,x_0) = \inf_{u\in\mathcal{U}_{t_0}} J_{\pi,t_0,x_0}[u] - J_{\pi',t_0,x_0}[\bar{u}] \le J_{\pi,t_0,x_0}[\bar{u}'] - J_{\pi',t_0,x_0}[\bar{u}'] \ .$$

Hence

$$|V_\pi(t_0,x_0) - V_{\pi'}(t_0,x_0)| \le \max\left\{\left|J_{\pi',t_0,x_0}[\bar{u}] - J_{\pi,t_0,x_0}[\bar{u}]\right|, \left|J_{\pi',t_0,x_0}[\bar{u}'] - J_{\pi,t_0,x_0}[\bar{u}']\right|\right\}.$$

Moreover, being both $\bar{u}$ and $\bar{u}'$ optimal control for some distribution on $\mathcal{A}$, we can use the uniform constant $\overline{C}_x$ given by Remark 4.6 and define an analogous uniform constants for the Lipschitz continuity of ℓ and h:

$$\overline{L}_\ell \coloneqq \|Q\|_2\, \overline{C}_x\,, \quad \overline{L}_h \coloneqq \|Q_f\|_2\, \overline{C}_x \ .$$

In this way, the estimate becomes independent from $\bar{u}$ and $\bar{u}'$ and thus from π and π':

$$|V_{\pi'}(t_0,x_0) - V_\pi(t_0,x_0)| \le C_{x_0}\, W_1(\pi,\pi') \ ,$$

where

$$C_{x_0} \coloneqq \left(T\,\overline{L}_\ell + \overline{L}_h\right)\overline{C}_x T\, e^{C_A T} \ .$$

Finally, noting that the estimate depends on x_0 only through its norm, we get that the bound is uniform in compact sets $K \subset \mathbb{R}^n$, letting

$$C_K \coloneqq \sup_{x_0\in K} C_{x_0} \ . \qquad \square$$

A straightforward consequence of the previous Theorem is the following convergence result for the value function V^N.

Corollary 4.14 (Convergence of the value function)**.** *If $\pi^N \xrightarrow{*} \pi^\infty$, then $V^N(t_0,x_0) \to V^\infty(t_0,x_0)$ for each $(t_0,x_0) \in [0,T]\times\mathbb{R}^n$. The convergence is uniform in compact sets $[0,T]\times K$, where $K\subset\mathbb{R}^n$ is compact.*

4.3.2 Convergence of the optimal control

In what follows, we will use $\bar{u}^N(\cdot)$ to denote the optimal control of *Problem* B_{π^N}. Furthermore, $\bar{x}^N_A(\cdot)$ and $p^N_A(\cdot)$ denote, respectively, the optimal trajectories and the multipliers relative to *Problem* B_{π^N}, that is the solutions of the differential systems (4.9) and (4.19).

The following theorem provides a strong convergence of $\bar{u}^N(\cdot)$ to the optimal control $\bar{u}^\infty(\cdot)$ of the limit problem B_{π^∞}, assuming that $\pi^N \xrightarrow{*} \pi^\infty$.

Theorem 4.15 (Convergence of the optimal control)**.** *Let the assumption **(SH)** be satisfied. Consider a sequence of probability distributions $\{\pi^N\} \subset \mathcal{P}(\mathcal{A})$, such that $\pi^N \xrightarrow{*} \pi^\infty$ and fix $0 \le t_0 < T$ and $x_0 \in \mathbb{R}^n$. If $\bar{u}^N(\cdot)$ and $\bar{u}^\infty(\cdot)$ are respectively the optimal controls of Problem B_{π^N} and B_{π^∞}, i.e. they satisfy* (4.10) *respectively for π^N and π^∞, then $\bar{u}^N(\cdot)$ converges uniformly to $\bar{u}^\infty(\cdot)$ for $N\to\infty$.*

Proof. Without loss of generality, one can take $t_0 = 0$, being all the other cases similar.
Lemma 4.5 assures that, for each $\pi^N \in \mathcal{P}(\mathcal{A})$, there exists a unique optimal process $\left\{(\bar{x}_A^N, \bar{u}^N)(\cdot) : \ A \in \mathcal{A}\right\}$. Taking into account Theorem 4.10 and Remark 4.11, that process satisfies the following necessary conditions: for each $N \in \mathbb{N}$, there exists a continuous function $p^N \colon [0,T] \times \mathcal{A} \to \mathbb{R}^n$, $p^N(t,A) \equiv p_A^N(t)$, such that

(i)
$$p_A^N(\cdot) \in W^{1,1}([0,T];\mathbb{R}^n) \qquad \forall\, A \in \mathcal{A}\,;$$

(ii)
$$\bar{u}^N(t) = +R^{-1}B^T \int_{\mathcal{A}} p_A^N(t)\, d\pi^N(A), \qquad \forall\, t \in [0,T]\ ;$$

(iii)
$$-\dot{p}_A^N(t) = A^T p_A^N(t) - Q\bar{x}_A^N(t) \qquad a.e.\ t \in [0,T],\ \forall A \in \mathcal{A}\,;$$

(iv)
$$-p_A^N(T) = Q_f \bar{x}_A^N(T) \qquad \forall A \in \mathcal{A}\,.$$

For each $A \in \mathcal{A}$, the family of functions

$$\mathcal{F}_A \coloneqq \left\{\left(\bar{x}_A^N, p_A^N\right)(\cdot)\right\}_{N\in\mathbb{N}}$$

is uniformly bounded due to Lemma 4.4 and Lemma 4.12. Moreover, one can find the bounds on the derivatives:

$$\begin{aligned} \int_0^T |\dot{x}_A^N(t)|dt &= \int_0^T \left|A\bar{x}_A^N(t) + B\bar{u}^N(t)\right| dt \le TC_A\overline{C}_x + \sqrt{T}\,\|B\|_2\,\overline{C}_u, \\ |\dot{p}_A^N(t)| &= |A^T p_A^N(t) - Q\bar{x}_A^N(t)| \le C_A\overline{C}_p + \|Q\|_2\,\overline{C}_x\,, \end{aligned} \tag{4.25}$$

for all $A \in \mathcal{A}$, a.e. $t \in [0,T]$. The second bound in (4.25) and the relation (ii), imply that also the map

$$[0,T] \ni t \mapsto \bar{u}^N(t) = +R^{-1}B^T \int_{\mathcal{A}} p_A^N(t)\, d\pi^N(A)$$

is equibounded and equicontinuous in N. For each $A \in \mathcal{A}$ fixed, one can then apply Theorem 2.5.3 on [143], implying the existence of some limit functions $x_A^\infty, p_A^\infty \in W^{1,1}([0,T];\mathbb{R}^n)$ and $u^\infty \in C([0,T];\mathbb{R}^m)$ such that

$$\begin{gathered} \dot{x}_A^N(t) \rightharpoonup \dot{x}_A^\infty(t) \text{ and } \dot{p}_A^N(t) \rightharpoonup \dot{p}_A^\infty(t) \quad \text{weakly in } L^1([0,T];\mathbb{R}^n) \text{ as } N \to \infty\ , \\ \bar{x}_A^N(t) \to x_A^\infty(t) \quad \text{and} \quad p_A^N(t) \to p_A^\infty(t) \quad \text{uniformly on } [0,T] \text{ as } N \to \infty\ , \\ \bar{u}^N(t) \to u^\infty(t) \quad \text{uniformly on } [0,T] \text{ as } N \to \infty\ , \end{gathered}$$

and such that, for each $A \in \mathcal{A}$, $(x_A^\infty, p_A^\infty)(\cdot)$ is a solution of the boundary value problem

$$\begin{cases} \dot{x}_A^\infty(t) = Ax_A^\infty(t) + Bu^\infty(t), & t \in [0,T] \\ -\dot{p}_A^\infty(t) = A^T p_A^\infty(t) - Qx_A^\infty(t), & t \in [0,T] \\ x_A^\infty(0) = x_0 \\ -p_A^\infty(T) = Q_f x_A^\infty(T)\,. \end{cases} \tag{4.26}$$

Furthermore, since $\pi^N \overset{*}{\rightharpoonup} \pi^\infty$, u^∞ satisfies the relation

$$[0,T] \ni t \mapsto u^\infty(t) = +R^{-1}B^T \int_{\mathcal{A}} p_A^\infty(t)\, d\pi^\infty(A). \tag{4.27}$$

In fact, the result follows from the estimate

$$\begin{aligned}\left|\int_{\mathcal{A}} p_A^N(t)d\pi^N(A) - \int_{\mathcal{A}} p_A^\infty(t)d\pi^\infty(A)\right| \le &\left|\int_{\mathcal{A}} p_A^N(t)d\pi^N(A) - \int_{\mathcal{A}} p_A^N(t)d\pi^\infty(A)\right| \\ &+ \left|\int_{\mathcal{A}} p_A^N(t)d\pi^\infty(A) - \int_{\mathcal{A}} p_A^\infty(t)d\pi^\infty(A)\right|\end{aligned} \tag{4.28}$$

for all $t \in [0,T]$, which implies that

$$\bar{u}^N(t) = R^{-1}B^T \int_{\mathcal{A}} p_A^N(t)d\pi^N(A) \to u^\infty(t) = R^{-1}B^T \int_{\mathcal{A}} p_A^\infty(t)d\pi^\infty(A)$$

uniformly on $[0,T]$ as $N \to \infty$.

Notice that the convergence is guaranteed along a subsequence, but one can say that the whole sequence converges since the limit does not depend on the subsequence (it solves (4.26)).
It remains to show that the limiting process $\{(x_A^\infty, u^\infty)(\cdot)\colon A \in \mathcal{A}\}$ is actually optimal for the Problem B (4.6) relative to π^∞. To this aim, let us stress the following properties of the cost functional $J_{\pi,0,x_0}$ in (4.7), using the lighter notation

$$J^N \coloneqq J_{\pi^N,0,x_0} \quad \text{and} \quad J^\infty \coloneqq J_{\pi^\infty,0,x_0} \ :$$

1) if $\pi^N \overset{*}{\rightharpoonup} \pi^\infty$, $J^N[u] \to J^\infty[u]$ for each $u \in \mathcal{U}_0$, since the map $A \mapsto x_A(\cdot)$ is continuous by Lemma 4.4;

2) each J^N is continuous with respect to u, since for each $A \in \mathcal{A}$, the map $u \mapsto x_A(\cdot;u)$ is continuous, again from Lemma 4.4.

Since $\bar{u}^N$ is the optimal control of Problem B_{π^N} and u is an admissible control for the same problem, then we get

$$J^N[\bar{u}^N] \le J^N[u], \qquad \forall N \in \mathbb{N}$$

so, letting $N \to \infty$, it easily follows from the previous relation and properties 1) and 2) that

$$J^\infty[u^\infty] \le J^\infty[u] \ .$$

In view of the uniqueness of the optimal control $\bar{u}^\infty$ for Problem B_{π^∞}, one can conclude that $u^\infty \equiv \bar{u}^\infty$. Hence also the process $\{(x_A^\infty, u^\infty)(\cdot) :\ A \in \mathcal{A}\}$ is optimal for the given problem. This concludes the proof. □

Remark 4.16 (Special Case: $\pi = \delta_{\widehat{A}}$). The particular case in which π is a Dirac delta $\delta_{\widehat{A}}$ for a given matrix $\widehat{A} \in \mathbb{R}^{n\times n}$ deserves special attention. Indeed, when $\pi = \delta_{\widehat{A}}$, the cost functional J_{π,t_0,x_0} (4.7) becomes the cost functional

$$J_{t_0,x_0}[u] \coloneqq \frac{1}{2}\int_{t_0}^T \Big(x(t)^T Q x(t) + u(t)^T R u(t)\Big)\, dt + \frac{1}{2}x(T)^T Q_f x(T) \ ,$$

and Problem B in (4.6) coincides with a standard Problem A (see (4.1))

$$\begin{cases} \text{minimize } \left\{ \frac{1}{2} \int_{t_0}^{T} \Big(x(t)^T Q x(t) + u(t)^T R u(t) \Big)\, dt + \frac{1}{2} x(T)^T Q_f x(T) \right\} \\ \text{over } (x,u)(\cdot) \text{ such that } u : [t_0, T] \to \mathbb{R}^m \text{ is measurable and} \\ \dot{x}(t) = \widehat{A} x(t) + B u(t), \qquad t \in [t_0, T] \\ x(t_0) = x_0 \ . \end{cases} \tag{4.29}$$

Furthermore, the definition of the value function and of the optimal control we gave in (4.8) and (4.10) agree, in this special case, with the classic definitions in control theory (see (4.4) and (4.5)):

$$V(t_0, x_0) \coloneqq \inf_{u \in \mathcal{U}_{t_0}} J_{t_0,x_0}[u] \quad \text{and} \quad \bar{u} \coloneqq \arg\min_{u \in \mathcal{U}_{t_0}} J_{t_0,x_0}[u] \, .$$

If we apply Corollary 4.14 and Theorem 4.15 to a sequence π^N converging to $\delta_{\widehat{A}}$, then we obtain

$$V^N(t_0, x_0) \to V(t_0, x_0) \qquad \forall\, t_0 \in [0, T],\ x_0 \in \mathbb{R}^n$$

and

$$\bar{u}^N \to \bar{u} \qquad \text{uniformly in } [0, T] \, ,$$

where V^N and $\bar{u}^N$ are respectively the value function in (4.8) and the optimal control (4.10) relative to π^N.

4.4 On finite support measures converging to $\delta_{\widehat{A}}$

In this section, we will assume that $\mathcal{A}$ is a finite set, namely $\mathcal{A} \coloneqq \{A_1, \dots, A_M\}$ for some integer $M \in \mathbb{N}$. Let us consider a sequence of probability distributions $\{\pi^N\} \subset \mathcal{P}(\mathcal{A})$, that can be written as

$$\pi^N \coloneqq \sum_{i=1}^{M} \alpha_i^N \delta_{A_i}, \ \text{ for some } \ \alpha_i^N \geq 0 \ \text{ such that } \ \sum_{i=1}^{M} \alpha_i^N = 1, \quad \forall N \in \mathbb{N}. \tag{4.30}$$

For a given $t_0 \in [0, T]$ and $x_0 \in \mathbb{R}^n$, suppose that the underlying dynamics governing the optimal control problem is a standard Problem A (see (4.1)):

$$\begin{cases} \text{minimize } J_{t_0,x_0}[u] \\ \text{over } (x,u)(\cdot) \text{ such that } u : [t_0, T] \to \mathbb{R}^m \text{ is measurable and} \\ \dot{x}(t) = \widehat{A} x(t) + B u(t), \qquad t \in [t_0, T] \\ x(t_0) = x_0 \ , \end{cases} \tag{4.31}$$

where $\widehat{A} \in \mathcal{A}$ and the cost functional $J_{t_0,x_0}[u]$ is defined as in (4.2):

$$J_{t_0,x_0}[u] \coloneqq \frac{1}{2} \int_{t_0}^{T} \Big(x(t)^T Q x(t) + u(t)^T R u(t) \Big)\, dt + \frac{1}{2} x(T)^T Q_f x(T) \ . \tag{4.32}$$

Without loss of generality, one can set $\widehat{A} \equiv A_1$. For each $t_0 \in [0,T]$ and $x_0 \in \mathbb{R}^n$, the value function $V(t_0, x_0)$ for this problem has been defined in (4.4).

One can expect that, after some interactions with the system, it would be possible to construct a sequence of probability distributions $\{\pi^N\} \subset \mathcal{P}(\mathcal{A})$ capturing the current belief that one has about the real system (4.31) and such that, when N is sufficiently large, π^N gets closer and closer to δ_{A_1} and eventually $\pi^N \overset{*}{\rightharpoonup} \delta_{A_1}$.

For each fixed $N \in \mathbb{N}$, one can reformulate Problem B associated to π^N as a classical LQR problem on an augmented system of dimension nM:

$$\begin{cases} \text{minimize } J^N_{t_0,X_0}[u] \\ \text{over } (X,u)(\cdot) \text{ such that } u:[t_0,T]\to\mathbb{R}^m \text{ is measurable and} \\ \dot{X}(t) = \widetilde{A}X(t) + \widetilde{B}u(t), \qquad t \in [t_0,T] \\ X(t_0) = X_0\ , \end{cases} \tag{4.33}$$

with cost functional

$$J^N_{t_0,X_0}[u] \coloneqq \frac{1}{2}\int_{t_0}^{T} \Big(X(t)^T \widetilde{Q}^N X(t) + u(t)^T R u(t) \Big)\, dt + \frac{1}{2} X(T)^T \widetilde{Q}^N_f X(T)\ . \tag{4.34}$$

where we have used the compact notation $X(t) \coloneqq (x_{A_1}(t), \dots, x_{A_M}(t)) \in \mathbb{R}^{nM}$ and $X_0 \in \mathbb{R}^{nM}$ is made by vector x_0 repeated M times. The matrices of the augmented system are defined as

$$\widetilde{A} = \begin{pmatrix} A_1 & 0 & \cdots & 0 \\ 0 & A_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & A_M \end{pmatrix}, \quad \widetilde{B} = \begin{pmatrix} B \\ B \\ \vdots \\ B \end{pmatrix},$$

$$\widetilde{Q}^N = \begin{pmatrix} \alpha_1^N Q & 0 & \cdots & 0 \\ 0 & \alpha_2^N Q & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \alpha_M^N Q \end{pmatrix} \text{ and } \widetilde{Q}^N_f = \begin{pmatrix} \alpha_1^N Q_f & 0 & \cdots & 0 \\ 0 & \alpha_2^N Q_f & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \alpha_M^N Q_f \end{pmatrix}.$$

In this section, we will use $V^N(t_0, X_0)$ to denote the value function related to the LQR problem (4.33), namely

$$V^N(t_0, X_0) = \inf_{u \in \mathcal{U}_{t_0}} J^N_{t_0,X_0}[u]. \tag{4.35}$$

Since the optimal control problem (4.33) can be seen as a classic LQR problem, then one has the following relation between the value function and the optimal control in feedback form (see Section 2.1.2 and Remark 2.15): there exists P^N such that

$$V^N(t_0, X_0) = X_0^T P^N(t_0) X_0, \quad \nabla_X V^N(t_0, X_0) = 2P^N(t_0)X_0 \tag{4.36}$$

$$\bar{u}^N(s, X_0) = -R^{-1}\widetilde{B}^T P^N(t_0) X_0, \tag{4.37}$$

where $[t_0, T] \ni t \mapsto P^N(t) \in \mathbb{R}^{nM \times nM}$ solves the Riccati equation

$$\begin{cases} \widetilde{A}^T P(t) + P(t)\widetilde{A} - P(t)\widetilde{B}R^{-1}\widetilde{B}^T P(t) + \widetilde{Q}^N = -\dot{P}(t), & t \in [t_0, T] \\ P(T) = \widetilde{Q_f}^N \ . \end{cases} \tag{4.38}$$

For $x_0 \in \mathbb{R}^n$, we will use the notation $V^N(t_0, x_0)$ to denote the value function $V^N(t_0, X_0)$ evaluated at $X_0 = (x_0, \dots, x_0) \in \mathbb{R}^{nM}$.
Now we can summarize the results of the previous section applied to problem (4.33).

Corollary 4.17. *Let the assumption **(SH)** be satisfied. For each $0 \leq t_0 < T$, $x_0 \in \mathbb{R}^n$ and $\{\pi^N\} \subset \mathcal{P}(\mathcal{A})$, the optimal control problems* (4.33) *and* (4.31) *satisfy the following conditions:*

i) problem (4.33) *and* (4.31) *admit a unique optimal process $\left\{(\bar{x}_A^N, \bar{u}^N)(\cdot) : \ A \in \mathcal{A}\right\}$ and $(\bar{x}, \bar{u})(\cdot)$, respectively;*

ii) for each $K \subset \mathbb{R}^n$, $\exists\, C_K$ such that

$$||V^N - V||_{\infty,K} \leq C_K \, W_1(\pi^N, \delta_{\widehat{A}}) \ ; \tag{4.39}$$

iii) if, moreover, $\{\pi^N\}$ is such that $\pi^N \overset{*}{\rightharpoonup} \delta_{\widehat{A}}$, then the optimal control $\bar{u}^N \to \bar{u}$ uniformly for $t \in [t_0, T]$.

For each $N \in \mathbb{N}$, the solution of (4.38) is related to the matrices $X^N, Y^N : [t_0, T] \to \mathbb{R}^{nM \times nM}$, such that the pair $(X^N, Y^N)(\cdot)$ is the solution of the backward Hamiltonian differential equation

$$\begin{cases} \begin{bmatrix} \dot{X}(t) \\ \dot{Y}(t) \end{bmatrix} = H^N \begin{bmatrix} X(t) \\ Y(t) \end{bmatrix} & \text{for } t \in [t_0, T] \\ \begin{bmatrix} X(T) \\ Y(T) \end{bmatrix} = \begin{bmatrix} I \\ \widetilde{Q_f}^N \end{bmatrix} \ , \end{cases} \tag{4.40}$$

where

$$H^N = \begin{bmatrix} \widetilde{A} & -\widetilde{B}R^{-1}\widetilde{B}^T \\ \widetilde{Q}^N & -\widetilde{A}^T \end{bmatrix} \ . \tag{4.41}$$

The relation between the solutions of (4.38) and (4.40) was stated in precise terms by Coppel in [33, pp. 274-275]:

Theorem 4.18. *Suppose that **(SH)** holds true. Let $X, Y : [t_0, T] \to \mathbb{R}^{nM \times nM}$ be the solutions of the Hamiltonian differential problem* (4.40)*. Then*

1. $X(t)$ is non-singular for all $t \in [t_0, T]$;

2. the solution of (4.38) *is*

$$\widetilde{P}(t) = Y(t)X^{-1}(t), \qquad t \in [t_0, T] \ . \tag{4.42}$$

We are now ready to strengthen the results of the previous section by showing that, in the case in which the sequence of probability measures and its limit have finite support, then also the solution of the Riccati equation (4.38) has good convergence properties:

Theorem 4.19. *(Convergence of the Riccati equation) Let the assumption **(SH)** be satisfied. Suppose that the sequence $\{\pi^N\} \subset \mathcal{P}(\mathcal{A})$ is such that $\pi^N \overset{*}{\rightharpoonup} \delta_{A_1}$, that is, for each $i = 1, \dots, M$ the weights converge:*

$$\begin{aligned} &\alpha_1^N \to 1 \\ &\alpha_i^N \to 0 \qquad\qquad \text{for } i = 2, \dots, M. \end{aligned}$$

when $N \to \infty$. Then the sequence of matrices $\{\widetilde{P}^N(t)\} \subset \mathbb{R}^{nM \times nM}$ which solve the Riccati equation (4.38) for each $N \in \mathbb{N}$ converges to the matrix

$$\bar{P}(t) = \begin{pmatrix} P(t) & 0 & \cdots & 0 \\ 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 \end{pmatrix},$$

uniformly on $t \in [t_0, T]$, where $P(t) \in \mathbb{R}^{n \times n}$ is the solution of the Riccati equation related to the optimal control problem (4.31) with state matrix A_1, namely:

$$\begin{cases} A_1^T P(t) + P(t) A_1 - P(t) B R^{-1} B^T P(t) + Q = -\dot{P}(t), & t \in [t_0, T], \\ P(T) = Q_f \ , \end{cases} \tag{4.43}$$

Proof. Consider the Hamiltonian systems in (4.40) related to all different distributions π^N. Notice that the norm of H^N in (4.41) can be bounded:

$$\left\| H^N \right\|_2 \leq 2C_A + \|Q\|_2 + M \ \|B\|_2^2 \left\| R^{-1} \right\|_2 ,$$

for all $N \in \mathbb{N}$.

Using Grönwall's Lemma, one can easily show that the pair of matrices (X^N, Y^N) solution to (4.40) is uniformly bounded and that, using again (4.40), $(\dot{X}^N, \dot{Y}^N)$ is uniformly integrally bounded, for all $N \in \mathbb{N}$. So it is possible to apply Theorem 2.5.3 of [143] to show that the pair (X^N, Y^N) converges to some matrices (X^∞, Y^∞) solution of the system

$$\begin{cases} \begin{bmatrix} \dot{X}(t) \\ \dot{Y}(t) \end{bmatrix} = H^\infty \begin{bmatrix} X(t) \\ Y(t) \end{bmatrix} \qquad \text{for } t \in [t_0, T] \\ \begin{bmatrix} X(T) \\ Y(T) \end{bmatrix} = \begin{bmatrix} I \\ \widetilde{Q_f}^\infty \end{bmatrix} , \end{cases} \tag{4.44}$$

where

$$H^\infty = \begin{bmatrix} \widetilde{A} & -\widetilde{B} R^{-1} \widetilde{B}^T \\ \widetilde{Q}^\infty & -\widetilde{A}^T \end{bmatrix} \tag{4.45}$$

and

$$\widetilde{Q}^{\infty} = \begin{pmatrix} Q & 0 & \cdots & 0 \\ 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 \end{pmatrix}, \qquad \widetilde{Q}_f^{\infty} = \begin{pmatrix} Q_f & 0 & \cdots & 0 \\ 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 \end{pmatrix}.$$

Theorem 4.18 guarantees that the matrix $X^{\infty}(t)$ is also nonsingular for each $t \in [t_0, T]$, and the matrix $\widetilde{P}^{\infty}(t) \coloneqq Y^{\infty}(t)X^{\infty}(t)^{-1}$ is the solution of the Riccati equation

$$\begin{cases} \widetilde{A}^T\widetilde{P}(t) + \widetilde{P}(t)\widetilde{A} - \widetilde{P}(t)\widetilde{B}R^{-1}\widetilde{B}^T\widetilde{P}(t) + \widetilde{Q}^{\infty} = -\dot{\widetilde{P}}(t) & t \in [t_0, T] \\ \widetilde{P}(T) = \widetilde{Q}_f^{\infty}\ . \end{cases} \tag{4.46}$$

Since each $X^N(t)^{-1}$ is continuous and well defined for each $N \in \mathbb{N}$ and that $X^{\infty}(t)^{-1}$ is uniformly continuous on $[t_0, T]$, then

$$\widetilde{P}^N(t) \coloneqq Y^N(t)X^N(t)^{-1} \longrightarrow Y^{\infty}(t)X^{\infty}(t)^{-1} =: \widetilde{P}^{\infty}(t)$$

uniformly on $t \in [t_0, T]$. To conclude, consider the matrix $\bar{P}(t) \in \mathbb{R}^{nM \times nM}$,

$$\bar{P}(t) = \begin{pmatrix} P(t) & 0 & \cdots & 0 \\ 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 \end{pmatrix},$$

where $P(t) \in \mathbb{R}^{n \times n}$ is the unique solution of the Riccati equation (4.43). A direct verification shows that $\bar{P}(t) \in \mathbb{R}^{nM \times nM}$ also satisfies the Riccati equation (4.46). However the problem (4.46) admits a unique solution, implying that $\widetilde{P}^{\infty}(t) \equiv \bar{P}(t)$ for all $t \in [t_0, T]$. This ends the proof. □

Remark 4.20 (Feedback optimal controls). We would like to stress that the convergence of the Riccati matrix solution $\tilde{P}^N$ to P provided by Theorem 4.19 has the following implication: for each $(t_0, x_0) \in [0, T] \times \mathbb{R}^n$, we define

$$\begin{aligned} \bar{u}^N(t_0, x_0) &\coloneqq -R^{-1}\widetilde{B}^T P^N(t_0)(x_0, \ldots, x_0) \ \text{ and} \\ \bar{u}(t_0, x_0) &\coloneqq -R^{-1}B^T P(t_0)x_0, \end{aligned} \tag{4.47}$$

then $\bar{u}^N$ tend to $\bar{u}$ for N going to $+\infty$. Namely, that the optimal control of problem (4.33), which satisfies the formula (4.37), evaluated at $X_0 = (x_0, \ldots, x_0)$ converges pointwise to the optimal control of problem (4.31). Furthermore, for each $K \subset \mathbb{R}^n$ compact, one has that

$$\bar{u}^N(t_0, x_0) \to \bar{u}(t_0, x_0) \quad \text{uniformly on } [0, T] \times K. \tag{4.48}$$

It is important to point out that, whereas Theorem 4.15 proves the convergence for the class of optimal open-loop controls, Theorem 4.19 deals with the convergence of optimal controls in feedback form.

It remains an open question if Theorem 4.19 can be proved for a generic sequence of probability measures $\left\{\pi^N\right\}_{N \in \mathbb{N}} \subset \mathcal{P}(\mathcal{A})$ converging weakly-* to a generic probability measure $\pi \in \mathcal{P}(\mathcal{A})$. Such an issue is delicate and will be studied in the next future.

4.5 A numerical test

The aim of this section is to verify that the results summarized in Corollary 4.17 hold in a concrete example.

The model is the one presented in Section 4.4. The true dynamics is a controlled harmonic oscillator described by the matrices

$$\widehat{A} \coloneqq \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix} \quad \text{and} \quad B \coloneqq \begin{pmatrix} 0 \\ 1 \end{pmatrix} ;$$

the coefficients that are used to define the cost functional are

$$Q \coloneqq \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \quad \text{and} \quad R \coloneqq 0.1 \,,$$

and the final time is $T = 5$. Let us write π^N as in (4.30) for $M = 9$ and the 9 matrices are defined in a neighborhood of matrix $\widehat{A}$, i.e.

$$A_1 = \widehat{A} \,,$$

$$A_{2*j+i} \coloneqq \widehat{A} + (-1)^i \, 0.5 \, e_j \qquad i = 0, 1, \;\; j = 1, 2, 3, 4$$

where $\{e_j\}_{j=1,\dots,4}$ are the matrices of the canonical basis of $\mathbb{R}^{2\times 2}$. The probabilities α_i^N are defined according the following rule:

$$\alpha_1^N = 1 - \frac{1}{2^N} \,, \quad \alpha_i^N = \frac{1}{8} \frac{1}{2^N} \text{ for } i = 2, \dots, 9 \,.$$

Note that the Wasserstein distance with respect to the euclidean norm on $\mathbb{R}^{2\times 2}$ between π^N and $\delta_{\widehat{A}}$ can be computed exactly:

$$W_1(\pi^N, \delta_{\widehat{A}}) = \frac{1}{2^{N+1}} \,.$$

Both Problem A, that is the LQR problem with the matrix $\widehat{A}$ known, and Problem B, that is the LQR problem with the matrix $\widehat{A}$ unknown can be solved by finding the solution of a Riccati equation (see §IV.5 on Fleming-Rishel monograph [50]). We solved the equation numerically for $N = 0, \dots, 9$. For each N we computed the sup norm of the difference $\widetilde{V}^N - V$ and of the difference $\bar{u}^N - \bar{u}$, where $\bar{u}^N$ and $\bar{u}$ are respectively the optimal controls for the two problems starting from $x_0 = (1, 0)$. The results are summarized in Table 4.1.

Notice that when we increase N by one we halve the distance $W_1(\pi^N, \delta_{\widehat{A}})$ and Table 4.1 tells us that also the error $||\widetilde{V}^N - V||_{\infty,K}$, with $K \coloneqq [-2, 2]^2$, is halved; this is consistent with the estimate given by Corollary 4.17. At the same time, we remark that the error $||\bar{u}^N - \bar{u}||_\infty$ is halved as well, even if we didn't have any estimate on the convergence rate of the optimal controls. We can say that in this example both the errors are going to 0 with order 1.

The optimal trajectory of Problem A starting from $x_0 = (1, 0)$ is represented in Figure 4.1. For Problem B the optimal trajectory is actually made of 9 trajectories, the costs of which are weighted averaged in order to compute the cost functional

Table 4.1. Errors for value functions and optimal controls related to π^N with $N = 0, \dots, 9$ with respect to the true value function and the true optimal control of Problem A. The initial point for the optimal control is $x_0 = (1, 0)$.

N	α_1	$\|\|\widetilde{V}^N - V\|\|_{\infty,[0,5]\times K}$	*order*	$\|\|\bar{u}^N - \bar{u}\|\|_\infty$	*order*
0	0	6.08e-0	-	5.25e-1	-
1	0.5	3.21e-0	0.92	3.21e-1	0.71
2	0.75	1.66e-0	0.95	1.82e-1	0.82
3	0.875	8.49e-1	0.97	9.78e-2	0.90
4	0.9375	4.29e-1	0.98	5.09e-2	0.94
5	0.9687	2.16e-1	0.99	2.59e-2	0.97
6	0.9844	1.08e-1	1.00	1.31e-2	0.99
7	0.9922	5.42e-2	1.00	6.59e-3	0.99
8	0.9961	2.71e-2	1.00	3.30e-3	1.00
9	0.9980	1.36e-3	1.00	1.65e-3	1.00

$\widetilde{J}$. Two examples of optimal *(multi-)trajectory*, respectively for π^0 and π^2, are represented in Figure 4.2 and Figure 4.3; note that the trajectory related to the true dynamics is $x^1(t)$, which is the darkest one. Finally, in Figure 4.4 the optimal controls for Problem B with $N = 0, \dots, 4$ are compared with the true optimal control.

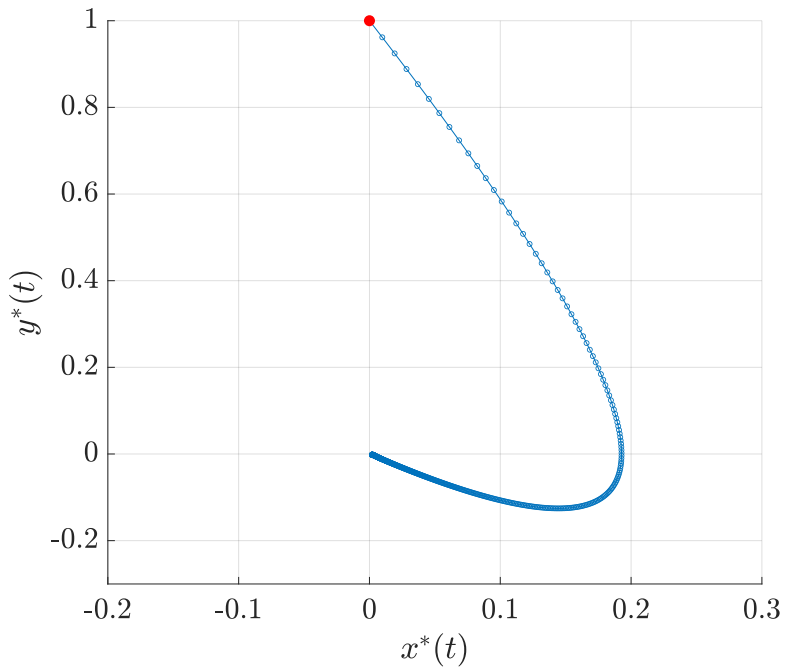


Figure 4.1. Optimal trajectory of Problem A, computed solving a 2-dimensional Riccati differential equation associated to matrix $\widehat{A}$. The initial point is indicated with a red dot.

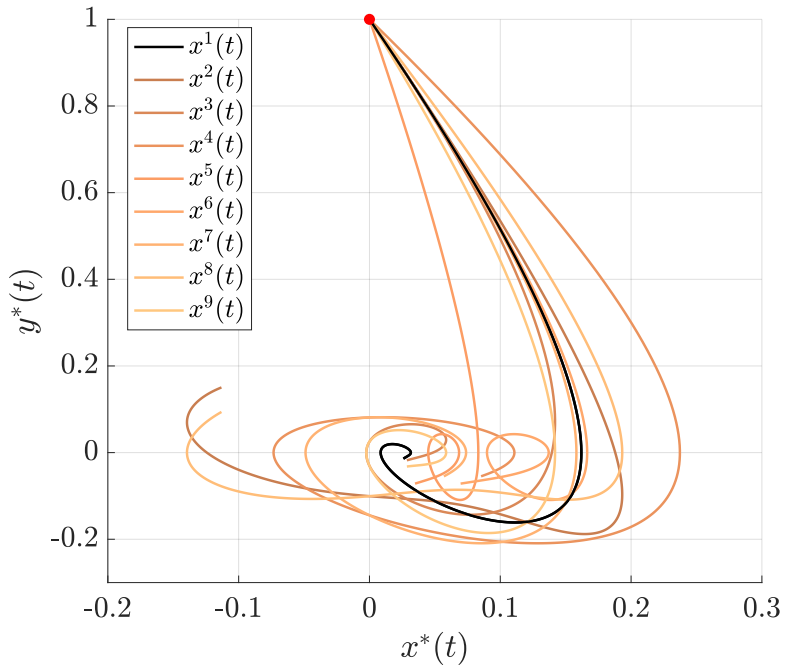


Figure 4.2. Optimal (multi-)trajectory for Problem B with π^0. To compute the optimal solution we solved a 18-dimensional Riccati differential equation. The initial point is indicated with a red dot.

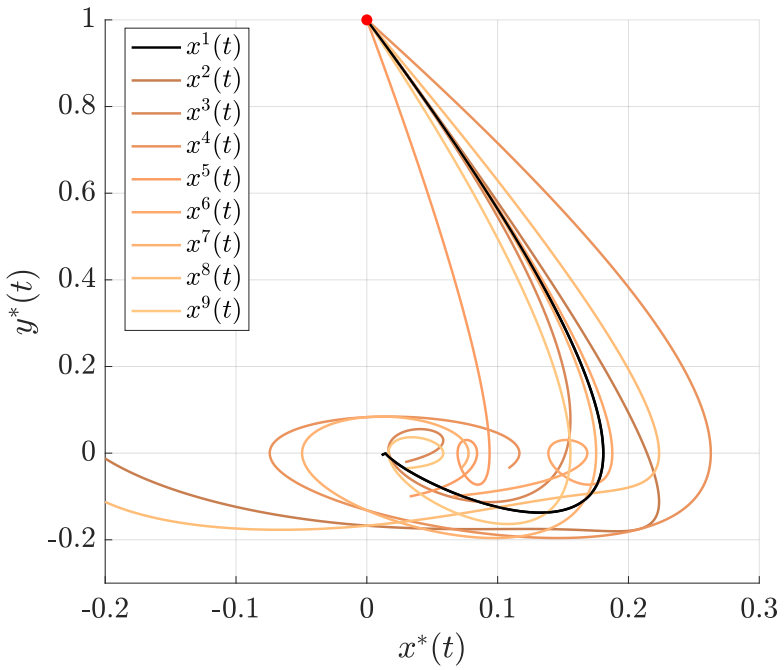


Figure 4.3. Optimal (multi-)trajectory for Problem B with π^2.

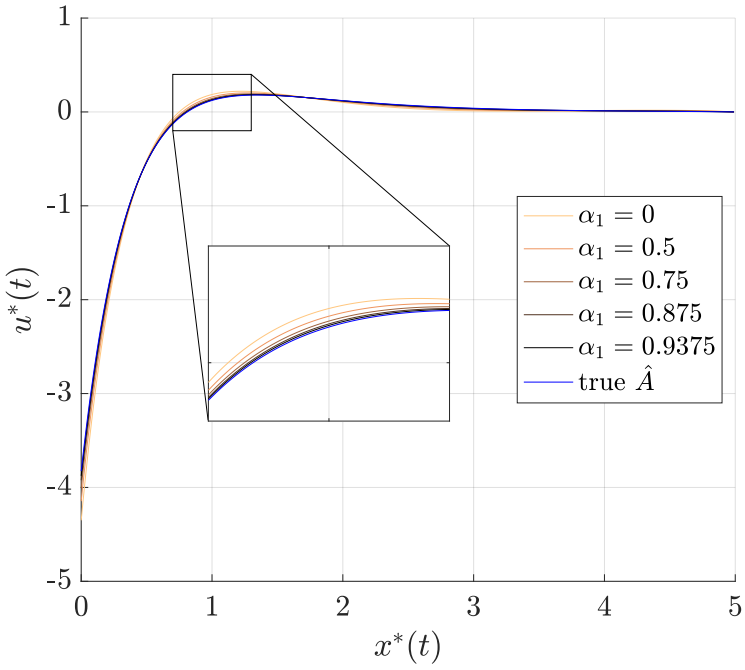


Figure 4.4. Comparison of the optimal controls of Problem B relative to different probability distributions π^N, with $N = 0, \dots, 4$, and the true optimal control of Problem A (in blue). In the legend we reported, for each N, the probability α_1 that the true matrix $A_1 \equiv \widehat{A}$ has under the distribution π^N. When $\alpha_1 \to 1$, the optimal control of Problem B converges to the true one.

Chapter 5

A new online algorithm for an LQR problem with partially unknown dynamics

In this chapter, we present a new algorithm for an LQR problem with partially unknown dynamics. Our setting here is slightly different from the one in Chapter 4. In fact, while in Chapter 4 a probability distribution describing our knowledge of $\widehat{A}$ was given, here we assume that the matrix $\widehat{A}$ is completely unknown at the beginning. When the dynamics is known, the optimal control is obtained in feedback form solving a backward Riccati differential equation (see Section 2.1.2). On the other hand, when the matrix $\widehat{A}$ is unknown, the LQR solution via the Riccati equation cannot be applied directly. We proposed a new *model-based* algorithm *to obtain an approximation of the dynamics and of the control at the same time* during a single simulation.

We already discussed the strong connection between RL and optimal control theory in the previous chapters. In particular, in Section 3.2 we identified some RL tasks as optimal control problems with unknown dynamics. We argue that these two research areas can greatly benefit from each other. Our algorithm fits well in this context, as it takes contributions from both fields since it can be considered as a case of Bayesian RL (cf. Section 2.4.3), but at the same time it borrows the LQR solution from optimal control theory to get the synthesis of a suitable control.

In particular, our algorithm is similar to PILCO [37], from which it takes the use of Gaussian distributions and the whole process of Bayesian update. However, we propose here some novelties. The first is that in PILCO the optimal control is chosen through a gradient descent algorithm in a class of controls, whereas we solve a Riccati differential equation to identify the minimizer. The second is that PILCO needs several trials to reconstruct the dynamics and stabilize the system. Our algorithm is designed to approximate the dynamics *and* to find a suitable control in a single run but can be applied only to linear dynamical systems. Finally, let us mention that other works have already studied LQR problems with unknown dynamics (see i.e. [52, 54, 88, 106] and references therein), but they all need several trials to converge, whereas our method works with just one. The contents of this chapter have been published in [104].

The chapter is structured as follows. In Section 5.1 we recall the LQR problem. In Section 5.2 we present our algorithm and discuss some implementation details. Finally, in Section 5.3 we show and discuss some numerical tests.

5.1 The classical LQR problem

We briefly recall the LQR problem (cf. Section 2.1.2) and state the standing assumptions for the algorithm. For $t_0 = 0$, the system dynamics is

$$\begin{cases} \dot{x}(t) = \widehat{A}x(t) + Bu(t), & t \in [0,T] \\ x(0) = x_0, \end{cases} \tag{5.1}$$

while the cost functional is

$$J_{x_0}[u] \coloneqq \frac{1}{2}\left(\int_0^T \left(x(t)^T Q x(t) + u(t)^T R u(t)\right) dt + x(T)^T Q_f x(T)\right). \tag{5.2}$$

We look for a control function $u(t) \in \mathbb{R}^m$ that minimizes J_{x_0} admissible controls $\mathcal{U} := \{u : [0,T] \to \mathbb{R}^m,\ \text{Lebesgue measurable}\}$. Our assumptions on the cost matrices are the following:

(SH): $Q, Q_f \in \mathbb{R}^{n\times n}$ are symmetric and semipositive definite and $R \in \mathbb{R}^{m\times m}$ is symmetric and positive definite.

When the dynamics is fully known, the optimal control is given by

$$u^*(t) = -R^{-1}B^T P(t)x(t) \qquad \forall\, t \in [t_0, T], \tag{5.3}$$

where $P(t)$ is the unique symmetric solution of a Riccati matrix differential equation

$$\begin{cases} -\dot{P}(t) = A^T P(t) + P(t)A - P(t)BR^{-1}B^T P(t) + Q, \quad t \in [t_0, T] \\ P(T) = Q_f. \end{cases} \tag{5.4}$$

We want to investigate: what happens if the dynamics is partially unknown? How can one find a suitable control? We considered the following framework:

Setting. *We assume that the matrices $B \in \mathbb{R}^{n\times m}$, $Q, Q_f \in \mathbb{R}^{n\times n}$ and $R \in \mathbb{R}^{m\times m}$ are given, whereas the state matrix $\widehat{A} \in \mathbb{R}^{n\times n}$ is unknown.*

5.2 Description of the algorithm

In this section, we describe our model-based algorithm able to solve the LQR problem without knowing matrix $\widehat{A}$. The algorithm is online, meaning that it doesn't need any previous simulation or computation. Our goal is twofold: first, we look for a good estimate for the unknown dynamics matrix $\widehat{A}$; and secondly, we want to choose a control that can steer the trajectory towards the origin. Furthermore, we assume that we can run the experiment just once, so the system must be controlled while the dynamics is still uncertain. To this end, we use a technique to get an estimate of the matrix $\widehat{A}$ and to update the control at the same time.

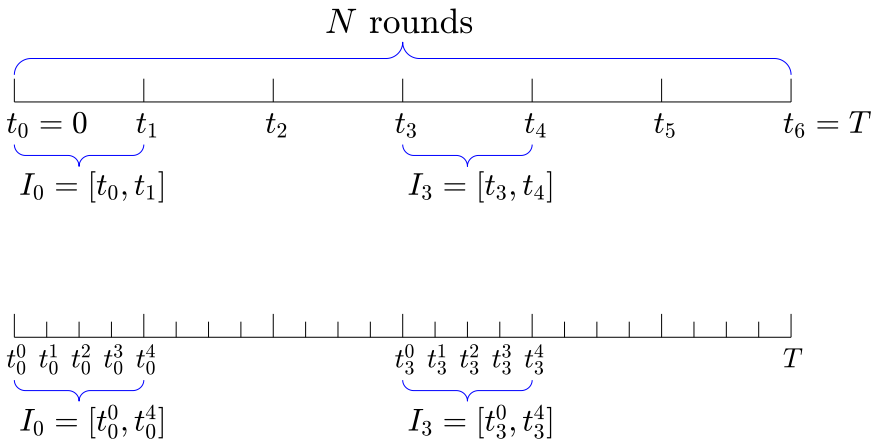


Figure 5.1. Time discretization for the proposed algorithm. The whole interval $[0, T]$ is divided into $N * S$ small time steps. These are grouped into N rounds of S small time steps. In this example, $N = 6$ and $S = 4$.

We divide the interval $[0, T]$ into equal time steps of length Δt, globally we will have $N * S$ small time steps, and we group them into N rounds of S steps each, as shown in Figure 5.1. The i-th round will be denoted by $I_i = [t_i, t_{i+1}]$ and a superscript index will indicate a single time step:

$$t_i^j = t_i + j\Delta t, \quad j = 0, \dots, S. \tag{5.5}$$

The current knowledge of the dynamics matrix $\widehat{A}$ is represented as a probability distribution over matrices. For each round, two major operations are carried out:

1. At the beginning of each round, a probability π_i is given from the previous round. We use the mean $\bar{A}_i$ of this distribution and compute a feedback control for the round solving a Riccati equation;

2. At the end of each round, the current probability distribution is updated using Bayesian formulas, according to the trajectory observed during the round. The output is a new probability distribution π_{i+1}.

The whole algorithm is summarized below as Algorithm 10. In the following, we will give more technical details.

5.2.1 Prior distribution

The algorithm requires the choice of a prior distribution π_0. We fix some $m_0 \in \mathbb{R}^n$ and $\Sigma_0 \in \mathbb{R}^{n \times n}$ and consider a random matrix A such that each of its rows r_k is distributed as an independent Gaussian vector with mean m_0 and covariance matrix

Algorithm 10 An online algorithm for a partially unknown LQR problem

Require: B, Q, Q_f and R

1: Divide $[0, T]$ into $N * S$ time steps of length Δt, with $\Delta t = \frac{T}{N*S}$;
2: Group the intervals in N *rounds*, each containing S time steps;
3: Choose a prior distribution π_0 over matrices;
4: **for** each *round* i from 0 to N-1 **do**
5: Find a feedback control u_i^* solving a Riccati equation with the mean matrix $\bar{A}_i$ of the distribution π_i
6: Use u_i^* as control for all the steps in the i-th *round*;
7: Observe the actual trajectory;
8: Update the distribution according to the data from the observed trajectory and compute π_{i+1}
9: **end**

Σ_0:

$$A = \begin{bmatrix} r_1^T \\ \vdots \\ r_n^T \end{bmatrix} \qquad r_k \sim \mathcal{N}(m_0, \Sigma_0) \qquad \forall\, k = 1, \ldots, n$$

A typical choice for the prior distribution, when no information about the true matrix $\widehat{A}$ is available, is $m_0 = (0, \ldots, 0)^T$ and $\Sigma_0 = n\, m\, I_n$. A high variance in the prior distribution results in little reliance on prior information, and is necessary when there is no information before the run. The choice of a Gaussian distribution as prior, together with a Gaussian likelihood function (5.13), ensures that the posterior distribution (5.14) is also Gaussian. A prior and a posterior distribution belonging to the same family of probability distributions are called *conjugate distributions*; such a prior is called a *conjugate prior* for the likelihood function. The use of conjugate priors is crucial in Bayesian regression, because it allows you to compute the posterior distribution with closed formulas, without the need for other calculations. Another famous conjugate prior for a Gaussian likelihood function is the Inverse Wishart distribution [102].

Finally, another possibility could be to consider a probability distribution on matrices, rather than on individual rows. However, this approach has the disadvantage of working with larger matrices. In fact, to work with a Gaussian distribution on matrices it would be necessary to save in memory a covariance matrix of dimension $n^2 \times n^2$, whereas when considering distributions on single rows it is sufficient to store n matrices of dimension $n \times n$. Furthermore, simple calculations show that the two approaches are equivalent if the noises on each component of y are considered independent.

5.2.2 Reconstruction of a feedback control at round I_i

At the beginning of the round $I_i = [t_i, t_{i+1}]$ our knowledge of the matrix $\widehat{A}$ is described by the distribution π_i. In order to find the control to apply, we solve the evolutive *Riccati* equation associated with the matrix $\bar{A}_i$, where $\bar{A}_i$ is the mean of

the distribution π_i. The Riccati equation reads

$$\begin{cases} -\dot{P}(t) = \bar{A}_i^T P(t) + P(t)\bar{A}_i - P(t)BR^{-1}B^T P(t) + Q \quad t \in [t_i, T] \\ P(T) = Q_f. \end{cases} \tag{5.6}$$

If we denote by $P_i(t)$ the solution of (5.6), our control will be the feedback control given by

$$u_i^*(t_i^j) = -R^{-1}B^T P_i(t_i^j) x_i^j \qquad j = 0, \dots S-1, \tag{5.7}$$

where $x_i^j = x(t_i^j) \in \mathbb{R}^n$ is the observed state of the system. Since the control must be defined for all instants $t \in [t_i, t_{i+1}]$, we choose a piecewise constant control: $u_i^*(t) = u_i^*(t_i^j) \in \mathbb{R}^m$ for $t \in [t_i^j, t_i^{j+1}]$. Notice that we need real-time observations of the system state to compute u_i^*, since it depends on the real trajectory $x(\cdot)$.

We emphasize the fact that in choosing the optimal control we do not take into account the uncertainty of the model, as happens in other probabilistic model-based RL algorithms (cf. Section 2.4.3), but we only use the mean $\bar{A}_i$ of the distribution π_i, according to the *"certainty equivalence"* principle of adaptive control theory (cf. Section 2.3.3).

5.2.3 Distribution update for the next round

At the end of each round, we can update the probability distribution based on the observed trajectory, using *Bayesian linear regression* formulas (see Section 2.3.2). More precisely, we know that during the time step $[t_i^j, t_i^{j+1}]$ the system evolves according to (5.1) where we plug in the chosen piecewise constant control, which is constantly equal to $u^*(t_i^j)$ for all $t \in [t_i^j, t_i^{j+1}]$. We can easily get an approximation of the state derivative with a finite difference scheme using the state observations $x_i^j = x(t_i^j) \in \mathbb{R}^n$. By a first-order scheme, rearranging the terms, we get

$$\widehat{A} x_i^j \simeq \frac{x_i^{j+1} - x_i^j}{\Delta t} - B u^*(t_i^j) \qquad j = 0, \dots, S-1. \tag{5.8}$$

For each $j = 0, \dots, S-1$, we interpret these data as observations of the dynamics with a Gaussian noise ε due to the error in the derivative approximation, which in this case is $\mathcal{O}(\Delta t)$, and in the measurements:

$$\widehat{A} x_i^j + \varepsilon = \frac{x_i^{j+1} - x_i^j}{\Delta t} - B u^*(t_i^j) \qquad \text{with } \varepsilon \sim \mathcal{N}(0, \sigma^2 I_S). \tag{5.9}$$

The standard deviation σ is a parameter to be chosen. When the measurement errors are negligible, we can choose σ according to the order of the derivative approximation, e.g. $\sigma = \mathcal{O}(\Delta t)$.
Denoting by y^j the right-hand side in (5.8),

$$y^j := \frac{x_i^{j+1} - x_i^j}{\Delta t} - B u^*(t_i^j), \tag{5.10}$$

we can rewrite equation (5.9) as

$$y^j = \widehat{A} x_i^j + \varepsilon \qquad \text{with } \varepsilon \sim \mathcal{N}(0, \sigma^2 I_S). \tag{5.11}$$

We treat and approximate each row $\widehat{r}_k$ of $\widehat{A}$ separately. The k-th component of y^j satisfies

$$y^j_{(k)} = \widehat{r}_k x^j_i + \varepsilon \qquad \text{with } \varepsilon \sim \mathcal{N}(0, \sigma^2), \tag{5.12}$$

for each $j = 0, \dots, S-1$. Equation (5.12) is of the same form as equation 2.76. We can therefore estimate the vector $\widehat{\theta} \equiv \widehat{r}_k$ using Bayesian linear regression. The likelihood function (cf. equation (2.77)) for the k-th component $y^j_{(k)}$ is thus Gaussian, and precisely

$$p(y^j_{(k)} | x^j_i, \widehat{r}_k) \sim \mathcal{N}(\widehat{r}_k x^j_i, \sigma^2). \tag{5.13}$$

For each row k we have a prior distribution $r_k \sim \mathcal{N}(m_0, \Sigma_0)$ (cf. equation (2.79)) given by π_i, the distribution from the previous round. We define X as the matrix whose columns are $x^0_i, \dots, x^{S-1}_i$ and y as the column vector $(y^0_{(k)}, \dots, y^{S-1}_{(k)})^T$. Applying BLR, we obtain a posterior distribution (cf. equations (2.80) and (2.81))

$$p(r_k | y, X) \sim \mathcal{N}(m, \Sigma), \tag{5.14}$$

where

$$\Sigma^{-1} = \frac{1}{\sigma^2} X X^T + \Sigma_0^{-1} \quad \text{and} \quad m = \Sigma(Xy/\sigma^2 + \Sigma_0^{-1} m_0). \tag{5.15}$$

5.2.4 Higher-order schemes

We already noticed that, if there are no significant measurement errors, the noise in (5.12) mainly depends on the precision in the derivative approximation in (5.8). Consequently, using more accurate schemes to approximate the derivative can provide more accurate approximations of the matrix $\widehat{A}$. For instance, we could use a second-order scheme or a fourth-order scheme, which are given respectively by

$$\dot{x}(t^j_i) \simeq \frac{x^{j+1}_i - x^{j-1}_i}{2\,\Delta t} \qquad \text{and} \qquad \dot{x}(t^j_i) \simeq \frac{-x^{j+2}_i + 8x^{j+1}_i - 8x^{j-1}_i + x^{j-2}_i}{12\,\Delta t}.$$

The interested reader can find more details on higher-order finite difference schemes in [133].

However, the use of high-order schemes has some drawbacks. First, trajectory regularity is required in the interval containing the nodes used in the approximation. While a first-order approximation uses only two nodes, higher-order approximations use more nodes, thus the control cannot jump at each time step, but rather we have to keep it constant for multiple steps. This forces the use of longer rounds, delaying the learning process. Second, as we will see in the numerical tests, this order only concerns the convergence of the estimate $\bar{A}_i$ to the true matrix $\widehat{A}$ as the round number i increases, whereas the cost of the trajectory produced by the algorithm converges only with numerical order 1.

5.2.5 Heuristic argument for convergence

The algorithm cannot find the *optimal* control for the problem, since at the beginning the matrix $\widehat{A}$ is unknown, and it needs at least some steps to have a good approximation for $\widehat{A}$. However, from Bayesian regression theory (see e.g. [119]) we know that the more "good" data we observe, the more precise our distribution π_i

becomes, eventually converging to the Dirac delta $\delta_{\widehat{A}}$. The choice of data is crucial, since they must provide new information with respect to previous data. This is why we speak of "good" data. We have observed experimentally that our algorithm always generates good data, except when the true matrix $\widehat{A}$ has some symmetries (cf. Section 5.3.4). In the latter cases, the rank of the trajectory matrix X may not be maximum and the information available may not be sufficient to reconstruct the matrix. In this case, the presence of some noise might help.

Let us now consider a case when the distribution π_i is converging to $\delta_{\widehat{A}}$. Then, also the mean matrix $\bar{A}_i$ of π_i is converging to $\widehat{A}$. Thus, after few rounds, the feedback control computed by the algorithm (which is using $\bar{A}_i$) in the interval $[t_i, T]$ should be close to the optimal control of a trajectory of the real dynamics which starts from the same point x_i^0. It should not be difficult to rigorously prove the convergence of the algorithm and possibly provide an error estimate, assuming that the matrix X has maximum rank. Notice that this argument is very close to the theory developed in Chapter 4. The difference with the framework presented there is that here we do not use our probabilistic model by solving an average cost problem, but we only use the average of the distribution, following the certainty equivalence principle.

Finally, when $\Delta t \to 0$, the algorithm reaches earlier a good estimate of the dynamics matrix. This means that also the computed control is closer to the optimal one. All these heuristics are confirmed by the numerical simulations in the next section.

5.3 Numerical tests

The following numerical tests for the algorithm described above were performed in MATLAB and took few seconds to run.

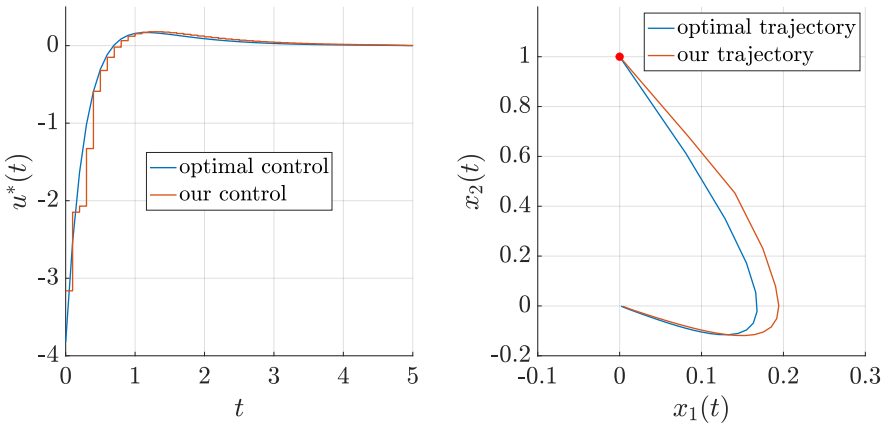
5.3.1 Test 1

We first consider a dynamical system where the state lies in $\mathbb{R}^2$ and the control is 1-dimensional, i.e. $n = 2$ and $m = 1$. The LQR problem is defined by the following matrices:

$$\widehat{A} = \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix} \; B = \begin{pmatrix} 0 \\ 1 \end{pmatrix} \; Q = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \; R = 0.1 \; Q_f = \begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix}.$$

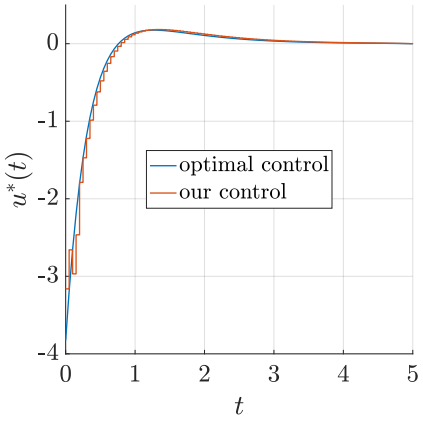
The time horizon is set to $T = 5$ and the starting point is $x_0 = (0, 1)^T$. We assume that the matrix $\widehat{A}$ is unknown to the algorithm, though we use it to simulate the dynamics, as if we were interacting with a real environment. We choose the prior distribution as recommended in Section 5.2, using $m_0 = (0, 0)^T$ and $\Sigma_0 = 2I_2$; for all the tests we set $\sigma = \sqrt{10\Delta t^p}$ and $S = 2p$, where p is the order of the scheme used in the derivative approximation.

Fig. 5.2 shows the *piecewise constant* controls chosen by the algorithm with $p = 1$ for different values of Δt and the corresponding trajectories. Recall that the matrix $\widehat{A}$ is completely unknown at the beginning, so the control we apply in the first steps depends only on the prior distribution we have chosen and clearly is not

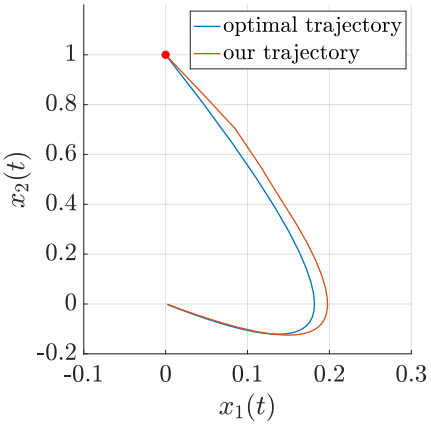


(a) control $\Delta t = 0.1$

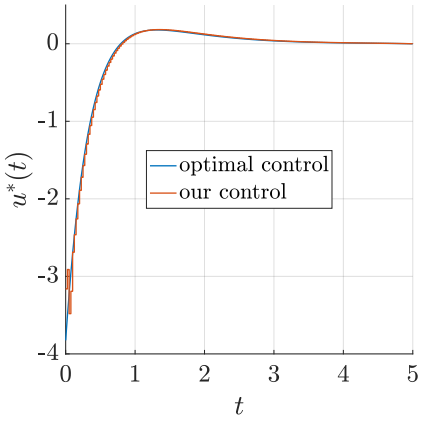
(b) trajectory $\Delta t = 0.1$



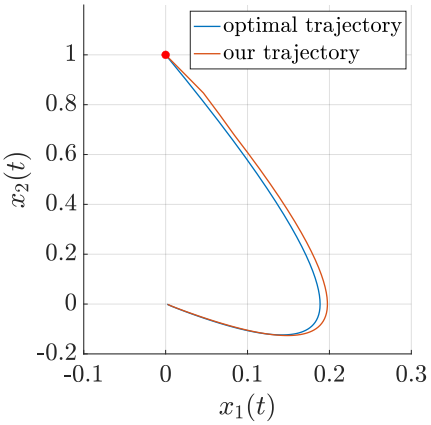
(c) control $\Delta t = 0.05$



(d) trajectory $\Delta t = 0.05$



(e) control $\Delta t = 0.025$



(f) trajectory $\Delta t = 0.025$

Figure 5.2. Test 1: Simulations for different values of Δt. The left column shows the control chosen by the algorithm as a function of time (in red) compared with the optimal control (in blue), computed knowing the matrix $\widehat{A}$; the right column shows the trajectories in $\mathbb{R}^2$. The red dot is the trajectory starting point.

accurate. This causes the trajectory to deviate from the optimal solution. However, after few steps the matrix $\widehat{A}$ is well approximated and the algorithm manages to steer the state towards the origin anyway. In Table 5.1 we have reported the cost of the solution found by the algorithm for different choices of Δt, compared with the optimal cost; the latter has been computed solving the Riccati equation relative to the true matrix $\widehat{A}$. When Δt is smaller, the algorithm recovers the matrix $\widehat{A}$ quickly and thus the deviation from the optimal solution is smaller. This confirms the heuristic argument presented in Section 5.2.5. We can also observe a numerical order of convergence almost equal to 1.

Table 5.1. Test 1: Cost of the trajectory for different values of Δt, when using a first-order scheme. The cost is compared with the optimal cost C^* computed knowing $\widehat{A}$.

Δt	Cost	Error	Order
0.1	0.3897	0.0072	-
0.05	0.3866	0.0041	0.81
0.025	0.3849	0.0023	0.82
0.0125	0.3838	0.0013	0.88
C^*	0.3825	-	-

We tried different finite difference schemes for the approximation of the state derivatives (see Section 5.2.4). Table 5.2 shows the error in the approximation of $\widehat{A}$ at the end of the simulation, when using schemes of order $p = 1, 2, 4$ and for different values of Δt. As expected, when we consider more accurate approximations of the gradient, we get better estimations of the matrix $\widehat{A}$. Unfortunately, the same does not hold for the solution costs, which are not significantly improved if compared with the ones found by the first-order approximation (see Table 5.3). Indeed, the cost obtained by the methods using the second-order and fourth-order schemes is higher than the cost obtained by the first-order scheme method. This is due to the fact that errors in the control are made mainly at the beginning, when the dynamics is still unknown and the distribution π is the one chosen arbitrarily as prior. Therefore, methods that use higher-order approximations are "disadvantaged" because they require to keep the first control constant for longer rounds, and this may take the system away from the optimal trajectory.

5.3.2 Test 2

For the second test we choose $n = 4$, $m = 3$ and $T = 10$. The true $\widehat{A}$ (unknown) is

$$\widehat{A} = \begin{pmatrix} -0.0215 & -0.7776 & -0.1922 & 0.9123 \\ -0.3246 & 0.5605 & -0.8071 & 0.1504 \\ 0.8001 & -0.2205 & -0.7360 & -0.8804 \\ -0.2615 & -0.5166 & 0.8841 & -0.5304 \end{pmatrix},$$

Table 5.2. Test 1: Error in the approximation of $\widehat{A}$ at the end of the simulation using different Δt and different finite difference schemes for the derivatives approximation; p is the scheme order.

	$p=1$		$p=2$		$p=4$	
Δt	Error	Order	Error	Order	Error	Order
0.1	0.167	-	9.74e-3	-	1.63e-4	-
0.05	0.087	0.94	1.91e-3	2.3	3.00e-6	5.8
0.025	0.045	0.95	4.14e-4	2.2	1.03e-7	4.8
0.0125	0.018	1.00	6.30e-5	2.1	1.70e-9	4.5

Table 5.3. Test 1: Cost of the trajectory for different values of Δt, when using a second-order and a fourth-order scheme. The cost is compared with the optimal cost C^* computed knowing $\widehat{A}$.

	$p=2$			$p=4$		
Δt	Cost	Error	Order	Cost	Error	Order
0.1	0.4003	0.0178	-	0.5884	0.2059	-
0.05	0.3892	0.0067	1.41	0.3995	0.0169	3.6
0.025	0.3864	0.0038	0.80	0.3888	0.0063	1.4
0.0125	0.3847	0.0021	0.86	0.3862	0.0037	0.78
C^*	0.3825	-	-	0.3825	-	-

while the matrix B (known) is

$$B = \begin{pmatrix} -0.2937 & -0.6620 & -0.0982 \\ 0.6424 & 0.2982 & 0.0940 \\ -0.9692 & 0.4634 & -0.4074 \\ -0.9140 & 0.2955 & 0.4894 \end{pmatrix}.$$

The cost matrices are respectively $Q = \frac{1}{4} I_4$, $R = \frac{1}{3} I_3$ and $Q_f = I_4$, and the starting point is $x_0 = ones(4, 1)$, i.e. the $\mathbb{R}^4$ vector with all ones. We set $\Delta t = 0.025$, $S = 4$ and use a second-order approximation for the derivatives. Fig. 5.3 shows the three components of the control found by the algorithm, together with the optimal control. In Fig. 5.4 we plot the components of the corresponding trajectory, which lives in $\mathbb{R}^4$. The components are drawn in pairs, in two graphs on the $\mathbb{R}^2$ plane. The behavior observed in Test 1 is even more visible here: in the first steps the control is not accurate since we do not know $\widehat{A}$, but after few steps the algorithm learns more on the matrix and manages to bring the state to the origin. The cost of the control found by the algorithm is 1.111, whereas the optimal cost computed using $\widehat{A}$ is 1.056.

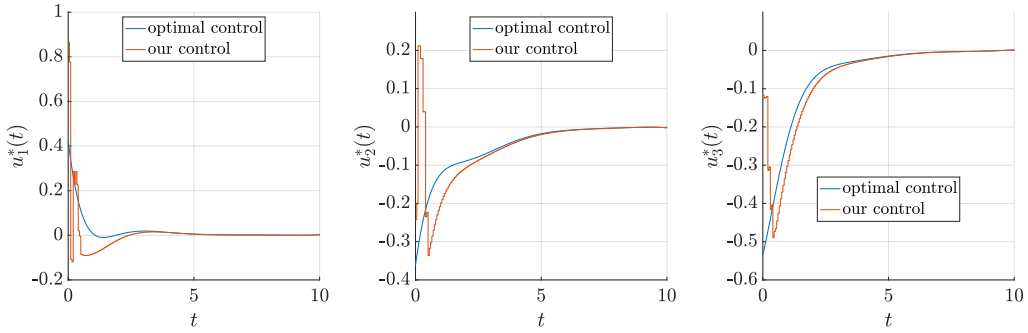


Figure 5.3. Test 2: The three components of the control (in red) compared with the components of the optimal control (in blue).

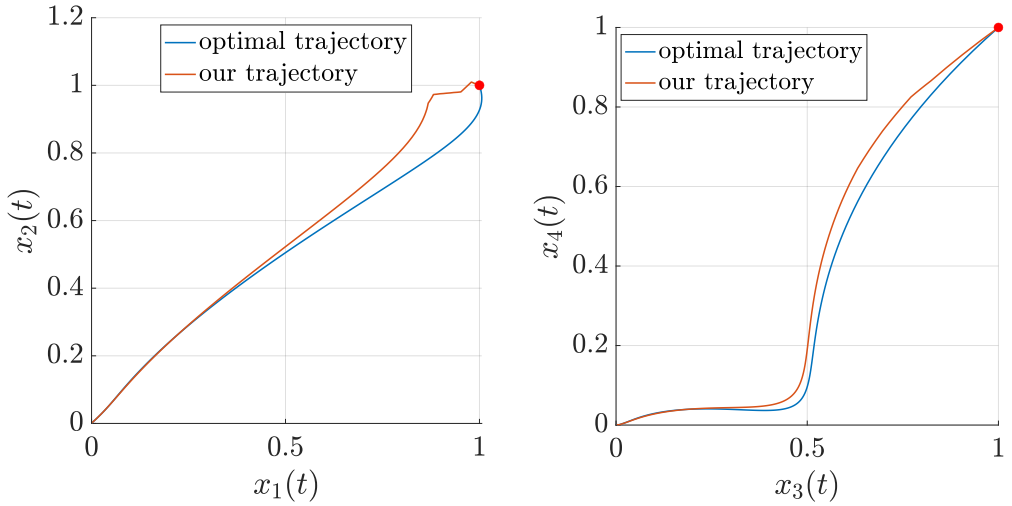


Figure 5.4. Test 2: The state trajectory in $\mathbb{R}^4$, represented by projecting components in pairs: (x_1, x_2) on the left and (x_3, x_4) on the right. The red dots indicate the starting point of the trajectory.

5.3.3 Test 3

In the third test we considered a system with state dimension $n = 10$ and control dimension $m = 5$. The matrices A and B were generated randomly with elements between -1 and 1; due to the large dimension, we choose not to write them down. We set $\Delta t = 0.01$, $S = 2$ and we used a first-order scheme for the derivative approximation. The initial point was $ones(10, 1)$. The cost obtained by the algorithm was about 3.32, while the optimal cost was about 2.52. In Figure 5.5 we reported the 5 components of the control, together with the components of the optimal control. In Figure 5.6 we have drawn the 10 components of the trajectory grouped in pairs. The qualitative behavior of the algorithm is similar to what happened in previous tests. In fact, at the beginning the control is far from the optimal one and the trajectory moves away from the optimal one. However, after a few rounds the estimate of the matrix $\widehat{A}$ becomes more precise and eventually the algorithm manages to bring the state towards the origin.

5.3.4 Test 4

To conclude, we present here a simulation in which the matrix $\widehat{A}$ is simple, but our algorithm still can't reconstruct it. The reason is that the data given by the trajectory, i.e. the matrix X of the state observations, don't contain enough information. However, we will observe that despite this the algorithm manages to control the system well.

The state dimension is $n = 10$ and the control $u(t)$ is 1-dimensional. The matrices that define the system dynamics are $\widehat{A} = -I_{10}$ and $B = ones(10, 1)$, the matrix in $\mathbb{R}^{10\times 1}$ with all ones. The cost matrices are $Q = \frac{1}{10} I_{10}$, $R = 1$ and $Q_f = \mathbf{0}_{10}$. Starting from $x_0 = ones(10, 1)$, each component k of the state is independent of the others and follows the same dynamics

$$\begin{cases} \dot{x}_{(k)}(t) = -x_{(k)}(t) + u(t) & t \in [0, 2] \\ x_{(k)}(0) = -1. \end{cases} \tag{5.16}$$

Notice that the notation we use here is different from the one used in Section 5.2.3, where the symbol x_i^j denoted the whole vector state at time t_i^j, i.e. $x_i^j = x(t_i^j) \in \mathbb{R}^n$. Here instead we denote by $x_{(k)}(t) \in \mathbb{R}$ the single k-th component of the state vector $x(t)$ at time $t \in [0, 2]$. Since the system dynamics (5.16) and the cost matrices completely symmetric in the components, the problem is equivalent to a 1-dimensional optimal control problem:

$$\begin{cases} \text{minimize } \int_0^2 \left[\frac{1}{10} \left(x_{(1)}(t) \right)^2 + u(t)^2 \right] dt \\ \text{over } (x, u)(\cdot) \text{ such that } u : [0, 2] \to \mathbb{R} \text{ measurable and} \\ \dot{x}_{(1)}(t) = -x_{(1)}(t) + u(t), \quad t \in [0, 2], \\ x_{(1)}(0) = -1. \end{cases} \tag{5.17}$$

However, our algorithm doesn't know this, so it will try to solve a 10-dimensional control problem. By using a time-step of $\Delta t = 0.025$ and a first-order scheme in the

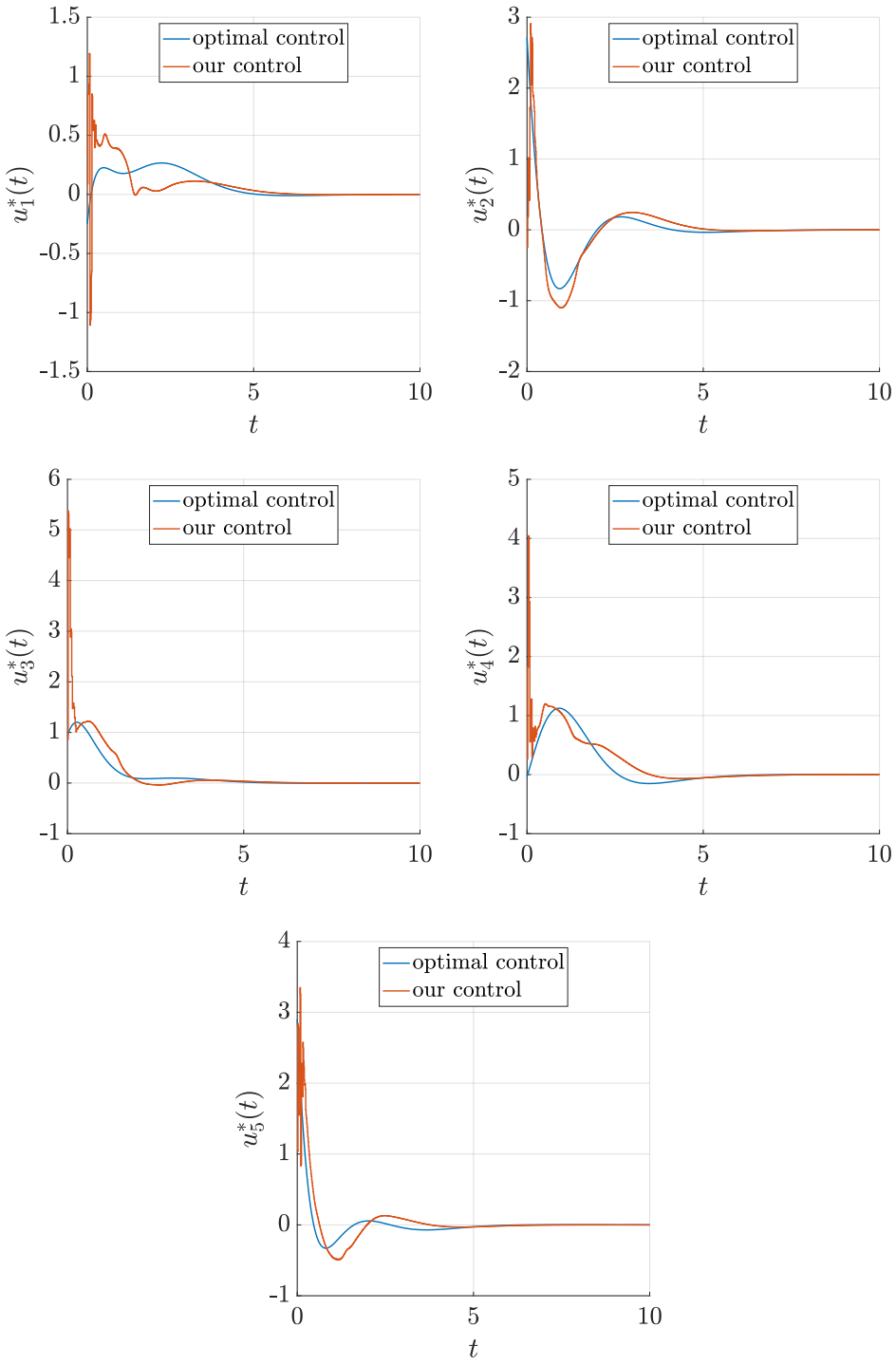


Figure 5.5. Test 3: The five components of the control (in red) compared with the components of the optimal control (in blue).

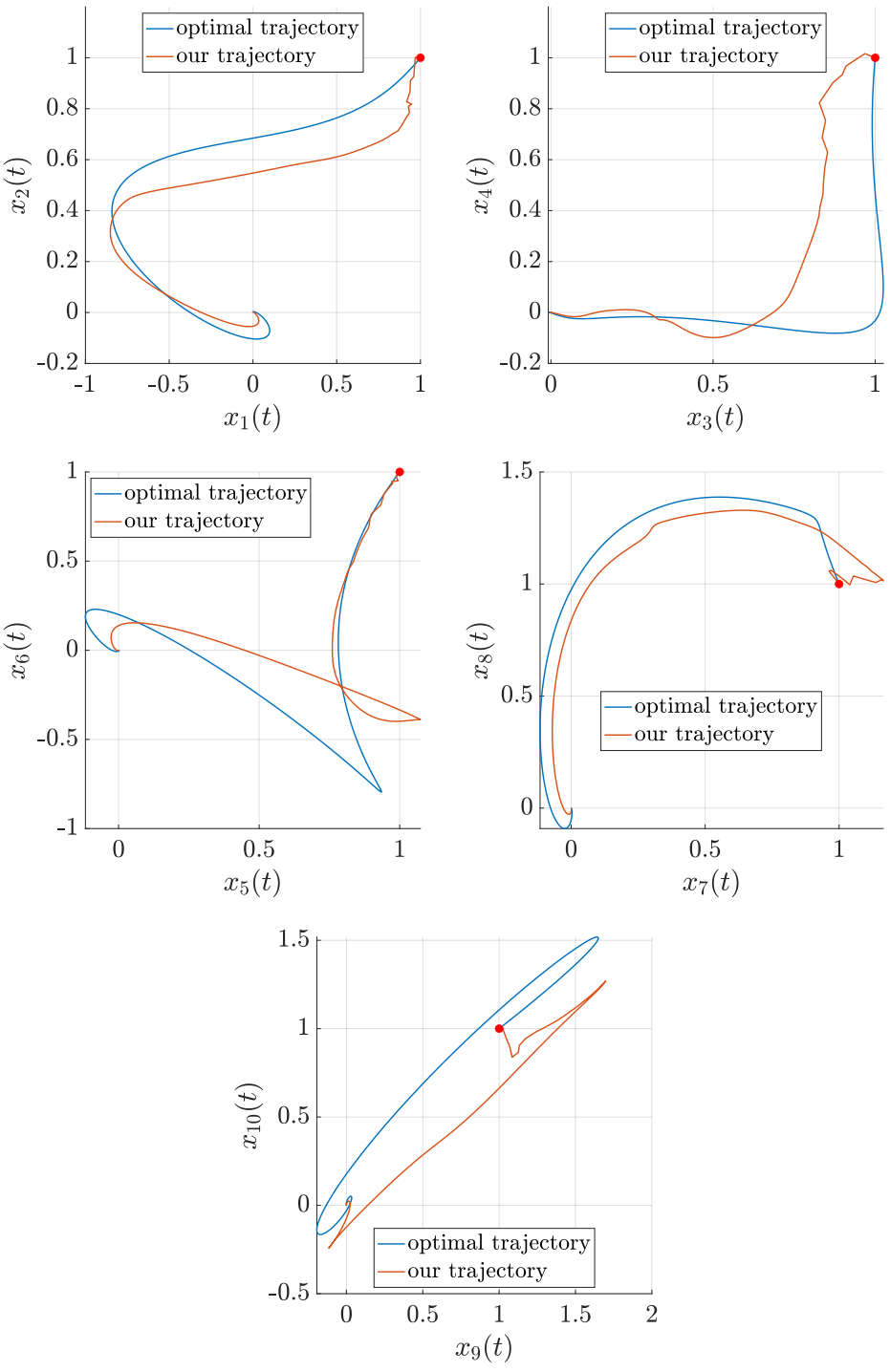


Figure 5.6. Test 3: The state trajectory in $\mathbb{R}^{10}$, represented by projecting components in pairs onto the $\mathbb{R}^2$ plane. The red dots indicate the starting point of the trajectory.

derivative approximation, the matrix it reconstructs is a full matrix (whereas $\widehat{A}$ was diagonal) with all elements equal to -0.0982:

$$\widetilde{A} = \begin{pmatrix} -0.0982 & -0.0982 & \cdots & -0.0982 \\ -0.0982 & -0.0982 & \cdots & -0.0982 \\ \vdots & \vdots & \ddots & \vdots \\ -0.0982 & -0.0982 & \cdots & -0.0982 \end{pmatrix} \in \mathbb{R}^{10\times 10}. \tag{5.18}$$

The reason is that the data gathered during the simulation don't contain enough information to reconstruct the corrected matrix. In particular, as described in Section 5.2.3, to update our model of matrix $\widehat{A}$ we use the state observations from the trajectory, gathered in a matrix X whose columns are the state vectors at different time instants. Since in this example the trajectory is the same along each component, the matrix X will have constant columns and its rank will be 1. In Section 2.3.2 we observed that the mean of the distribution provided by BLR is somewhat close to the LS solution and converges to it as the noise vanishes. We can expect that since the LS solution (2.74) doesn't work well when the rank of X is not full, BLR will have similar problems. In this simulation, the rank of X was 1 and this is why the reconstruction of $\widetilde{A}$ was not good. Despite this, we can observe in Figure 5.7 how the algorithm still manages to reconstruct the optimal control, except for the first instants of time in which it has no knowledge of the dynamics, as already observed in the previous tests. In Figure 5.8, we have drawn the trajectory reconstructed by the algorithm along the first component (along the other components it is identical, given the symmetry of the problem). We note that, although the matrix $\widehat{A}$ was not reconstructed correctly, the algorithm managed to control the system well, bringing the state towards the origin. Finally, notice that because of the problem symmetries the reconstructed matrix $\widetilde{A}$ in (5.18) has the same effect on the state as $\widetilde{A}' = 10 * (-0.0982)\, I_{10} = -0.982\, I_{10}$.

As a general comment, the BLR formulas may not work for systems with symmetries and for which the trajectory matrix X does not admit maximum rank. As a result, the algorithm may not correctly reconstruct matrix $\widehat{A}$. But even in these cases the algorithm may be able to control the dynamics well.

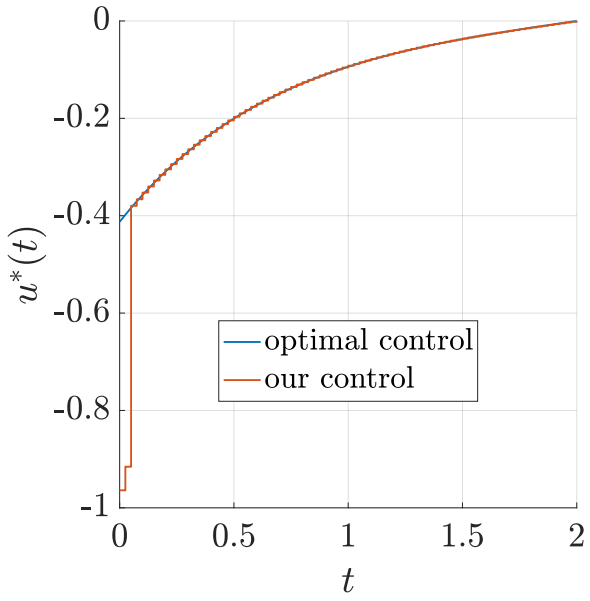


Figure 5.7. Test 4: The 1-dimensional control reconstructed by our algorithm compared with the optimal control.

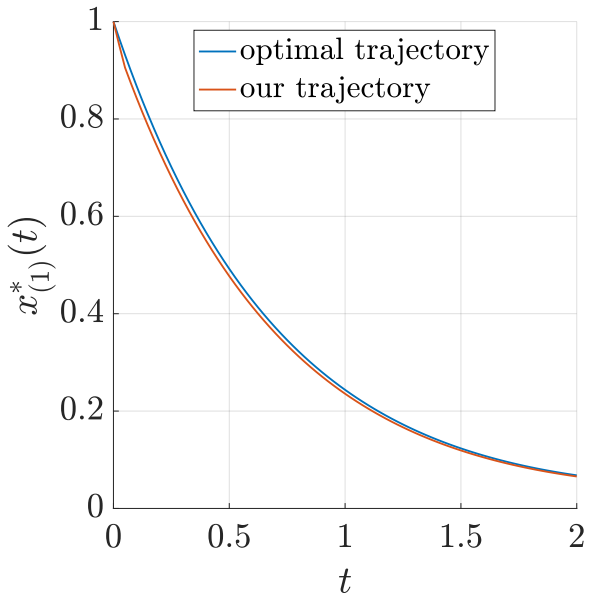


Figure 5.8. Test 4: The trajectory obtained by our algorithm projected along the first component, compared with the optimal trajectory. The starting point is $x_{(1)}(0) = 1$.

Conclusion and perspectives

This thesis has investigated the connection between optimal control and reinforcement learning. These two research areas deal with similar problems but address them differently. We believe that each can benefit from the other. For example, one can draw on the long-standing theory of optimal control to prove convergence results of RL algorithms, which are still missing in most cases. On the other hand, some RL tools are achieving excellent results and could be integrated with classical control methods, as it has already been done with Model Predictive Control (see e.g. [73, 32]). In this thesis, we have worked in both directions, and we can therefore divide our contributions into two parts.

Average cost optimal control problems

In Chapter 3, we have introduced and studied an optimal control problem in an uncertain environment, following some recent works [98, 99]. In our framework, the degree of uncertainty of the control system is captured by a probability measure defined on a compact space of functions. The cost functional is defined as an average cost along all possible trajectories. In Section 3.2 we showed how this framework is closely related to a class of probabilistic MBRL algorithms that aim at designing a probabilistic model of the dynamics rather than providing a pointwise estimate. Having established this connection is in itself a relevant result, as it is a starting point to better understand these RL algorithms. Then, we proved a convergence result for the value function of the average cost problem in a generic, nonlinear setting. Namely, that the value function V_π of the average cost problem converges towards the value function of the classical, underlying optimal control problem, when the probability distribution π converges (w.r.t. the Wasserstein distance (3.6)) to the Dirac delta $\delta_{\widehat{f}}$ of the true dynamics. This result sheds new light on the convergence of MBRL algorithms.

In Chapter 4 we analyzed the same framework in the context of linear-quadratic control systems and proved stronger results. Under standard hypotheses on the system dynamics and the cost functional, we proved that the open-loop, optimal control $\bar{u}_\pi$ of the average cost problem converges to the open-loop, optimal control $\bar{u}$ of the actual system as soon as the distribution π is sufficiently close to a Dirac delta $\delta_{\widehat{A}}$ evaluated at the actual system matrix $\widehat{A}$. We also showed that, when the probability distribution π is a discrete probability measure, then also the closed-loop optimal control of the average cost problem converges to the closed-loop optimal control of the actual system. This further confirms that, when applied to linear systems with quadratic costs, that class of MBRL algorithms converges to the correct

solution, provided that the model they have reconstructed closely approximates the dynamics of the system. Furthermore, even if the probabilistic model does not converge to the Dirac delta evaluated at the true dynamics, these results help us understand what the algorithm is doing and how it can be improved.

Future directions As a future direction, we would like to prove similar convergence results for the optimal control in wider classes of nonlinear systems, e.g. control affine optimal control problem with convex functional, getting closer to the problem formulation studied in [37]. Some numerical examples in Section 3.4 seem to be a good omen for future investigations in this direction. We believe this is a good track to finally be able to prove the convergence of some MBRL algorithms. Another step to take concerns the convergence of the probabilistic model to the true dynamics of the system. This property depends on how the probabilistic model is constructed in the *model learning* step (cf. Section 2.4.3) and can vary according to the algorithm. A greater understanding of the properties a model must have can help us build new and efficient algorithms that are guaranteed to converge to the exact solution.

Numerical methods for LQR problems with unknown dynamics

We also have proposed a new algorithm to deal with LQR problems when the state matrix $\widehat{A}$ is unknown. Our algorithm is designed to approximate the matrix $\widehat{A}$ *and* to find a suitable control that brings the system towards the origin in a single simulation. It takes contributions from both optimal control and RL and is an example of how this integration can produce innovative algorithms. In particular, it uses a Bayesian linear regression to reconstruct the initially unknown state matrix. The feedback control is computed by solving the Riccati differential equation corresponding to the mean matrix of the probabilistic model. Numerical tests presented in Section 5.3 have demonstrated how it achieves excellent results, managing to control the system *online*, despite not knowing $\widehat{A}$ at the beginning of the simulation. We also observed that the state matrix is not well approximated when the rank of the trajectory matrix X does not have full rank, but the algorithm still manages to control the system. Our algorithm can be integrated with higher order approximation schemes, and we have seen how this improves the approximation of $\widehat{A}$.

Future directions Certainly, several directions of research remain open. In the near future we would like to work on the convergence of the method, so far only sketched in Section 5.2.5, and on a possible error estimate, which could concern not only our algorithm but more in general all Bayesian linear regression methods. It would be interesting to investigate whether the Bayesian update of the distribution described in Section 5.2.3 could be made more efficient by using Kalman filtering [72] and related theory. Then we would like to extend the algorithm to other settings, also considering the control matrix B or the cost matrices Q and R as unknown. Another interesting direction is the extension of our approach to nonlinear problems. These are much more complex to solve online, since a generic Hamilton-Jacobi-Bellman equation requires a considerable computational cost to solve, especially in high dimension. Finally, a great application of the algorithm would be the identification and control of linear PDEs, which are transformed into LQR problems through a time discretization.

Bibliography

[1] P. Abbeel, M. Quigley, and A. Y. Ng, *Using inaccurate models in reinforcement learning*, in Proceedings of the 23rd international conference on Machine learning, 2006, pp. 1–8.

[2] J. Ackermann, *Robust control: the parameter space approach*, Springer Science & Business Media, 2002.

[3] A. Alla, M. Falcone, and D. Kalise, *An efficient policy iteration algorithm for dynamic programming equations*, SIAM Journal on Scientific Computing, 37 (2015), pp. A181–A200.

[4] B. D. Anderson and J. B. Moore, *Optimal control: linear quadratic methods*, Courier Corporation, 2007.

[5] U. M. Ascher, R. M. Mattheij, and R. D. Russell, *Numerical solution of boundary value problems for ordinary differential equations*, SIAM, 1995.

[6] K. J. Åström and P. Eykhoff, *System identification—a survey*, Automatica, 7 (1971), pp. 123–162.

[7] K. J. Åström and B. Wittenmark, *Adaptive control*, Courier Corporation, 2013.

[8] C. G. Atkeson and J. C. Santamaria, *A comparison of direct and model-based reinforcement learning*, in Proceedings of International Conference on Robotics and Automation, vol. 4, 1997, pp. 3557–3564 vol.4.

[9] Y. Bar-Shalom and E. Tse, *Dual effect, certainty equivalence, and separation in stochastic control*, IEEE Transactions on Automatic Control, 19 (1974), pp. 494–500.

[10] D. Barber and C. M. Bishop, *Ensemble learning in bayesian neural networks*, Nato ASI Series F Computer and Systems Sciences, 168 (1998), pp. 215–238.

[11] M. Bardi and I. Capuzzo Dolcetta, *Optimal Control and Viscosity Solutions of Hamilton-Jacobi-Bellman Equations*, Birkhäuser, 1997.

[12] A. G. Barto, R. S. Sutton, and C. Watkins, *Sequential decision problems and neural networks*, Advances in neural information processing systems, 2 (1989), pp. 686–693.

[13] T. Başar and P. Bernhard, *H_∞ optimal control and related minimax design problems: a dynamic game approach*, Springer, 2008.

[14] A. Becker, P. Kumar, and C.-Z. Wei, *Adaptive control with the stochastic approximation algorithm: Geometry and convergence*, IEEE Transactions on Automatic Control, 30 (1985), pp. 330–338.

[15] R. E. Bellman, *The theory of dynamic programming*, Bulletin of the American Mathematical Society, 60 (1954), pp. 503–515.

[16] R. E. Bellman, *Dynamic Programming*, Courier Dover Publications, 1957.

[17] R. E. Bellman, *A markovian decision process*, Journal of mathematics and mechanics, 6 (1957), pp. 679–684.

[18] D. P. Bertsekas, *Approximate dynamic programming*, Citeseer, 2008.

[19] D. P. Bertsekas, *Value and policy iterations in optimal control and adaptive dynamic programming*, IEEE transactions on neural networks and learning systems, 28 (2015), pp. 500–509.

[20] D. P. Bertsekas, *Reinforcement and Optimal Control*, Athena Scientific, 2019.

[21] D. P. Bertsekas et al., *Dynamic programming and optimal control: Vol. 1*, Athena scientific Belmont, 2000.

[22] D. P. Bertsekas and J. N. Tsitsiklis, *Neuro-dynamic programming*, Athena Scientific, 1996.

[23] P. Bettiol and N. Khalil, *Necessary optimality conditions for average cost minimization problems*, Discrete & Continuous Dynamical Systems-B, 24 (2019), p. 2093.

[24] C. M. Bishop, *Pattern Recognition and Machine Learning*, Information Science and Statistics, Springer, New York, 2006.

[25] V. G. Boltyansky and A. G. Poznyak, *The Robust Maximum Principle*, Boston Birkhäuser, 2012.

[26] G. E. Box and G. C. Tiao, *Bayesian inference in statistical analysis*, vol. 40, John Wiley & Sons, 2011.

[27] L. Breiman, J. H. Friedman, R. A. Olshen, and C. J. Stone, *Classification and regression trees*, Routledge, 2017.

[28] M. D. Buhmann, *Radial basis functions*, Acta numerica, 9 (2000), pp. 1–38.

[29] E. F. Camacho and C. B. Alba, *Model predictive control*, Springer science & business media, 2013.

[30] I. Capuzzo Dolcetta and M. Falcone, *Discrete dynamic programming and viscosity solutions of the Bellman equation*, in Annales de l'Institut Henri Poincaré C, Analyse non linéaire, vol. 6, Elsevier, 1989, pp. 161–183.

[31] G. Chowdhary, H. A. Kingravi, J. P. How, and P. A. Vela, *A Bayesian nonparametric approach to adaptive control using Gaussian processes*, in CDC, IEEE, 2013, pp. 874–879.

[32] K. Chua, R. Calandra, R. McAllister, and S. Levine, *Deep Reinforcement Learning in a Handful of Trials using Probabilistic Dynamics Models*, in Advances in Neural Information Processing Systems, vol. 2018-December, 2018, pp. 4754–4765.

[33] W. A. Coppel, *18.—Linear-quadratic optimal control*, Proceedings of the Royal Society of Edinburgh Section A: Mathematics, 73 (1975), pp. 271–289.

[34] M. G. Crandall and P.-L. Lions, *Viscosity solutions of hamilton-jacobi equations*, Transactions of the American mathematical society, 277 (1983), pp. 1–42.

[35] B. De Finetti, *Theory of probability: A critical introductory treatment*, vol. 6, John Wiley & Sons, 2017.

[36] M. P. Deisenroth, *Efficient reinforcement learning using Gaussian processes*, vol. 9, KIT Scientific Publishing, 2010.

[37] M. P. Deisenroth, D. Fox, and C. E. Rasmussen, *Gaussian processes for data-efficient learning in robotics and control*, IEEE transactions on pattern analysis and machine intelligence, 37 (2013), pp. 408–423.

[38] M. P. Deisenroth and C. E. Rasmussen, *PILCO: A model-based and data-efficient approach to policy search*, in Proceedings ICML-11, 2011, pp. 465–472.

[39] A. Der Kiureghian and O. Ditlevsen, *Aleatory or epistemic? Does it matter?*, Structural safety, 31 (2009), pp. 105–112.

[40] K. Doya, *Reinforcement learning in continuous time and space*, Neural Computation, 12 (2000), pp. 219–245.

[41] J. Doyle, K. Glover, P. Khargonekar, and B. Francis, *State-space solutions to standard H_2 and H_∞ control problems*, in 1988 American Control Conference, IEEE, 1988, pp. 1691–1696.

[42] N. R. Draper and H. Smith, *Applied regression analysis*, vol. 326, John Wiley & Sons, 1998.

[43] H. Drucker, C. J. Burges, L. Kaufman, A. Smola, V. Vapnik, et al., *Support vector regression machines*, Advances in neural information processing systems, 9 (1997), pp. 155–161.

[44] M. Falcone, *A numerical approach to the infinite horizon problem of deterministic control theory*, Applied Mathematics and Optimization, 15 (1987), pp. 1–13.

[45] M. Falcone, *Numerical solution of dynamic programming equations*, in Optimal Control and Viscosity Solutions of Hamilton-Jacobi-Bellman Equations, Birkhäuser, 1997. Appendix A of [11].

[46] M. Falcone and R. Ferretti, *Semi-Lagrangian approximation schemes for linear and Hamilton—Jacobi equations*, SIAM, 2013.

[47] J. A. Farrell and M. M. Polycarpou, *Adaptive approximation based control: unifying neural, fuzzy and traditional adaptive approximation approaches*, vol. 48, John Wiley & Sons, 2006.

[48] A. Feldbaum, *Dual control theory. I-IV*, Avtomatika i Telemekhanika, 21 (1960-1961), pp. 1240–1249.

[49] N. M. Filatov and H. Unbehauen, *Adaptive dual control: Theory and applications*, vol. 302, Springer Science & Business Media, 2004.

[50] W. H. Fleming and R. W. Rishel, *Deterministic and stochastic optimal control*, vol. 1, Springer Science & Business Media, 2012.

[51] W. H. Fleming and H. M. Soner, *Controlled Markov processes and viscosity solutions*, vol. 25, Springer Science & Business Media, 2006.

[52] J. Fong, Y. Tan, V. Crocher, D. Oetomo, and I. Mareels, *Dual-loop iterative optimal control for the finite horizon LQR problem with unknown dynamics*, Systems & Control Letters, 111 (2018), pp. 49–57.

[53] A. Friedman, *Differential games*, Courier Corporation, 2013.

[54] J. A. Frueh and M. Q. Phan, *Linear quadratic optimal learning control (LQL)*, International Journal of Control, 73 (2000), pp. 832–839.

[55] Y. Gal, *Uncertainty in deep learning*, PhD thesis, University of Cambridge, 2016.

[56] Y. Gal, R. McAllister, and C. E. Rasmussen, *Improving PILCO with Bayesian neural network dynamics models*, in Data-Efficient Machine Learning workshop, ICML, vol. 4, 2016, p. 25.

[57] A. Gelman, J. B. Carlin, H. S. Stern, and D. B. Rubin, *Bayesian data analysis*, Chapman and Hall/CRC, 1995.

[58] M. Ghavamzadeh, S. Mannor, J. Pineau, A. Tamar, et al., *Bayesian Reinforcement Learning: A Survey*, Foundations and Trends® in Machine Learning, 8 (2015), pp. 359–483.

[59] I. Goodfellow, Y. Bengio, and A. Courville, *Deep learning*, MIT press, 2016.

[60] G. C. Goodwin, P. J. Ramadge, and P. E. Caines, *Discrete time stochastic adaptive control*, SIAM Journal on Control and Optimization, 19 (1981), pp. 829–853.

[61] A. K. Gupta, K. G. Smith, and C. E. Shalley, *The interplay between exploration and exploitation*, Academy of management journal, 49 (2006), pp. 693–706.

[62] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine, *Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor*, in International Conference on Machine Learning, PMLR, 2018, pp. 1861–1870.

[63] J. Hall, C. E. Rasmussen, and J. Maciejowski, *Reinforcement learning with reference tracking control in continuous state spaces*, in 2011 50th IEEE Conference on Decision and Control and European Control Conference, IEEE, 2011, pp. 6019–6024.

[64] L. K. Hansen and P. Salamon, *Neural network ensembles*, IEEE transactions on pattern analysis and machine intelligence, 12 (1990), pp. 993–1001.

[65] J. Harrison, *Lecture Notes on Optimal and Learning-Based Control*. `https://github.com/StanfordASL/AA203-Notes`, May 2020.

[66] A. E. Hoerl and R. W. Kennard, *Ridge regression: applications to nonorthogonal problems*, Technometrics, 12 (1970), pp. 69–82.

[67] I. M. Horowitz, *Synthesis of feedback systems*, Academic Press, 1963.

[68] R. A. Howard, *Dynamic programming and Markov processes.*, John Wiley, 1960.

[69] R. Isaacs, *Differential games: a mathematical theory with applications to warfare and pursuit, control and optimization*, Courier Corporation, 1999.

[70] H. M. James, N. B. Nichols, and R. S. Phillips, *Theory of servomechanisms*, vol. 25, McGraw-Hill New York, 1947.

[71] M. Janner, J. Fu, M. Zhang, and S. Levine, *When to trust your model: Model-based policy optimization*, in Advances in Neural Information Processing Systems, vol. 32, 2019.

[72] R. E. Kalman, *A New Approach to Linear Filtering and Prediction Problems*, Journal of Basic Engineering, 82 (1960), pp. 35–45.

[73] S. Kamthe and M. P. Deisenroth, *Data-efficient reinforcement learning with probabilistic model predictive control*, in International Conference on Artificial Intelligence and Statistics, PMLR, 2018, pp. 1701–1710.

[74] K. J. Keesman, *System identification: an introduction*, Springer Science & Business Media, 2011.

[75] W. G. Kelley and A. C. Peterson, *Difference equations: an introduction with applications*, Academic press, 2001.

[76] E. D. Klenske and P. Hennig, *Dual control for approximate bayesian reinforcement learning*, The Journal of Machine Learning Research, 17 (2016), pp. 4354–4383.

[77] P. Klink, *Model-based reinforcement learning from pilco to pets*, in Reinforcement Learning Algorithms: Analysis and Applications, Springer, 2021, pp. 165–175.

[78] G. Konidaris, S. Osentoski, and P. Thomas, *Value function approximation in reinforcement learning using the fourier basis*, in Twenty-fifth AAAI conference on artificial intelligence, 2011.

[79] S. N. Kumpati, P. Kannan, et al., *Identification and control of dynamical systems using neural networks*, IEEE Transactions on neural networks, 1 (1990), pp. 4–27.

[80] H. J. Kushner and P. G. Dupuis, *Numerical methods for stochastic control problems in continuous time*, vol. 24, Springer Science & Business Media, 2001.

[81] H. Kwakernaak and R. Sivan, *Linear optimal control systems*, vol. 1, Wiley-interscience New York, 1972.

[82] I. D. Landau, R. Lozano, M. M'Saad, and A. Karimi, *Adaptive control: algorithms, analysis and applications*, Springer Science & Business Media, 2011.

[83] C. L. Lawson and R. J. Hanson, *Solving least squares problems*, SIAM, 1995.

[84] Y. LeCun, Y. Bengio, and G. Hinton, *Deep learning*, nature, 521 (2015), pp. 436–444.

[85] J. Y. Lee, J. B. Park, and Y. H. Choi, *Integral reinforcement learning for continuous-time input-affine nonlinear systems with simultaneous invariant explorations*, IEEE Transactions on Neural Networks and Learning Systems, 26 (2014), pp. 916–932.

[86] A. M. Legendre, *Nouvelles méthodes pour la détermination des orbites des comètes; par AM Legendre...*, chez Firmin Didot, libraire pour lew mathematiques, la marine, l . . . , 1806.

[87] F. L. Lewis and D. Vrabie, *Reinforcement learning and adaptive dynamic programming for feedback control*, IEEE circuits and systems magazine, 9 (2009).

[88] N. Li, I. Kolmanovsky, and A. Girard, *LQ control of unknown discrete-time linear systems—A novel approach and a comparison study*, Optimal Control Applications and Methods, 40 (2019), pp. 265–291.

[89] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra, *Continuous control with deep reinforcement learning*, in 4th International Conference on Learning Representations (ICLR), 2016.

[90] L. Ljung, *System identification*, in Signal analysis and prediction, Springer, 1998, pp. 163–173.

[91] L. Ljung, C. Andersson, K. Tiels, and T. B. Schön, *Deep learning and system identification*, IFAC-PapersOnLine, 53 (2020), pp. 1175–1181.

[92] J. Lohéac and E. Zuazua, *From averaged to simultaneous controllability*, in Annales de la Faculté des sciences de Toulouse: Mathématiques, vol. 25, 2016, pp. 785–828.

[93] D. J. C. Mackay, *Bayesian methods for adaptive models*, PhD thesis, California Institute of Technology, 1992.

[94] H. Mania, S. Tu, and B. Recht, *Certainty equivalence is efficient for linear quadratic control*, Advances in Neural Information Processing Systems, 32 (2019), pp. 10154–10164.

[95] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, et al., *Human-level control through deep reinforcement learning*, Nature, 518 (2015), pp. 529–533.

[96] T. M. Moerland, J. Broekens, and C. M. Jonker, *Model-based reinforcement learning: A survey*, arXiv preprint arXiv:2006.16712, (2020).

[97] R. Munos, *A study of reinforcement learning in the continuous case by the means of viscosity solutions*, Machine Learning, 40 (2000), pp. 265–299.

[98] R. Murray and M. Palladino, *A model for system uncertainty in reinforcement learning*, Systems & Control Letters, 122 (2018), pp. 24–31.

[99] R. Murray and M. Palladino, *Modelling uncertainty in reinforcement learning*, in 2019 IEEE 58th Conference on Decision and Control (CDC), IEEE, 2019, pp. 2436–2441.

[100] R. M. Neal, *Bayesian learning for neural networks*, vol. 118, Springer Science & Business Media, 2012.

[101] J. Nocedal and S. Wright, *Numerical optimization*, Springer Science & Business Media, 2006.

[102] A. O'Hagan and J. J. Forster, *Kendall's advanced theory of statistics, volume 2B: Bayesian inference*, vol. 2, Arnold, 2004.

[103] M. J. Orr et al., *Introduction to radial basis function networks*, 1996.

[104] A. Pacifico, A. Pesare, and M. Falcone, *A new algorithm for the LQR problem with partially unknown dynamics*, in 2021 Large-Scale Scientific Computing (LSSC), Springer International Publishing, to appear.

[105] M. Palladino, *Necessary conditions for adverse control problems expressed by relaxed derivatives*, Set-Valued Var. Anal, 24 (2016).

[106] B. Pang, T. Bian, and Z.-P. Jiang, *Adaptive dynamic programming for finite-horizon optimal control of linear time-varying discrete-time systems*, Control Theory and Technology, 17 (2019), pp. 73–84.

[107] A. Perez, R. Platt, G. Konidaris, L. Kaelbling, and T. Lozano-Perez, *Lqr-rrt*: Optimal sampling-based motion planning with automatically derived extension heuristics*, in 2012 IEEE International Conference on Robotics and Automation, IEEE, 2012, pp. 2537–2542.

[108] A. Pesare, M. Palladino, and M. Falcone, *Convergence results for an averaged LQR problem with applications to Reinforcement Learning*, Mathematics of Control, Signals, and Systems, 33 (2021), p. 379–411.

[109] A. Pesare, M. Palladino, and M. Falcone, *Convergence of the value function in optimal control problems with unknown dynamics*, in 2021 European Control Conference (ECC), IEEE, to appear.

[110] H. J. Pesch and M. Plail, *The cold war and the maximum principle of optimal control*, Optimization Stories. Documenta Mathematica, (2012).

[111] V. Peterka, *Bayesian System Identification*, Automatica, 17 (1981), pp. 41–53.

[112] L. S. Pontryagin, *Mathematical theory of optimal processes*, CRC press, 1987.

[113] W. B. Powell, *Approximate Dynamic Programming: Solving the curses of dimensionality*, vol. 703, John Wiley & Sons, 2007.

[114] W. B. Powell, *A unified framework for stochastic optimization*, European Journal of Operational Research, 275 (2019), pp. 795–821.

[115] W. B. Powell, *From reinforcement learning to optimal control: A unified framework for sequential decisions*, in Handbook of Reinforcement Learning and Control, Springer, 2021, pp. 29–74.

[116] L. B. Prasad, B. Tyagi, and H. O. Gupta, *Optimal control of nonlinear inverted pendulum system using pid controller and lqr: performance analysis without and with disturbance input*, International Journal of Automation and Computing, 11 (2014), pp. 661–670.

[117] M. L. Puterman, *Markov decision processes: discrete stochastic dynamic programming*, John Wiley & Sons, 2014.

[118] A. V. Rao, *A survey of numerical methods for optimal control*, Advances in the Astronautical Sciences, 135 (2009), pp. 497–528.

[119] C. Rasmussen and C. Williams, *Gaussian Processes for Machine Learning*, Adaptive Computation and Machine Learning, MIT Press, Cambridge, MA, USA, Jan. 2006.

[120] B. Recht, *A tour of reinforcement learning: The view from continuous control*, Annual Review of Control, Robotics, and Autonomous Systems, 2 (2019), pp. 253–279.

[121] W. Reid, *Ordinary Differential Equations*, Applied Mathematics Series, Wiley, 1971.

[122] I. M. Ross, R. J. Proulx, M. Karpenko, and Q. Gong, *Riemann–Stieltjes optimal control problems for uncertain dynamic systems*, Journal of Guidance, Control, and Dynamics, 38 (2015), pp. 1251–1263.

[123] P. E. Rossi, G. M. Allenby, and R. McCulloch, *Bayesian statistics and marketing*, John Wiley & Sons, 2012.

[124] W. Rudin et al., *Principles of mathematical analysis*, vol. 3, McGraw-hill New York, 1964.

[125] G. A. Rummery and M. Niranjan, *On-line Q-learning using connectionist systems*, vol. 37, Citeseer, 1994.

[126] J. Schulman, S. Levine, P. Abbeel, M. Jordan, and P. Moritz, *Trust region policy optimization*, in International conference on machine learning, PMLR, 2015, pp. 1889–1897.

[127] C. E. Shannon, *Xxii. programming a computer for playing chess*, The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science, 41 (1950), pp. 256–275.

[128] D. Silver et al., *Mastering the game of go with deep neural networks and tree search*, Nature, 529 (2016), pp. 484–489.

[129] H. A. Simon, *Dynamic programming under uncertainty with a quadratic criterion function*, Econometrica, Journal of the Econometric Society, (1956), pp. 74–81.

[130] P. Soravia, H_∞ *control of nonlinear systems: Differential games and viscosity solutions*, SIAM Journal on control and optimization, 34 (1996), pp. 1071–1097.

[131] J. Sternby, *A simple dual control problem with an analytical solution*, IEEE Transactions on Automatic Control, 21 (1976), pp. 840–844.

[132] S. M. Stigler, *Gauss and the invention of least squares*, the Annals of Statistics, (1981), pp. 465–474.

[133] J. C. Strikwerda, *Finite difference schemes and partial differential equations*, SIAM, 2004.

[134] R. S. Sutton, *Learning to predict by the methods of temporal differences*, Machine learning, 3 (1988), pp. 9–44.

[135] R. S. Sutton, *Generalization in reinforcement learning: Successful examples using sparse coarse coding*, Advances in neural information processing systems, (1996), pp. 1038–1044.

[136] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*, MIT Press, Cambridge, MA, second ed., 2018.

[137] R. S. Sutton, A. G. Barto, and R. J. Williams, *Reinforcement Learning is Direct Adaptive Optimal Control*, IEEE Control Systems, 12 (1992), pp. 19–22.

[138] R. Tibshirani, *Regression shrinkage and selection via the lasso*, Journal of the Royal Statistical Society: Series B (Methodological), 58 (1996), pp. 267–288.

[139] C. Torrence and G. P. Compo, *A practical guide to wavelet analysis*, Bulletin of the American Meteorological society, 79 (1998), pp. 61–78.

[140] P. M. Van Den Hof and R. J. Schrama, *Identification and control—closed-loop issues*, Automatica, 31 (1995), pp. 1751–1770.

[141] C. Villani, *Optimal transport: old and new*, vol. 338, Springer Science & Business Media, 2008.

[142] R. B. Vinter, *Minimax optimal control*, SIAM Journal on Control and Optimization, 44 (2005), pp. 939–968.

[143] R. B. Vinter, *Optimal control*, Springer Science & Business Media, 2010.

[144] T. Wang, X. Bao, I. Clavera, J. Hoang, Y. Wen, E. Langlois, S. Zhang, G. Zhang, P. Abbeel, and J. Ba, *Benchmarking model-based reinforcement learning*, arXiv preprint arXiv:1907.02057, (2019).

[145] J. Warga, *Nonsmooth problems with conflicting controls*, SIAM Journal on control and optimization, (1991).

[146] C. J. C. H. Watkins, *Learning from delayed rewards*, PhD thesis, King's College, Cambridge United Kingdom, 1989.

[147] B. Wittenmark, *Stochastic adaptive control methods: a survey*, International Journal of Control, 21 (1975), pp. 705–730.

[148] B. Wittenmark, *Adaptive dual control methods: An overview*, Adaptive Systems in Control and Signal Processing 1995, (1995), pp. 67–72.

[149] J. Yong and X. Y. Zhou, *Stochastic controls: Hamiltonian systems and HJB equations*, vol. 43, Springer Science & Business Media, 1999.

[150] G. Zames, *Feedback and optimal sensitivity: Model reference transformations, multiplicative seminorms, and approximate inverses*, IEEE Transactions on automatic control, 26 (1981), pp. 301–320.

[151] K. Zhou and J. C. Doyle, *Essentials of robust control*, vol. 104, Prentice hall Upper Saddle River, NJ, 1998.

[152] R. Zhu, D. Zeng, and M. R. Kosorok, *Reinforcement learning trees*, Journal of the American Statistical Association, 110 (2015), pp. 1770–1784.

[153] E. Zuazua, *Averaged control*, Automatica, 50 (2014), pp. 3077–3087.